Ensemble et Fusion d'Agents RL pour la Robustesse aux Attaques Adverses

Lucas Schott, Elies Gherbi, Hatem Hajri, Sylvain Lamprier

Ensemble et Fusion d'Agents RL pour la Robustesse aux Attaques Adverses

Lucas Schott[1,2], Elies Gherbi[1], Hatem Hajri[3], Sylvain Lamprier[4,2]

[1] Institut de Recherche Technologique SystemX, Palaiseau, France
[2] MLIA, ISIR, Sorbonne Université, Paris, France
[3] Safran Tech, Châteaufort, France
[4] LERIA, Université d'Angers, France

5 juillet 2025

Résumé

L'Apprentissage par Renforcement Profond (RL) permet d'entraîner des agents autonomes à prendre des décisions dans des environnements complexes, mais reste sensible aux variations et perturbations des observations. L'entraînement adverse permet d'améliorer la robustesse des agents en les exposant à des attaques adverses. L'association des méthodes adverses avec des méthodes ensemblistes et la fusion de modèles peut permettre de renforcer la généralisation de la robustesse. Cet article compare les méthodes d'attaques et d'entraînement adverses tout en explorant les méthodes ensemblistes et la fusion de modèles en RL.

Mots-clés

Robustesse, Attaques Adverses, Apprentissage par Renforcement, Ensemble, Fusion de Modèles

Abstract

Deep Reinforcement Learning (RL) trains autonomous agents for decision-making in complex environments but remains sensitive to condition variations and observation perturbations. Adversarial training enhances robustness by exposing agents to adversarial attacks. Combining this with ensemble methods and model fusion can further improve generalization of robustness. This work compares adversarial attack and training methods while exploring ensembling and model fusion in RL.

Keywords

Robustness, Adversarial Attacks, Reinforcement Learning, Ensemble, Model Merging

1 Introduction

L'Apprentissage par Renforcement Profond (Deep Reinforcement Learning - DRL) a des performances remarquables dans divers domaines, notamment les jeux [21, 28], la robotique [18], la conduite autonome [15] et la gestion de l'énergie [32]. En combinant l'Apprentissage par Renforcement (RL) et les Réseaux de Neurones Profonds (DNN), le DRL permet d'entraîner des politiques neuronales efficaces sur des tâches complexes sans supervision experte.

Toutefois, malgré ses performances remarquables en environnement contrôlé, le DRL reste vulnérable aux variations de conditions et aux perturbations du monde réel. Il peine notamment à être appliqué dû à l'écart entre simulation et application réelle [5]. De plus, les attaques adverses, qui perturbent les entrées pour induire des erreurs de décision, posent un défi supplémentaire [3, 14, 22], mais peuvent aussi être exploitées pour renforcer la robustesse des agents via l'entraînement adverse. Par ailleurs, des approches ensemblistes bien établies et de fusion de modèles qui ont récemment fait leurs preuves peuvent aider à améliorer la généralisation et la résistance de ces agents.

Cet article étudie les méthodes d'attaque et l'impact de l'apprentissage adverses sur l'amélioration de la robustesse des politiques apprises. Il explore les approches ensemblistes et de fusion de modèles pour le RL et étudie la généralisation de la robustesse de ces méthodes avec une étude comparatives expérimentale.

Les principales contributions sont :

- L'étude comparative expérimentale des attaques par gradient sur le RL.
- L'étude des entraînements adverses et de la transférabilité de la robustesse des agents obtenus à d'autre attaques.
- L'étude des approches ensemblistes et de fusion de modèles pour la généralisation de la robustesse en RL.

2 Contexte

2.1 Apprentissage par Renforcement Profond et POMDP

L'Apprentissage par Renforcement (RL) entraîne un agent à interagir avec un environnement en effectuant des actions et en recevant des récompenses, afin d'apprendre une politique optimale $\pi(a|s)$ maximisant la récompense cumulée attendue. Dans un cadre réaliste, l'agent ne perçoit qu'une observation partielle de l'état, définissant un Processus de Décision de Markov Partiellement Observable (POMDP),

formalisé par $\Omega = (S, A, T, R, X, O)$. L'agent reçoit une observation x_t, applique une action a_t et atteint un nouvel état s_{t+1} selon la transition $T(s_+|s, a)$. L'objectif est d'apprendre une politique π^* qui maximise la récompense cumulative pondérée $\sum_t \gamma^t R(s_t, a_t, s_{t+1})$.
Pour résoudre des problèmes complexes à haute dimension, le RL est couplé aux réseaux de neurones profonds (DNNs), formant ainsi le DRL. Ces réseaux ajustent leurs poids via la descente de gradient stochastique (SGD) pour approximer une politique ou une fonction de valeur. Le DRL peut être model-based ou model-free, et s'adapte aux espaces d'actions discrets ou continus. Parmi les algorithmes clé figurent DQN, DDPG, PPO, SAC et d'autres encore, utilisés selon le contexte.

2.2 Problèmes de Robustesse dans le DRL

Le DRL permet aux agents d'apprendre par interaction avec l'environnement, mais il reste vulnérable aux incertitudes environnementales et aux attaques adverses sur les DNNs, compromettant la fiabilité des décisions, notamment en conduite autonome ou cybersécurité.

Incertitudes Environnementales : Le gap de réalité, soit l'écart entre simulation et monde réel, peut induire des comportements sous-optimaux face à des situations imprévues. La nature stochastique des environnements (bruit, perturbations) accentue cette instabilité.

Attaques Adverses sur les DNNs : Les attaques adverses exploitent la sensibilité des DNNs aux perturbations minimes des entrées, faussant les décisions des agents DRL, comme démontré sur les DQNs [1, 13]. Elles reposent sur la génération d'exemples adversariaux :

$$\min_{x'} ||x - x'|| \quad s.t. \quad f_\theta(x) \neq f_\theta(x')$$

Ces attaques peuvent manipuler l'apprentissage ou provoquer des comportements dangereux. Malgré des stratégies de défense comme l'entraînement adverse, la robustesse du DRL face aux menaces évolutives reste un défi majeur.

2.3 Améliorer la Robustesse des Agents DRL

La robustesse du DRL repose sur deux approches principales : Safe RL, qui impose des contraintes de sécurité, et RL Adverse, qui expose l'agent à des perturbations pour renforcer sa résilience.

Safe RL : vise à éviter les actions risquées via des boucliers bloquant les décisions dangereuses [8] ou des filtres excluant ces actions [12]. Des contraintes peuvent aussi être intégrées dans la fonction de perte [30]. Pour stabiliser l'agent face aux variations environnementales, on utilise les architectures récurrentes [29], le lissage aléatoire [4] ou l'exploration bruitée [17]. La distillation défensive [24] réduit la sensibilité aux perturbations mais reste vulnérable aux attaques avancées.

RL Adverse : entraîne l'agent à résister aux perturbations. La détection adverse filtre les entrées corrompues [20], tandis que l'entraînement adverse [22] expose l'agent aux pires scénarios via une optimisation min-max :

$$\min_{\pi} \max_{\delta \in \Delta} J(\pi, \delta)$$

où $J(\pi, \delta)$ est le coût sous perturbation δ.
Cet article se concentre sur l'entraînement adverse, utilisé contre les attaques adverses et les incertitudes environnementales.

2.4 Approches Ensemblistes et Fusion de Modèles

Les approches ensemblistes sont largement utilisées en apprentissage supervisé (par exemple : bagging, boosting, forêts aléatoires) pour réduire la variance et améliorer la robustesse aux perturbations. En RL, l'agrégation de plusieurs politiques s'avère également efficace pour atténuer le sur-apprentissage et mieux gérer la diversité des situations. Parallèlement, la fusion de modèles (aussi appelée soupe de modèles ou weight averaging) consiste à moyenner les poids de plusieurs réseaux de même architecture afin de combiner différents régimes d'apprentissage. Popularisée dans le domaine de la classification d'images et des grands modèles de langage, cette technique préserve une partie des bénéfices de l'ensembling (diversité, amélioration des performances) tout en limitant le coût d'inférence, puisqu'un seul modèle fusionné est finalement utilisé. Toutefois, elle réduit la capacité à estimer l'incertitude, l'agent ne pouvant plus s'appuyer sur la variance d'un ensemble de décisions.

3 Formalisation et Périmètre

Cette section unifie les formulations de l'apprentissage robuste adverse pour le DRL.

3.1 Problème de Robustesse en RL

Le problème d'optimisation s'écrit :

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\Omega \sim \Phi(\cdot|\pi)} \mathbb{E}_{\tau \sim \pi^\Omega}[R(\tau)]$$

où Φ est la distribution des environnements possibles, π^Ω celle des trajectoires sous π, et $R(\tau)$ la récompense cumulée. À l'entraînement, $\Phi(\Omega|\pi)$ est inconnue et l'agent s'adapte à un unique MDP.
Pour formaliser la robustesse, on introduit $\phi(\Omega)$ une fonction d'altération qui modifie certaines composantes du MDP. On introduit alors une distribution d'altération $\Phi(\phi|\pi)$:

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\phi \sim \Phi(\cdot|\pi)} \mathbb{E}_{\tau \sim \pi^{\phi,\Omega}}[R(\tau)]$$

Nous distinguons les cas suivant : - Altérations des observations (ϕ_O) : Affectent la perception de l'agent sans modifier l'état réel, pouvant résulter d'attaques, d'écarts simulation-réalité ou de capteurs défaillants. - Altérations des dynamiques (ϕ_T) : Modifient les transitions, changeant directement les effets des actions.
Les autres altérations ($\phi_S, \phi_A, \phi_X, \phi_R$) sont hors du champ de cette étude.

3.2 Attaques Adverses et RL Robuste

Inspiré de l'optimisation robuste distributionnelle (DRO) [26], le problème devient :

$$\pi^* = \arg\max_{\pi} \min_{\widetilde{\Phi} \in \mathcal{R}} \mathbb{E}_{\phi \sim \widetilde{\Phi}(\cdot|\pi)} \mathbb{E}_{\tau \sim \pi^{\phi,\Omega}}[R(\tau)]$$

où $\mathcal{R}$ est l'ensemble des distributions de perturbations. Trop large, $\mathcal{R}$ rend π inefficace ; trop restreint, il limite la généralisation.
Au lieu d'optimiser sur toutes les perturbations, nous utilisons l'entraînement adverse, où un agent adverse ξ maximise l'impact des perturbations sur π :

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\tau \sim \pi^{\xi^*,\Omega}}[R(\tau)] \quad (1)$$
$$s.t. \qquad \xi^* = \arg\min_{\xi} \Delta^{\pi,\Omega}(\xi)$$

où $\Delta^{\pi,\Omega}(\xi)$ est l'objectif de ξ, et $\pi^{\xi,\Omega}(\tau)$ la distribution des trajectoires sous π dans un POMDP modifié.
ξ altère les transitions $\tau_t = (s_t, x_t, a_t, s_{t+1})$, affectant l'observation O ou la dynamique T. L'entraînement adverse prépare ainsi π aux pires scénarios.
Dans la suite de cet article nous menons une analyse des attaques adverses contre les agents RL, en nous concentrant sur les attaques de observations en comparant leurs objectifs et leurs applications. [27] contient une taxonomie incluant les attaques des dynamiques de l'environnement.

4 Attaques sur les Observations

Nous étudions ici la catégorie de perturbation affectant la fonction d'observation O du POMDP, inspirée des attaques adverses en apprentissage supervisé. L'objectif est de modifier les entrées perçues par l'agent π, simulant ainsi par exemple des erreurs de capteurs. Cette section examine les principales attaques adverses sur les observations qu'on nomme les attaques par gradient. Il existe d'autre types d'attaques, comme les attaques black-box et politiques adverses, mais que nous n'allons pas étudier dans le cadre de ces travaux.

4.1 Attaques Supervisée

Les attaques par gradient cherchent à maximiser l'écart entre la sortie attendue et celle obtenue en présence d'une perturbation. Ces attaques ont d'abord été développés pour les modèles des classification et de régression supervisés, elles exploitent le gradient du modèle pour générer des perturbations efficaces. Les méthodes notables sont :
– FGM [9] qui est la méthode de base qui calcul le gradient d'une erreur sur la sortie des modèles pour l'accentuer en modifiant toute l'observation x d'une certaine magnitude de perturbation δ,

$$x' = x + a^{\xi,X} \quad \text{with} \quad a^{\xi,X} = \nabla_x \mathcal{L}\big(f(x), y\big)$$

– PGD, BIM, MI-FGSM, MIM, NM-FGM [19, 16] sont des méthodes se basant sur le même principe que FGM avec différentes variations. Ces méthodes peuvent être itératives (nécessiter plusieurs steps intermédiaires pour générer la perturbation), et utiliser des mécanismes de conservation de l'inertie d'un step ou d'une perturbation à l'autre.
– Les Auto Attacks [7, 6] sont des méthodes qui vont appliquer plusieurs des attaques précédentes ainsi que d'autres attaques black-box, pour modifier l'observation x d'une certaine magnitude de perturbation δ en suivant démarche d'exploration aléatoire avec plusieurs initialisations et différents paramètres pour trouver la perturbation la plus proche de l'optimal, la perturbation qui va changer le plus la décision de l'agent.
– C&W, DeepFool, JSMA, VFGA [2, 23, 25, 10] dont le but de minimiser la quantité de perturbation produite. En cherchant le point de frontière de décision le plus proche.

4.2 Adaptation des attaques au RL

Toutes ces attaques ayant été développée dans le domaine supervisé, elles peuvent être appliqué au RL. Une spécificité du RL est qu'un agent peut être composé de plusieurs réseaux de neurones. Et les attaques ci-dessus peuvent être déclinés et appliqués de divers façons sur les différents réseaux de neurones d'un agent. On peut par exemple appliquer une attaque sur un modèle qui prend en entrée une observation et donne en sortie une action, comme Q ou Π (comme dans tous les algorithmes DQN, PPO, SAC, DDPG, etc) comme on le ferait en attaquant un modèle de classification ou de régression :

$$x' = x + a^{\xi,X} \quad \text{with} \quad a^{\xi,X} = \nabla_x \mathcal{L}\big(Q(x), y\big)$$
$$x' = x + a^{\xi,X} \quad \text{with} \quad a^{\xi,X} = \nabla_x \mathcal{L}\big(\Pi(x), y\big)$$

Aussi certains agents RL sont composés d'un modèle V, qui calcule la valeur d'une observation (comme l'algorithme PPO) sur lequel on peut aussi appliquer des attaques :

$$x' = x + a^{\xi,X} \quad \text{with} \quad a^{\xi,X} = \nabla_x \mathcal{L}\big(V(x), y\big)$$

D'autres agents encore, qu'on appelle Acteur-Critique, sont composés d'un modèle Π qui choisit l'action suivi d'un modèle Q qui calcul la valeur du couple observation-action (comme les algorithmes SAC et DDPG) dans ce cas-là, on peut choisir d'attaques soit uniquement le modèle Π comme précédemment. Soit le modèles Q :

$$x' = x + a^{\xi,X} \quad \text{with} \quad a^{\xi,X} = \nabla_x \mathcal{L}\big(Q(x, a), y\big)$$

Ou alors les deux modèles en même temps, en propagent le gradient de l'input a de Q à la sortie de Π :

$$a^{\xi,A} = \nabla_a \mathcal{L}\big(Q(x, a), y\big)$$
$$a^{\xi,X} = \nabla_x \mathcal{L}\big(\Pi(x), \Pi(x) + a^{\xi,A}\big)$$
$$x' = x + a^{\xi,X}$$

Par la suite, on dénotera le nom des attaques suivi de `_D` pour les attaques appliqués sur un simple réseau à action discrètes Q ou Π, `_C` pour les attaques appliqués sur un simple réseau à action continue Π, `_V` pour les attaques appliqués sur un de valeur V, et `_QAC` pour les attaques appliqués sur les deux réseaux d'un agent Acteur-Critic, les réseaux Q et Π. Dans les expérimentations on utilisera notamment, les attaques `FGM_D`, `FGM_C`, `FGM_V` et `FGM_QAC`. Les méthodes `FGM_D`, `FGM_C` peuvent être utilisée de manière `tageted` ou `untargeted`. Et les méthodes `FGM_D`, `FGM_C` peuvent être appliqué en cherchant à maximiser (`MAX`) ou à minimiser (`MAX`) la valeur prédite.

5 Entraînement Adverses

L'entraînement adverse en RL renforce la robustesse des agents en les exposant à des perturbations durant l'apprentissage, simulant ainsi des attaques réelles. Basé sur un jeu min-max [22], il prépare l'agent aux pires scénarios en perturbant ses observations ou la dynamique de l'environnement.
Un agent π_1, entraîné jusqu'à convergence dans Ω, est confronté à un adversaire ξ_1. Si ξ_1 nécessite un apprentissage (ex. politique adverse), il est pré-entraîné dans Ω^{π_1}, sinon, les attaques par gradient exploitent directement les gradients de π_1.
Une approche simple consiste à entraîner π face à un adversaire fixe ξ_1, limitant la divergence hors distribution, mais risquant un sur-ajustement. D'autres stratégies existent : entraînement continu, alterné, ou via Fictitious Self Play [11].

6 Ensemble et Fusion de Modèles

L'ensembling en apprentissage par renforcement combine plusieurs politiques en une seule pour améliorer la robustesse et la généralisation, tout en réduisant le surapprentissage et la vulnérabilité aux attaques adverses. Des méthodes comme Ensemble Policy Optimization (EPO) [31] exploitent cette approche pour atténuer les modes d'échec dans des environnements à forte variance. Toutefois, cette technique augmente le coût de calcul, car elle nécessite l'inférence de plusieurs modèles à chaque décision.
Étant donné une observation x, chacun des agent de paramètres θ_i prend une décision $a_i \in \mathcal{A}$:

$$a_i = \pi(x, \theta_i)$$

Les décisions des différents agents sont alors agrégées en une seule.

$$a_{ens} = \arg\max_{a \in \mathcal{A}} \sum_{i=1}^{M} \big(a = \pi(x, \theta_i)\big).$$

Une alternative récente, la fusion de modèles, issue de la classification d'images et des modèles génératifs, consiste à moyenner les poids de plusieurs modèles de même architecture pour en produire un unique.

$$\overline{\theta} = \frac{1}{M} \sum_{i=1}^{M} \theta_i.$$

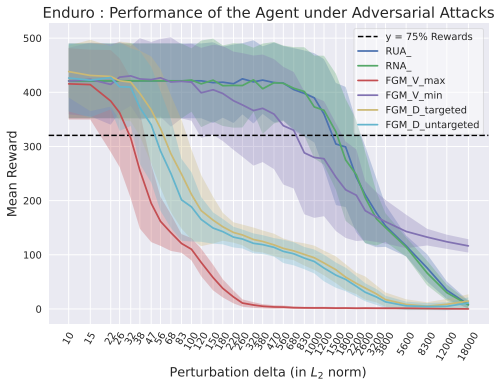
La décision est alors directement issue de l'inférence de ce modèle unique fusionné $\overline{\theta}$

$$a_{fusion} = \pi(x, \overline{\theta}).$$

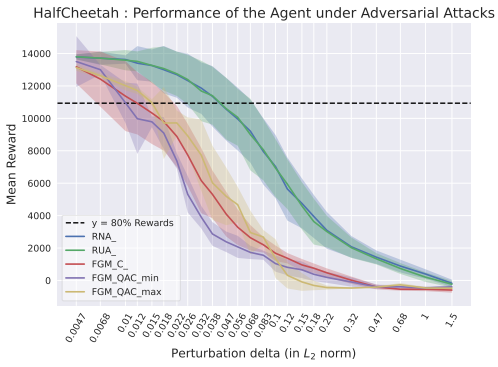
Cette méthode peut conserver certains bénéfices de l'ensembling (diversité d'apprentissage, atténuation des faiblesses individuelles) sans accroître le coût d'inférence. En revanche, elle supprime la capacité à quantifier l'incertitude, n'offrant plus qu'une seule décision par entrée. La fusion de modèles a pour le moment fait ses preuves sur de très gros modèles de classification et génératif, nous allons explorer son utilisation avec des petits modèles de décision.

7 Expérimentations

Pour les expérimentations suivantes, nous nous concentrons sur les attaques adverses faisant de la perturbation des observations. Nous travaillons sur 2 environnements de RL :
– Un environnement de jeux vidéo Atari `ALE/Enduro-v5`. Cet environnement est à observations sous forme d'images, et à actions discrètes. Nous avons entraîné un agent DRL sur cet environnement en utilisant l'algorithme PPO. Ensuite, nous appliquons 6 attaques différentes sur cet agent : les attaques `RUA`, `RNA`, qui sont des attaques aléatoire uniforme et normale, `FGM_D` targeted et untargeted, `FGM_V` min et max, sur un intervalle de puissances de perturbations δ donné, en norme L_2
– Un environnement de simulation physique d'agent articulé Mujoco `HalfCheetah-v5`. Cet environnement est à observations sous forme de vecteur d'informations, et à actions continues. Nous avons entraîné un agent DRL sur cet environnement en utilisant l'algorithme SAC. Ensuite, nous appliquons 5 attaques différentes sur cet agent, les attaques `RUA`, `RNA`, `FGM_C` targeted et untargeted, `FGM_QAC` min et max, sur un intervalle de puissances de perturbations δ donné, en norme L_2.



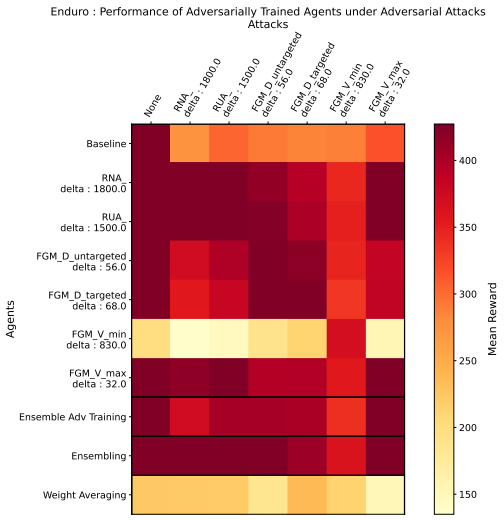
(a) Perturbation de l'agent sur `Enduro-v5`.



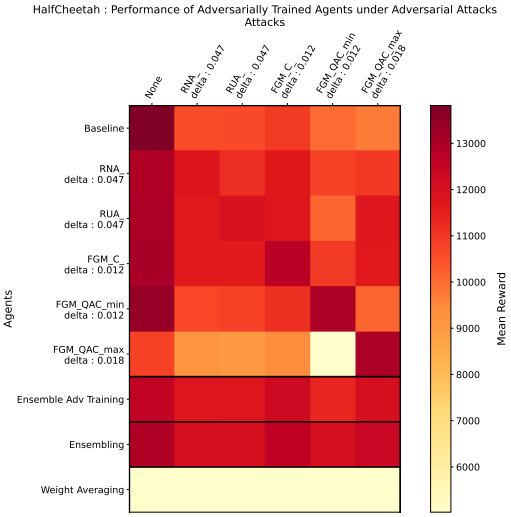
(b) Perturbation de l'agent sur `HalfCheetah-v5`.

FIGURE 1 – Agents attaqués durant l'évaluation par différentes attaques en faisant varier δ.

Attaques de l'Agent Initial : Les figures 1a et 1b montrent l'impact des perturbations selon la puissance δ. Les attaques aléatoires (RUA, RNA) requièrent de fortes pertur-



(a) Matrice d'évaluation des agents obtenus sur `Enduro-v5`.



(b) Matrice d'évaluation des agents obtenus sur `HalfCheetah-v5`.

FIGURE 2 – Matrice d'Evaluation des Agents, évalués face à chacune des attaques avec la puissance de perturbation associée δ.

bations pour dégrader l'agent, tandis que les attaques par gradient (FGM...) y parviennent avec de faibles valeurs de δ.

Pour déterminer la puissance à employer lors de l'entraînement adversaire, nous fixons un seuil de 80% (ligne pointillée en noir) des performances nominales. Nous retenons alors la plus petite valeur de δ qui fait passer l'agent sous ce seuil pour chaque attaque. Ainsi, nous produisons cinq agents entraînés de façon adverse (un par attaque) et conservons l'agent initial, soit six au total.

Matrice d'Evaluation des Agents Finutinés et Combinés : Nous évaluons ensuite ces six agents individuellement contre chaque attaque, ainsi qu'en mode ensembliste et via la fusion de modèles. Les figures 2a et 2b présentent les performances obtenues : en ligne i, l'agent évalué, et en colonne j, l'attaque appliquée (avec sa puissance δ). La première colonne et la diagonale sont plus sombres, indiquant que chaque agent se comporte mieux sans perturbation ou face à l'attaque rencontrée lors de son entraînement – sans généraliser à d'autres attaques.

Toutefois, les trois dernières lignes montrent que l'agent **Ensemble Adv Training** (entraîné simultanément sur toutes les attaques) et l'agent **Ensembling** affichent une robustesse généralisée. À l'inverse, l'agent **Weight Averaging** (fusion de tous les autres modèles) ne généralise pas à toutes les perturbations et perd en performance même en condition nominale.

8 Conclusion et Perspectives

Nous avons présenté et comparé diverses attaques adverses ciblant les agents d'apprentissage par renforcement, ainsi que des stratégies d'entraînement adverses pour renforcer leur robustesse. Les approches ensemblistes sont performantes et robustes, mais sont très coûteuses en calcules, la fusion de modèles pourrait être prometteuses pour améliorer la résistance aux attaques, mais nécessite encore de l'exploration pour son utilisation sur des petits modèles de décisions.

Perspectives :

- **Attaques multiples.** Combiner plusieurs types d'attaques (observations, dynamiques, etc.) dans un scénario unifié.
- **Adaptation continue.** Ajuster dynamiquement la puissance et la fréquence des perturbations en situation réelle.
- **Fusions de modèles.** Trouver un moyen de faire fonctionner cette approche sur de petits modèles de décision.

Remerciements

Ce travail a été soutenu par le gouvernement français dans le cadre du programme "France 2030", au sein de l'Institut de Recherche Technologique SystemX, dans le cadre du programme Confiance.ai. Ces travaux ont bénéficié d'un accès aux moyens de calcul de l'IDRIS au travers de l'allocation de ressources AD011015866 attribuée par GENCI.

Références

[1] Vahid Behzadan and Arslan Munir. Vulnerability of deep reinforcement learning to policy induction attacks. In Petra Perner, editor, *Machine Learning and Data Mining in Pattern Recognition*. Springer International Publishing, 2017.

[2] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, May 2017.

[3] Tong Chen et al. Adversarial attack and defense in reinforcement learning-from ai security view. *Cybersecurity*, December 2019.

[4] Jeremy Cohen et al. Certified adversarial robustness via randomized smoothing. In *international confe-*

rence on machine learning, pages 1310–1320. PMLR, 2019.

[5] Jack Collins et al. Quantifying the reality gap in robotic manipulation tasks. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6706–6712. IEEE, 2019.

[6] 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.

[7] Yinpeng Dong et al. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9185–9193, 2018.

[8] Kunal Garg et al. Learning safe control for multi-robot systems : Methods, verification, and open challenges. *Annual Reviews in Control*, 57 :100948, 2024.

[9] Ian J. Goodfellow et al. Explaining and harnessing adversarial examples, March 2015.

[10] Hatem Hajri et al. Neural adversarial attacks with random noises. *International Journal on Artificial Intelligence Tools*, 2022.

[11] Johannes Heinrich et al. Fictitious self-play in extensive-form games. In *International conference on machine learning*. PMLR, 2015.

[12] Kai-Chieh Hsu et al. The safety filter : A unified view of safety-critical control in autonomous systems. *Annual Review of Control, Robotics, and Autonomous Systems*, 7, 2023.

[13] Sandy Huang et al. Adversarial attacks on neural network policies, February 2017.

[14] Inaam Ilahi et al. Challenges and countermeasures for adversarial attacks on deep reinforcement learning. *IEEE Transactions on Artificial Intelligence*, April 2022.

[15] B. Ravi Kiran et al. Deep reinforcement learning for autonomous driving : A survey, January 2021.

[16] Ezgi Korkmaz. Nesterov momentum adversarial perturbations in the deep reinforcement learning domain. In *International Conference on Machine Learning, ICML*, 2020.

[17] Aashish Kumar. *Enhancing performance of reinforcement learning models in the presence of noisy rewards*. Thesis, April 2019.

[18] Sergey Levine et al. End-to-end training of deep visuomotor policies, April 2016.

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

[20] Jan Hendrik Metzen et al. On detecting adversarial perturbations, February 2017.

[21] Volodymyr Mnih et al. Human-level control through deep reinforcement learning. *Nature*, February 2015.

[22] Janosch Moos et al. Robust reinforcement learning : A review of foundations and recent advances. *Machine Learning and Knowledge Extraction*, 4(1) :276–315, 2022.

[23] Seyed-Mohsen Moosavi-Dezfooli et al. Deepfool : a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2574–2582, 2016.

[24] Nicolas Papernot et al. Distillation as a defense to adversarial perturbations against deep neural networks, March 2016.

[25] Nicolas Papernot et al. The limitations of deep learning in adversarial settings. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, March 2016.

[26] Hamed Rahimian and Sanjay Mehrotra. Distributionally robust optimization : A review. *arXiv preprint arXiv :1908.05659*, 2019.

[27] Lucas Schott et al. Robust deep reinforcement learning through adversarial attacks and training : A survey. *arXiv preprint arXiv :2403.00420*, 2024.

[28] David Silver et al. Mastering the game of go with deep neural networks and tree search. *Nature*, January 2016.

[29] Daan Wierstra et al. Solving deep memory pomdps with recurrent policy gradients. In *Artificial Neural Networks–ICANN 2007 : 17th International Conference, Porto, Portugal, September 9-13, 2007, Proceedings, Part I 17*, pages 697–706. Springer, 2007.

[30] Qisong Yang et al. Wcsac : Worst-case soft actor critic for safety-constrained reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10639–10646, 2021.

[31] Zhengyu Yang et al. Towards applicable reinforcement learning : Improving the generalization and sample efficiency with policy ensemble. *arXiv preprint arXiv :2205.09284*, 2022.

[32] Dongxia Zhang et al. Review on the research and practice of deep learning and reinforcement learning in smart grids. *CSEE Journal of Power and Energy Systems*, September 2018.