Rapport d'analyse de Soft Actor-Critic

Simon Groc, Yassin Lahbib

[1] Master AI2D Sorbonne Université

Abstract :

Les algorithmes d'*apprentissage par renforcement* hors-politique, en particulier *Soft Actor-Critic (SAC)*, démontrent une grande efficacité dans les espaces d'actions continues. Leur adaptation à des espaces discrets (*Discrete SAC*, ou *DSAC*) soulève cependant plusieurs questions : la non-stationnarité des cibles Q, l'alignement des décisions entre l'*actor* et le *critic*, ainsi que le rôle de la régularisation par l'entropie. Dans ce travail, nous implémentons, au sein de la bibliothèque *BBRL*, six algorithmes classiques (*DQN*, *Double DQN*, *DDPG*, *TD3*, *SAC* continu et *DSAC*), et nous nous focalisons sur l'étude du comportement de *DSAC* dans l'environnement *CartPole-v1*.
Nous définissons un taux d'accord entre l'*actor* et le *critic*, comme la proportion de pas où l'action la plus probable selon la politique coïncide avec celle de plus haute *Q-value*. Nous comparons ce taux à une référence aléatoire et observons qu'il reste proche de 50 %, même après plusieurs milliers d'itérations, ce qui indique une dissociation persistante entre les deux modules. Par ailleurs, nous évaluons l'influence de la mise à zéro du coefficient d'entropie sur l'apprentissage, et montrons qu'elle n'entraîne pas d'amélioration notable de leur cohérence.

Contents

1 Introduction

Ce rapport s'inscrit dans le cadre de l'UE *Projet ANDROIDE* du Master *AI2D* (Intelligence Artificielle et Algorithmes de Décision), sous la supervision du professeur Olivier Sigaud. L'objectif principal de ce projet est de se familiariser avec la bibliothèque **BBRL** (BlackBoard Reinforcement Learning), développée par le laboratoire **ISIR** (Institut des Systèmes Intelligents et de Robotique), et de l'utiliser pour implémenter différents algorithmes d'apprentissage par renforcement.

La bibliothèque BBRL permet de concevoir des algorithmes de *Reinforcement Learning* à un niveau d'abstraction élevé. Elle repose sur une architecture modulaire fondée sur le concept d'*agents* interagissant dans un *workspace*, à l'image de l'approche « tableau noir » des systèmes multi-agents, d'où elle tire son nom.

Dans un premier temps, nous avons réimplémenté plusieurs algorithmes classiques de l'apprentissage par renforcement : **DQN** (Deep Q-Learning), **DDQN** (Double DQN), **DDPG** (Deep Deterministic Policy Gradient), **TD3** (Twin Delayed Deep Deterministic Policy Gradient), **SAC** (Soft Actor–Critic), ainsi qu'une version discrète que nous avons développée nous-mêmes : **DSAC** (Discrete Soft Actor–Critic).

La seconde partie du projet a été consacrée à l'analyse expérimentale de DSAC. Nous avons notamment étudié la convergence entre l'*actor* et le *critic*, et comparé les performances obtenues à celles du DQN sur des environnements discrets.

L'ensemble du code, des notebooks explicatifs et des résultats expérimentaux est disponible dans le dépôt GitHub suivant : `https://github.com/Hello-IA/PANDROIDE`

2 Contribution dans la littérature

2.1 Algorithmes classiques d'apprentissage par renforcement

Les premiers algorithmes d'apprentissage par renforcement tabulaire, tels que *Q-Learning* ou *SARSA*, permettent à un agent d'apprendre une politique optimale à partir d'une table de valeurs. Ces méthodes fonctionnent bien pour des environnements discrets simples, mais ne s'adaptent pas aux grands espaces d'états ou d'actions.

2.2 Limites des approches classiques

Les méthodes tabulaires ne s'appliquent pas aux environnements continus, tandis que les méthodes par *policy gradient* nécessitent souvent des techniques de régularisation complexes et sont sensibles aux hyperparamètres.

2.3 Soft Actor-Critic (SAC)

L'algorithme *Soft Actor-Critic* (SAC) est une méthode d'apprentissage par renforcement hors-politique basée sur un compromis entre récompense et entropie. Il est particulièrement efficace dans les environnements continus, grâce à l'optimisation simultanée de l'*actor* et du *critic*. Cependant, la version originale est limitée aux espaces d'actions continues.

2.4 Vers une version discrète de SAC

Récemment, des travaux proposent des adaptations de *SAC* à des environnements discrets, comme dans l'article *"Soft Actor-Critic for Discrete Action Settings"* de Christodoulou (2019). Cet article apporte les modifications nécessaires pour permettre à *SAC* de fonctionner dans des contextes à actions discrètes.

2.5 Notre contribution

Dans le cadre de ce projet, nous nous intéressons à la version discrète de *SAC*, et plus précisément à l'alignement entre les décisions prises par l'*actor* et les valeurs estimées par le *critic*. Lors de l'évaluation, nous comparons l'action choisie par l'*actor* (c'est-à-dire celle ayant la probabilité maximale) avec celle que le *critic* estime comme ayant la plus grande *Q-value*. Cette comparaison permet d'étudier le niveau de cohérence entre les deux composantes du modèle, que nous appelons taux d'accord, et d'évaluer l'impact de certains choix d'hyperparamètres sur cette métrique.

2.6 Les précédents travaux de recherche

Plusieurs contributions récentes ont tenté d'adapter le cadre du *Soft Actor-Critic* (SAC) aux espaces d'actions discrètes, chacune soulignant à sa manière les défis de convergence liés au compromis récompense–entropie et à la non-stationnarité des cibles Q.

- **Christodoulou (2019) [1]** propose une version *Discrete SAC* où la politique stochastique est construite via une *softmax* appliquée aux *Q-values*. Bien que cette approche conserve l'idée de maximiser l'entropie, elle souffre d'instabilités au cours de l'apprentissage.
- **Yuan et al. (2021) [2]** proposent d'auto-régler le coefficient d'entropie α et d'ajouter une pénalité de type *KL divergence* entre l'ancienne et la nouvelle politique. Cette régularisation limite les sauts brusques dans les politiques apprises, mais complexifie l'optimisation.
- **Revisiting Discrete SAC (2024) [3]** identifie un phénomène de *target shift* — un décalage entre les poids du réseau Q et ceux du *target network* — et montre qu'un gel plus fréquent de ces derniers (via un *Polyak averaging* avec un faible taux) améliore la stabilité du processus d'apprentissage.

Dans notre travail, nous ne cherchons pas à améliorer l'apprentissage de DSAC, mais plutôt à observer si l'*actor* et le *critic* convergent vers des décisions similaires. En théorie, leurs sorties devraient être proches à un facteur près : le *critic* retourne des *Q-values* tandis que l'*actor* retourne une distribution de probabilité sur les actions. Nous explorons aussi l'effet d'une mise à zéro du coefficient d'entropie sur le comportement de l'apprentissage.

3 Algorithmes implémentés

3.1 Reward Prediction Error

Pour que le réseau apprenne, il a besoin pour chaque échantillon i de minimiser le RPE (*Reward Prediction Error*) :

$$\delta_i = r_{i+1} + \gamma \max_a Q_\phi(s_{i+1}, a) - Q_\phi(s_i, a_i)$$

où r représente la **récompense**. L'agent apprend en comparant la *Q-value* de s_i et a_i (où i correspond au i-ème échantillon de notre *replay buffer*) avec la meilleure *Q-value* de l'état s_{i+1}, c'est-à-dire l'état dans lequel on arrive après avoir effectué l'action a_i à partir de l'état s_i. On multiplie la *Q-value* future par γ pour indiquer que les estimations futures sont moins importantes, et on ajoute la récompense de l'action réellement prise.

Cela se résume intuitivement par : Ton action a_i avait en réalité une valeur de $r_{i+1} + \gamma \times$ (meilleure valeur future), alors que tu pensais que c'était $Q(s_i, a_i)$. Apprends de cette erreur pour mieux prévoir la prochaine fois. Si la RPE se minimise et tend vers 0, cela signifie que :

$$r_{i+1} + \gamma \times (\text{meilleure valeur future}) = Q(s_i, a_i)$$

ce qui indique que notre modèle sait prédire la meilleure action à prendre en anticipation du futur.

3.2 Deep Q-Learning

C'est l'algorithme de *reinforcement learning* le plus simple que nous avons implémenté. Il consiste à reprendre le *Q-learning* classique en remplaçant la *Q-table* par un réseau de neurones : le *Q-network*. Ce réseau prend en entrée un état et renvoie en sortie les *Q-values* pour chaque action possible. Il y a donc autant de sorties que d'actions. Cela permet de généraliser même lorsque l'espace des états/actions devient trop grand pour être représenté explicitement.

3.3 Double Deep Q-Learning

L'algorithme *Double DQN* corrige un biais du *DQN* classique : la surestimation des *Q-values*. En effet, dans *DQN*, le même réseau est utilisé pour choisir et évaluer l'action, ce qui peut induire un biais. *Double DQN* propose de décorréler cette sélection et cette évaluation : l'action optimale est choisie à l'aide du *Q-network* principal, mais sa valeur est estimée via le *target network*. Cette séparation permet de limiter la surestimation et rend l'apprentissage plus stable. Dans nos expériences, cette version s'est montrée plus fiable.

3.4 Deep Deterministic Policy Gradient (DDPG)

L'algorithme *DDPG* est conçu pour les environnements à états et actions continues. Il repose sur deux réseaux :

- Un *actor* qui propose une action donnée un état ;
- Un *critic* qui estime la *Q-value* de cette action dans cet état.

Le *critic* est entraîné via une perte de type moindres carrés, et l'*actor* est mis à jour en suivant le gradient de la *Q-value*.

3.5 Twin Delayed Deep Deterministic Policy Gradient (TD3)

TD3 améliore *DDPG* avec :

- Deux *critics* : on utilise le minimum des deux estimations pour éviter la surestimation.

- Une politique cible bruitée (*target noise*) : cela empêche d'apprendre des pics non réalistes dans les *Q-values*.

3.6 Soft Actor-Critic (SAC)

SAC introduit l'entropie dans la fonction objectif pour encourager l'exploration. Contrairement à *TD3*, il utilise une politique stochastique. Cela permet une meilleure robustesse et évite les minima locaux.

$$J(\alpha) = \mathbb{E}_{s_t \sim \mathcal{D}} \left[\mathbb{E}_{a_t \sim \pi_\theta(\cdot|s_t)} \left[-\alpha \log \pi_\theta(a_t|s_t) - \alpha \mathcal{H}_0 \right] \right]$$

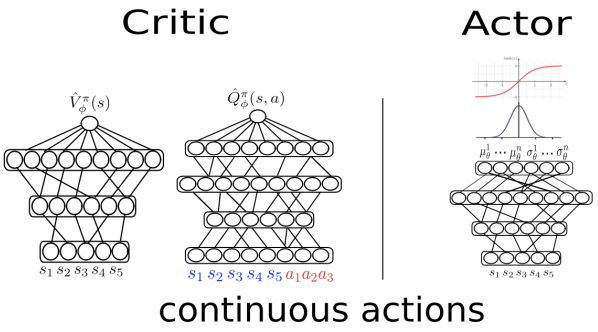

Figure 1. Structure des réseaux *actor* et *critic* dans SAC

3.7 Discrete Soft Actor-Critic (DSAC)

Dans *DSAC*, les états et actions sont discrets. Les deux réseaux ont la même structure :

- Le *actor* produit une distribution de probabilité sur les actions ;
- Le *critic* renvoie les *Q-values* de chaque action.

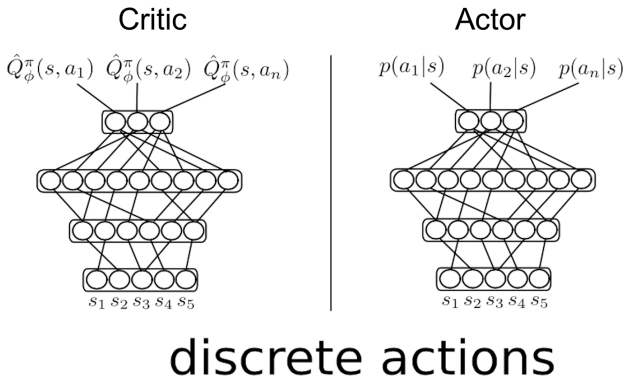

Figure 2. Structure des réseaux pour l'algorithme *DSAC*

4 Résultats

4.1 Comparaison entre DQN et Double DQN

Dans le cadre du travail sur le *notebook* dédié à *DQN*, nous comparons les performances de l'algorithme *DQN* classique (algorithme 1, en bleu sur la figure) avec celles de sa version améliorée, *Double DQN* (algorithme 2, en orange).

Pour cette comparaison, nous utilisons un test t de *Welch* à chaque étape d'évaluation. Ce test statistique permet de déterminer si la différence de performance entre les deux algorithmes est significative à un instant donné. Un point noir est affiché au-dessus des courbes lorsque cette différence est jugée significative.

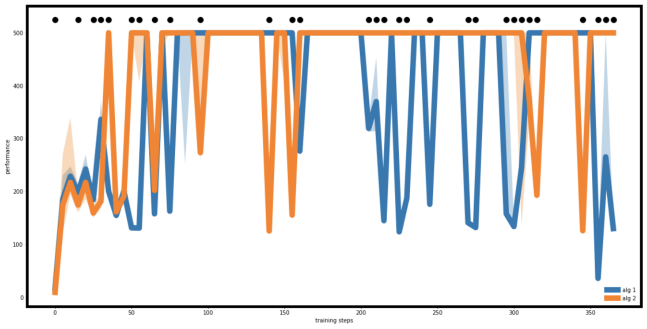

Figure 3. Comparaison des performances entre *DQN* et *DDQN* avec test t de *Welch* à chaque évaluation.

Nous observons que *Double DQN* obtient généralement de meilleures performances que *DQN*, notamment grâce à une réduction du biais de sur-estimation des *Q-values*. Cette différence, confirmée statistiquement sur plusieurs évaluations, souligne l'intérêt d'utiliser *DDQN* dans les environnements où ce biais peut ralentir ou perturber l'apprentissage.

4.2 Taux d'accord entre acteur et critique dans DSAC

Dans l'algorithme *Discrete Soft Actor-Critic* (*DSAC*), l'acteur et le critique sont structurés de manière identique : ce sont deux réseaux de neurones prenant en entrée un état et produisant un vecteur de taille égale au nombre d'actions disponibles.

La différence réside dans l'interprétation de ces vecteurs : l'acteur applique une fonction *softmax* pour produire une distribution de probabilité sur les actions, tandis que le critique considère directement les sorties comme des estimations de *Q-values*. Les deux réseaux produisent donc des sorties analogues, qui pourraient être proportionnelles.

Afin d'analyser leurs comportements respectifs, nous avons comparé, lors de chaque phase d'évaluation, l'action la plus probable selon l'acteur avec celle qui possède la plus grande *Q-value* selon le critique.

Nous calculons ainsi un *taux d'accord*, défini comme la proportion de cas où l'action choisie par l'acteur coïncide avec celle évaluée comme optimale par le critique. Ce taux permet d'évaluer si les deux modules convergent vers des décisions cohérentes.

Figure 4. Évolution du taux d'accord entre l'acteur et le critique sur 10 exécutions indépendantes de l'algorithme *DSAC*.

On observe que le taux d'accord reste globalement autour de 50 %. Dans l'environnement *CartPole-v1*, qui ne propose que deux actions, un tel taux peut être atteint aléatoirement, ce qui peut surprendre après de nombreuses itérations d'entraînement.

Cela peut s'expliquer par la différence de rôle entre les deux réseaux : l'acteur maximise l'entropie et favorise l'exploration, tandis que le critique se concentre sur l'évaluation de la valeur des actions. Il est donc normal qu'ils ne s'alignent pas systématiquement.

Cependant, ce faible taux pourrait également indiquer une implémentation imparfaite de l'algorithme. Une erreur dans la définition des sorties ou dans l'apprentissage des réseaux pourrait perturber la cohérence des décisions. Ces observations nous invitent donc à la prudence dans l'interprétation des résultats et à envisager des analyses complémentaires.

4.3 Comparaison entre DSAC et DQN

Étant donné que *DSAC* fonctionne avec des actions discrètes tout comme *DQN*, il nous a semblé pertinent de comparer les deux dans un même environnement. Pour ce faire, nous avons exécuté les deux algorithmes dans l'environnement *CartPole-v1* afin d'évaluer les performances de notre implémentation de *DSAC* face à un algorithme classique d'apprentissage par renforcement.

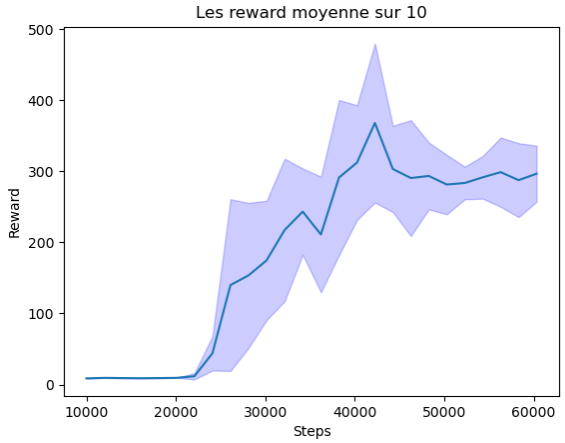

Figure 5. Récompenses obtenues par *DSAC* dans l'environnement *CartPole-v1*.

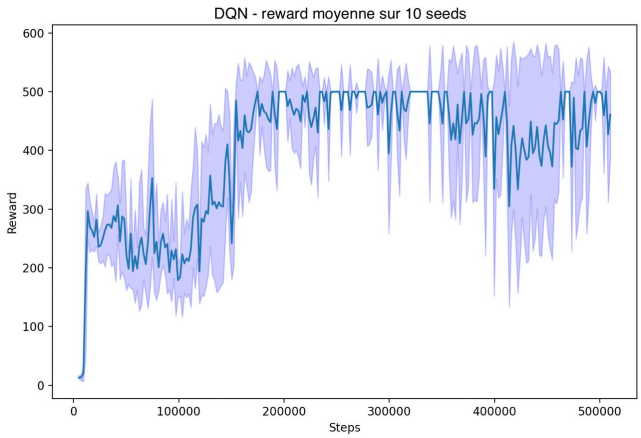

Figure 6. Récompenses obtenues par *DQN* dans l'environnement *CartPole-v1*.

On observe que *DQN* atteint rapidement de hautes récompenses, souvent proches du maximum (500), tandis que *DSAC* met davantage de temps à apprendre et ne parvient jamais à obtenir des performances significatives. Cela suggère que notre implémentation de *DSAC* est, dans sa forme actuelle, peu efficace, malgré l'optimisation des hyperparamètres via la bibliothèque *Optuna*.

Plusieurs hypothèses peuvent expliquer ce comportement :

- D'une part, la version classique de *DSAC*, sans ajustement spécifique, est reconnue comme peu performante dans la littérature.
- D'autre part, il est possible qu'une erreur dans notre implémentation (dans l'apprentissage de l'acteur ou du critique) limite l'apprentissage du modèle.

4.4 Comparaison entre DSAC avec et sans entropie

Nous avons également analysé l'influence de l'entropie sur les performances de *DSAC*.

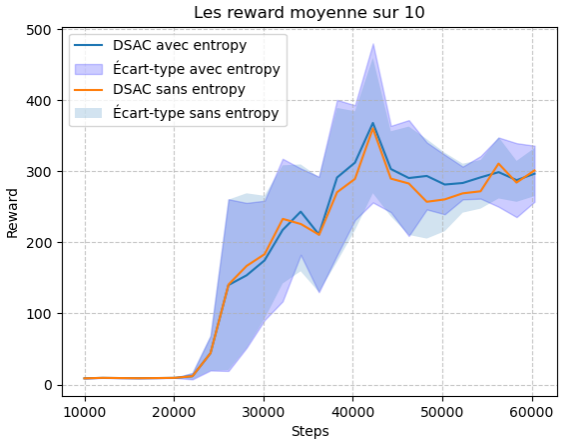

Figure 7. Comparaison des courbes de récompense pour *DSAC* avec et sans entropie.

Les résultats montrent peu de différence entre les versions avec et sans entropie, ce qui suggère que l'ajout de cette régularisation ne contribue pas significativement à l'apprentissage dans le cas d'un espace d'actions discret. Les deux exécutions ont été faites dans les mêmes conditions (mêmes *seeds*, mêmes hyperparamètres, même environnement), la seule variation étant la désactivation de l'apprentissage de l'entropie.

Ce résultat peut s'expliquer de plusieurs façons :

- L'algorithme *DSAC* est peut-être trop instable pour bénéficier de l'entropie sans ajustement supplémentaire.
- Il se peut que la simple transposition de *SAC* en version discrète ne suffise pas pour exploiter efficacement l'entropie.
- Enfin, une erreur de notre part dans la prise en compte du terme d'entropie dans la fonction objectif pourrait invalider son effet dans notre implémentation.

5 Discussion

Nos expériences mettent en évidence plusieurs points clés sur le comportement de *DSAC* dans un environnement discret simple (*CartPole-v1*) :

- **Taux d'accord acteur–critique voisin de l'aléatoire.** Malgré des milliers d'itérations, l'action privilégiée par l'acteur coïncide seulement dans 50 % des cas avec celle de plus haute *Q-value* estimée par le critique. Avec seulement deux actions possibles, ce résultat équivaut à un choix aléatoire et suggère que l'acteur ne parvient pas à s'aligner sur les évaluations du critique. Plusieurs facteurs peuvent l'expliquer :
  - l'entropie induit une exploration persistante chez l'acteur,
  - le critique, sans mécanisme de stabilisation adapté, peut produire des cibles Q non stationnaires,
  - enfin, un décalage dans la mise à jour conjointe des réseaux peut entraîner une désynchronisation.
- **Performances inférieures à *DQN*.** Comparé à *DQN*, *DSAC* apprend beaucoup plus lentement et n'atteint jamais les récompenses maximales. Ce comportement est cohérent avec la littérature, qui signale la sensibilité de *DSAC out-of-the-box* aux hyperparamètres et à l'absence de correctifs spécifiques (comme le *double Q-learning*, le lissage des cibles, etc.).
- **Rôle négligeable de l'entropie ?** La suppression pure et simple de la régularisation par l'entropie n'a pas d'impact notable ni sur le taux d'accord, ni sur la courbe de récompense (voir Fig. 7).
- **Limites de l'étude.** Notre analyse se limite à un seul environnement discret et à une implémentation de référence de *DSAC*. L'écart avec des variantes améliorées (par exemple *ReDSAC*, *SD-SAC*) n'a pas été exploré. De plus, le nombre d'exécutions indépendantes ainsi que la diversité des *seeds* pourraient être augmentés afin de renforcer la validité statistique de nos résultats.

Ces résultats renforcent l'idée que *DSAC*, sans aménagements complémentaires, peine à reproduire la stabilité et l'efficacité observées dans les environnements à actions continues. Ils motivent la mise en œuvre future de techniques d'amélioration spécifiques.

6 Conclusion

Dans ce travail, nous avons implémenté et comparé six algorithmes d'apprentissage par renforcement profond au sein de la bibliothèque *BBRL*, en nous focalisant sur l'adaptation de *Soft Actor-Critic* aux espaces d'actions discrets, connue sous le nom de *DSAC*, ainsi que sur la mesure de la cohérence entre l'acteur et le critique.

Nos résultats mettent en évidence plusieurs points clés :

- Un taux d'accord avoisinant les 50 % entre l'acteur et le critique, suggérant une absence d'alignement plus fine que le hasard dans leurs décisions.
- Des performances d'apprentissage de *DSAC* significativement inférieures à celles de *DQN* sur l'environnement *CartPole-v1*.
- Une suppression de la régularisation par l'entropie (paramètre fixé à zéro) n'améliore ni la cohérence entre les réseaux, ni la stabilité de l'apprentissage.

Ces observations révèlent les limites de la version *vanilla* de *DSAC*, notamment dans les environnements discrets, et soulignent la nécessité d'envisager des extensions plus robustes pour favoriser la convergence et l'harmonisation entre l'acteur et le critique (telles que le *double Q-learning*, les pénalisations *KL*, ou encore un *Polyak averaging* plus conservatif).

Enfin, bien que nos résultats soient cohérents avec certaines observations de la littérature, nous ne pouvons exclure la possibilité que notre implémentation comporte des erreurs. Une vérification approfondie du code source et une comparaison avec d'autres implémentations open-source seraient nécessaires pour confirmer ces hypothèses et renforcer la validité de nos conclusions.

Remerciements

Nous souhaitons remercier notre encadrant, le Professeur Olivier Sigaud, pour son accompagnement tout au long de ce projet. Ses explications claires, ses remarques constructives et sa disponibilité nous ont permis de mieux comprendre les concepts du *reinforcement learning* et de prendre en main efficacement la bibliothèque *BBRL*. Son soutien a été précieux pour nous guider dans

l'implémentation des différents algorithmes, et pour structurer nos expérimentations et nos analyses.

References

[1] arXiv:1910.07207v2 [cs.LG] 18 Oct 2019

[2] arXiv:2112.02852v1 [cs.LG] 6 Dec 2021

[3] arXiv:2209.10081v4 [cs.LG] 20 Nov 2024