Cahier des charges
Apprentissage par renforcement avec AFU

Présentation du projet

Contexte. L'apprentissage par renforcement profond a connu des avancées significatives ces dernières années, notamment grâce à des algorithmes comme *SAC* (Soft Actor-Critic) et *DDPG* (Deep Deterministic Policy Gradient) qui ont établi l'état de l'art dans le domaine. Ces algorithmes reposent sur une architecture *acteur-critique*, où un réseau de neurones (l'acteur) apprend à sélectionner des actions tandis qu'un autre (le critique) évalue leur qualité.

Dans ce contexte, l'algorithme *AFU* (Actor-Free critic Updates), récemment développé par M. Perrin à l'ISIR, propose une approche innovante qui s'écarte de ce paradigme. AFU se distingue par sa capacité à mettre à jour le critique indépendamment de l'acteur, contrairement aux approches traditionnelles comme SAC et DDPG où les deux sont interdépendants. Cette caractéristique pourrait lui conférer des avantages significatifs en termes de stabilité d'apprentissage et de généralisation.

Ce projet s'inscrit dans le cadre du parcours AI2D du Master d'Informatique de Sorbonne Université. Sous la direction de M. Sigaud, membre de l'ISIR, nous chercherons à valider expérimentalement les propriétés théoriques d'AFU et à explorer ses avantages potentiels par rapport aux approches traditionnelles.

Objectif de recherche. Notre étude se concentre sur deux hypothèses principales qui, si elles sont validées, pourraient positionner AFU comme une alternative aux algorithmes actuels.

(Apprentissage à partir de données aléatoires)

La première hypothèse explore la capacité d'AFU à apprendre efficacement à partir de données générées aléatoirement, contrairement aux algorithmes traditionnels d'apprentissage par renforcement. Bien que ces derniers soient qualifiés d'*off-policy*, ils peinent à exploiter des données très éloignées de leur politique courante, ce qui limite leur efficacité d'apprentissage lorsque les données d'exploration sont diverses ou sous-optimales.

AFU pourrait surmonter cette limitation grâce à sa conception qui dissocie complètement l'apprentissage du critique de la politique d'exploration. Cette séparation architecturale permettrait à l'algorithme d'être véritablement *off-policy*, capable d'extraire de la valeur même à partir de données générées par des politiques radicalement différentes ou complètement aléatoires.

(Transition offline/online)

La seconde hypothèse concerne la stabilité d'AFU lors du passage de l'apprentissage *offline* (utilisant un ensemble de données préexistant) à l'apprentissage *online* (permettant l'interaction directe avec l'environnement). Les algorithmes actuels souffrent généralement d'une dégradation notable des performances durant cette transition, due au changement brutal dans la distribution des données et à la surestimation des valeurs d'actions peu représentées dans l'ensemble initial.

AFU, par sa nature véritablement *off-policy* et son mécanisme d'estimation des valeurs maximales, pourrait maintenir des performances stables durant cette phase critique de transition, offrant ainsi une solution élégante à ce problème récurrent dans le domaine de l'apprentissage par renforcement.

Impact attendu. La validation de ces hypothèses aurait un impact positif dans le domaine du Reinforcement Learning :

- une plus grande flexibilité dans la collecte de données d'apprentissage, permettant d'utiliser des données issues de sources diverses sans nécessiter une politique d'exploration spécifique ;
- une transition plus fluide entre les phases d'apprentissage offline et online, réduisant ainsi les coûts et les risques associés à cette transition ;
- une meilleure généralisation à partir de données limitées, un aspect crucial pour les applications réelles où la collecte de données peut être coûteuse ou risquée.

Ces avancées pourraient être particulièrement utiles pour des applications robotiques où la collecte de données est coûteuse et la stabilité de l'apprentissage cruciale pour assurer la sécurité et l'efficacité des systèmes.

Cadre technique

Environnement de développement. Notre projet s'appuie sur un ensemble d'outils et de bibliothèques spécialisés pour l'apprentissage par renforcement :

BBRL (BlackBoard Reinforcement Learning) est une bibliothèque conçue à des fins éducatives pour faciliter l'implémentation des algorithmes d'apprentissage par renforcement. Elle adopte une approche *Blackboard* où différents agents interagissent dans un espace de travail partagé. Cette architecture modulaire facilite la conception et l'implémentation d'algorithmes complexes tout en maintenant une structure claire et compréhensible.

Nous utiliserons Python avec PyTorch comme framework de deep learning. PyTorch est une bibliothèque open-source de machine learning largement utilisée en recherche. Elle offre une grande flexibilité et une facilité d'utilisation pour le développement d'algorithmes d'apprentissage profond, avec des fonctionnalités avancées comme la différentiation automatique et le support GPU.

Pour les environnements de test, nous emploierons Gymnasium, une suite standardisée d'environnements pour l'apprentissage par renforcement. Gymnasium offre une grande variété d'environnements pour tester les algorithmes d'apprentissage par renforcement et une API uniforme qui facilite l'évaluation comparative de multiples algorithmes.

Implémentation. Notre approche d'implémentation comprend plusieurs phases progressives :

Dans un premier temps, nous développerons plusieurs algorithmes classiques de l'apprentissage par renforcement profond comme DQN, DDPG et SAC, en utilisant BBRL et PyTorch. Cette étape nous permettra de maîtriser l'architecture *acteur-critique* traditionnelle et d'établir une base de comparaison solide pour évaluer les performances d'AFU.

Ensuite, nous implémenterons l'algorithme AFU en nous basant sur les principes décrits dans la publication originale. Pour comprendre la spécificité d'AFU, il est important de saisir la différence fondamentale entre les approches actuelles et celle proposée par AFU.

Dans les algorithmes traditionnels comme DDPG et SAC, bien qu'ils soient souvent qualifiés d'algorithmes *off-policy*, leur mise à jour du critique dépend fortement de l'acteur. Cette dépendance

limite leur capacité à apprendre efficacement à partir de données très différentes de celles générées par leur politique actuelle. On peut faire un parallèle avec la différence entre *SARSA* et *Q-learning* dans le cas tabulaire : *SARSA* dépend de l'action suivante pour apprendre tandis que *Q-learning* calcule directement la valeur maximale de ses *Q-valeurs* (l'estimation de la qualité de chaque action à un état donné).

AFU propose une approche novatrice pour résoudre ce qu'on appelle le problème « max-Q » dans les espaces d'actions continues. Au lieu de s'appuyer sur l'acteur pour estimer les meilleures actions, AFU emploie une technique basée sur la régression et la mise à l'échelle conditionnelle des gradients pour estimer directement la valeur maximale du critique pour chaque état. Cette approche permet théoriquement à AFU d'être véritablement *off-policy*, c'est-à-dire capable d'apprendre à partir de données générées par n'importe quelle politique, même aléatoire.

Notre première expérience visera à vérifier cette capacité. Nous mettrons en place un protocole où, au lieu d'utiliser des couples état-action générés par la politique de l'agent, nous fournirons des couples état-action échantillonnés uniformément dans l'espace état-action. Si AFU parvient à converger dans ces conditions, cela démontrera sa capacité à apprendre à partir de données véritablement *off-policy*.

La seconde expérience explorera la stabilité d'AFU lors de la transition entre apprentissage offline et online. L'*apprentissage offline* est devenu un domaine de recherche important ces dernières années, car il permet d'entraîner des agents sans interaction coûteuse ou risquée avec l'environnement réel. Cependant, les algorithmes actuels comme *CQL* ou *IQL* souffrent souvent d'une chute de performance lors du passage à l'apprentissage online.

Cette dégradation s'explique notamment par le fait que ces algorithmes, pour éviter la surestimation des valeurs des actions non présentes dans la base de données, ont tendance à sous-estimer systématiquement ces valeurs. Lors du passage à l'apprentissage online, ces valeurs sont soudainement mises à jour, créant un déséquilibre qui perturbe l'apprentissage. AFU, grâce à son mécanisme d'estimation des valeurs maximales, pourrait offrir une estimation plus précise même pour les actions non vues, facilitant ainsi la transition vers l'apprentissage online.

Pour tester cette hypothèse, nous entraînerons d'abord AFU sur un ensemble de données pré-collectées, puis nous passerons à un apprentissage online en mesurant précisément l'évolution des performances pendant cette transition. Si AFU maintient des performances stables, cela validera sa robustesse face à ce défi de l'apprentissage par renforcement.

Protocole Expérimental

Étude de l'apprentissage à partir de données aléatoires. Notre première hypothèse est qu'AFU, contrairement aux algorithmes traditionnels comme SAC et DDPG, est capable d'apprendre efficacement à partir de données générées de manière complètement aléatoire.

Pour vérifier cette hypothèse, nous mettrons en place le protocole suivant :

1. Modification des environnements de test pour permettre un échantillonnage uniforme des états et des actions, indépendamment de la dynamique naturelle de l'environnement.
2. Entraînement de trois algorithmes (AFU, SAC et DDPG) dans des conditions identiques où les transitions (état, action) sont générées aléatoirement plutôt que par l'interaction de la politique de l'agent avec l'environnement.

3. Évaluation régulière de la performance des politiques apprises en les déployant dans l'environnement standard (non aléatoire).

4. Analyse statistique des courbes d'apprentissage pour déterminer :
   – la vitesse de convergence de chaque algorithme ;
   – la performance finale atteinte ;
   – la stabilité de l'apprentissage (variance des résultats).

Pour garantir la validité statistique de nos résultats, chaque expérience sera répétée avec au moins 15 graines aléatoires différentes. Nous utiliserons le test de Welch pour déterminer si les différences de performance entre AFU et les autres algorithmes sont statistiquement significatives.

L'hypothèse sera considérée comme validée si AFU montre une convergence significativement meilleure et plus stable que SAC et DDPG dans ce scénario d'apprentissage à partir de données aléatoires.

Analyse de la transition offline/online. Notre seconde hypothèse est qu'AFU maintient des performances plus stables lors de la transition entre apprentissage offline et online, comparativement aux algorithmes spécialisés dans l'apprentissage offline comme IQL.

Pour vérifier cette hypothèse, nous suivrons ce protocole :

1. Phase d'apprentissage offline :
   – utilisation d'un ensemble de données pré-collectées par une politique sous-optimale ;
   – entraînement d'AFU et d'algorithmes de comparaison (IQL, CQL) sur ces données pendant un nombre fixe d'itérations ;
   – évaluation régulière des performances pour tracer les courbes d'apprentissage offline.

2. Phase de transition :
   – passage progressif à l'apprentissage online en permettant aux algorithmes d'interagir directement avec l'environnement ;
   – suivi précis des performances pendant cette phase critique ;
   – calcul du « transition gap » : la différence de performance avant et après la transition.

3. Phase d'apprentissage online :
   – poursuite de l'entraînement en mode entièrement online ;
   – évaluation de la capacité des algorithmes à récupérer et à améliorer leurs performances après la transition.

Pour l'analyse statistique, nous mesurerons :
– l'ampleur de la chute de performance lors de la transition (transition gap) ;
– le temps nécessaire pour retrouver les performances pré-transition ;
– la performance finale après une période d'apprentissage online.

L'hypothèse sera validée si AFU présente un transition gap significativement plus faible que les algorithmes de comparaison et/ou une récupération plus rapide après la transition.

Environnements de test. Nous utiliserons une gamme diversifiée d'environnements pour tester nos hypothèses, chacun présentant des défis spécifiques :

(Cartpole (version continue)) Un poteau attaché à un chariot mobile, utilisé comme banc d'essai initial pour nos algorithmes.

(Pendulum) Pendule inversé à maintenir à la verticale, utilisé pour tester les performances de base en contrôle continu.

(Lunar Lander (version continue)) simulateur d'alunissage avec espace d'action bidimensionnel, adapté pour tester l'apprentissage purement off-policy.

(Swimmer) Un environnement MuJoCo modélisant un nageur articulé devant se déplacer efficacement, offrant un défi de coordination pour comparer AFU aux approches traditionnelles.

(Ant Maze) Simule une fourmi robotique navigant dans différents types de labyrinthes (U-maze, etc.). Cet environnement est utilisé pour évaluer la transition entre apprentissage offline et online.

D'autres environnements MuJoCo pourront être intégrés selon les besoins du projet.

Méthodologie d'évaluation

Métriques de performances. Pour évaluer rigoureusement les performances des algorithmes, nous utiliserons plusieurs métriques complémentaires :

(Efficacité d'apprentissage) Nous calculerons l'aire sous la courbe d'apprentissage (AUC) et en extrairons trois statistiques clés : le premier quartile (Q1), le troisième quartile (Q3) et la moyenne interquartile (IQM).

(Performance finale) Pour évaluer la convergence, particulièrement importante pour l'hypothèse off-policy où DDPG et SAC pourraient ne pas converger, nous calculerons l'IQM des performances sur les 10% finaux de l'entraînement.

(Stabilité d'apprentissage) Nous mesurerons la variance normalisée des performances, permettant de comparer directement la stabilité des trois algorithmes dans un whisker plot.

(Transition gap) Pour la seconde hypothèse, nous calculerons l'AUC à partir du point de transition entre apprentissage offline et online, fournissant une métrique robuste de l'adaptation post-transition.

(Validité statistique) Nous effectuerons au minimum 15 exécutions avec différentes graines aléatoires, et emploierons le test de Welch pour déterminer la significativité des différences ($p < 0.05$).

Analyse comparative. Notre analyse comparative se structurera autour de trois axes principaux :

Comparaison algorithmique : nous comparerons systématiquement AFU avec :
– SAC et DDPG comme représentants des algorithmes acteur-critique traditionnels
– IQL comme représentant des algorithmes spécialisés dans l'apprentissage offline

Analyse et visualisation : nous produirons des whisker plots et courbes d'apprentissage permettant de comparer efficacement les performances et la stabilité des différents algorithmes, avec une attention particulière à la transition offline/online.

Cette approche nous permettra de déterminer si AFU présente un avantage significatif dans les scénarios testés.

Organisation du projet

Planning. Notre projet se déroulera en trois phases principales, chacune avec des objectifs spécifiques :

(Phase 1 – Février) Mise en place et implémentation :
– configuration de l'environnement de développement et création du dépôt GitHub ;

– implémentation des algorithmes de référence (DQN, DDPG, SAC) ;
– développement de deux versions d'AFU : une implémentation BBRL et une adaptation de l'implémentation de M. Perrin ;
– mise en place des premiers environnements de test (Cartpole, Pendulum).

(Phase 2 – Mars) Expérimentation et collecte de données :
– extension aux environnements Lunar Lander, Swimmer et Ant Maze ;
– réalisation des expériences sur l'apprentissage à partir de données aléatoires ;
– collecte des données de performance pour tous les algorithmes ;
– optimisation des hyperparamètres d'AFU.

(Phase 3 – Avril-Mai) Analyse et finalisation :
– analyse statistique des résultats expérimentaux ;
– comparaison systématique des performances ;
– rédaction du rapport final ;
– préparation de la soutenance.

Livrables. À l'issue du projet, nous fournirons les éléments suivants :

– Code source documenté : implémentations d'AFU et des algorithmes de comparaison, scripts d'expérimentation et documentation.
– Résultats expérimentaux : données brutes, visualisations et analyses statistiques permettant de reproduire les expériences.
– Rapport final : document présentant le contexte, la méthodologie, les résultats, et les conclusions du projet.
– Présentation de soutenance : support visuel synthétisant les points clés et résultats significatifs.

Critères de réussite

Le projet sera considéré comme réussi s'il :

1. Démontre clairement si AFU possède ou non la capacité à apprendre efficacement à partir de données générées aléatoirement, avec des résultats statistiquement significatifs permettant de trancher cette question.
2. Quantifie précisément la stabilité d'AFU pendant la transition offline/online, en comparaison avec d'autres algorithmes spécialisés, fournissant ainsi une évaluation objective de cet aspect.
3. Produit des résultats reproductibles, documentés de manière exhaustive et soutenus par des analyses statistiques rigoureuses, garantissant ainsi la fiabilité scientifique des conclusions.
4. Livre une implémentation robuste et bien documentée d'AFU dans BBRL, contribuant ainsi aux outils disponibles pour la communauté de recherche en apprentissage par renforcement.

Ces critères garantissent que le projet, quelle que soit la validation ou l'invalidation des hypothèses initiales, produira des connaissances scientifiques valides et utiles sur les propriétés d'AFU et son positionnement par rapport aux algorithmes existants.