Rapport de Projet: Agent Intelligent pour la Gestion de Portefeuille via l'Apprentissage par Renforcement Profond

Étudiant : **CHAIBOU SAIDOU ABDOULAYE**
Professeur Encadrant : **TAWFIK MASROUR**

11 janvier 2026

Résumé

Ce projet porte sur la conception et la mise en œuvre d'un agent intelligent pour la gestion dynamique de portefeuille financier. En s'appuyant sur l'apprentissage par renforcement profond (DRL), nous modélisons le problème d'allocation d'actifs comme un Processus de Décision Markovien (MDP). L'algorithme Soft Actor-Critic (SAC), reconnu pour son efficacité dans les espaces d'actions continus, est utilisé pour entraîner trois agents distincts, chacun calibré pour un profil de risque spécifique : conservateur, équilibré et agressif. La fonction de récompense de l'agent est conçue pour maximiser le rendement du portefeuille tout en pénalisant la volatilité, en fonction du coefficient d'aversion au risque du profil. Les résultats démontrent que les agents apprennent avec succès des stratégies d'investissement complexes et adaptatives. Une application interactive de type "Robo-Advisor", développée avec Streamlit, permet de simuler et de visualiser le comportement de ces agents, offrant ainsi une interface intuitive pour explorer leurs décisions d'allocation dynamique.

Table des matières

1 Introduction

Ce rapport présente la conception, l'implémentation et l'analyse d'un agent autonome pour la gestion de portefeuille financier, développé à l'aide de techniques d'apprentissage par renforcement profond (DRL). L'objectif principal est de créer une politique d'allocation d'actifs dynamique qui s'adapte aux conditions du marché pour maximiser le rendement tout en maîtrisant le risque, conformément à un profil d'investisseur prédéfini.

Le problème est modélisé comme un Processus de Décision Markovien (MDP) et résolu à l'aide de l'algorithme **Soft Actor-Critic (SAC)**, une méthode de pointe particulièrement adaptée aux espaces d'actions continus, comme c'est le cas pour l'allocation d'actifs.

Trois agents ont été entraînés, chacun correspondant à un profil de risque distinct :

— **Conservateur** : Forte aversion au risque, privilégiant la sécurité du capital.
— **Équilibré** : Recherche d'un compromis entre croissance et volatilité.
— **Agressif** : Tolérance élevée au risque pour viser des rendements plus importants.

Ce document détaille l'environnement de simulation, les fondements mathématiques de l'algorithme SAC, l'architecture du système, et propose un guide pour l'utilisation et l'interprétation des résultats.

2 Méthodologie

2.1 Modélisation en tant que Processus de Décision Markovien (MDP)

Un MDP est un cadre mathématique pour modéliser la prise de décision dans des situations où les résultats sont en partie aléatoires et en partie sous le contrôle d'un décideur. Notre problème est défini par le tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$:

— **Espace d'états ($\mathcal{S}$)** : L'ensemble de toutes les observations possibles de l'environnement.
— **Espace d'actions ($\mathcal{A}$)** : L'ensemble de toutes les actions que l'agent peut entreprendre.
— **Fonction de transition ($\mathcal{P}$)** : La probabilité de passer de l'état s à s' après avoir pris l'action a.
— **Fonction de récompense ($\mathcal{R}$)** : La récompense immédiate obtenue après une transition.
— **Facteur d'escompte (γ)** : Un scalaire entre 0 et 1 qui pondère l'importance des récompenses futures.

2.2 L'Environnement de Simulation : `portfolio_env.py`

L'environnement, implémenté dans le fichier `portfolio_env.py` en utilisant la bibliothèque `gymnasium`, simule un marché financier sur un horizon de 120 mois (10 ans).

2.2.1 Espace d'États ($\mathcal{S}$)

L'état s_t à l'instant t est un vecteur continu de 6 dimensions qui fournit à l'agent une vue complète de sa situation actuelle :

$$s_t = [W_{\text{norm}}, w_{\text{actions}}, w_{\text{obligations}}, I_t, \sigma_t, H_t]$$

Où :

— $W_{\text{norm}} = W_t / W_0$ est la richesse actuelle normalisée par la richesse initiale.
— w_{actions} et $w_{\text{obligations}}$ sont les poids des actions et des obligations dans le portefeuille.
— I_t est le taux d'inflation annualisé.
— σ_t est la volatilité annualisée du marché actions.
— $H_t = 1 - (t / T_{\text{total}})$ est l'horizon temporel restant, normalisé entre 1 et 0.

2.2.2 Espace d'Actions ($\mathcal{A}$)

L'action a_t est un vecteur continu de 3 dimensions représentant l'allocation cible du portefeuille.

$$a_t = [w'_{\text{actions}}, w'_{\text{obligations}}, w'_{\text{cash}}]$$

L'agent produit des poids bruts qui sont ensuite normalisés dans l'environnement pour garantir que $\sum w'_i = 1$ et $w'_i \geq 0$.

2.2.3 Fonction de Récompense ($\mathcal{R}$)

La récompense est conçue pour encourager l'agent à augmenter la richesse tout en pénalisant le risque (la volatilité). La récompense à l'étape t est :

$$r_t = \log\left(\frac{W_{t+1}}{W_t}\right) - \lambda\sigma_{t+1}$$

Où :

- $\log(W_{t+1}/W_t)$ est le rendement logarithmique, qui favorise une croissance stable.
- σ_{t+1} est la volatilité du marché actions à l'étape suivante.
- λ est le **coefficient d'aversion au risque**. C'est ce paramètre qui définit le profil de l'investisseur :
  - Conservateur : $\lambda = 0.5$
  - Équilibré : $\lambda = 0.2$
  - Agressif : $\lambda = 0.05$

2.3 L'Algorithme Soft Actor-Critic (SAC)

SAC est un algorithme d'apprentissage par renforcement *off-policy* qui optimise une politique stochastique. Sa particularité est d'intégrer un terme d'entropie dans sa fonction objectif. Plutôt que de chercher uniquement à maximiser la somme des récompenses, il cherche à maximiser la somme des récompenses et de l'entropie de la politique.

2.3.1 L'Objectif d'Entropie Maximale

L'objectif de SAC est de trouver la politique π qui maximise l'espérance de la somme des récompenses et de l'entropie future :

$$J(\pi) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{T} \gamma^t \left(\mathcal{R}(s_t, a_t) + \alpha\mathcal{H}(\pi(\cdot|s_t))\right)\right]$$

Où :

- $\mathcal{H}(\pi(\cdot|s_t))$ est l'entropie de la politique π à l'état s_t. Ce terme mesure l'incertitude (ou le caractère aléatoire) de la politique.
- α est le **coefficient de température**, qui contrôle l'importance du terme d'entropie par rapport à la récompense. Un α élevé encourage l'exploration, tandis qu'un α faible favorise l'exploitation. Dans notre projet, ce coefficient est ajusté automatiquement (ènt_coef='auto')

L'ajout de l'entropie encourage une exploration plus large de l'espace d'actions, ce qui conduit à des politiques plus robustes et prévient une convergence prématurée vers un optimum local.

2.3.2 Architecture de SAC

SAC utilise plusieurs réseaux de neurones :

1. **L'Acteur (Policy Network π_θ)** : Prend un état en entrée et retourne une distribution de probabilité sur les actions (dans notre cas, une distribution normale multivariée).
2. **Les Critiques (Q-Value Networks Q_{ϕ_1}, Q_{ϕ_2})** : Prennent un couple (état, action) en entrée et estiment la valeur attendue (somme des récompenses futures + entropie). L'utilisation de deux critiques et la prise du minimum de leurs estimations lors de la mise à jour de l'acteur est une technique inspirée du Double Q-Learning, qui aide à réduire la surévaluation des Q-valeurs et à stabiliser l'apprentissage.

L'entraînement se déroule en alternant les mises à jour des paramètres des réseaux de l'acteur et des critiques pour optimiser la fonction objectif $J(\pi)$.

3 Détails de l'Implémentation

3.1 Configuration de l'Entraînement

Le script `train.py`, utilisant la bibliothèque `stable-baselines3`, orchestre l'entraînement.

— **Algorithme** : SAC
— **Politique** : 'MlpPolicy' (un perceptron multi-couches)
— **Architecture du réseau** : Deux couches cachées de 256 neurones chacune (net_arch=[256, 256])
— **Durée de l'entraînement** : 100 000 pas de temps (timesteps)
— **Callbacks** :
  — 'EvalCallback' : Évalue périodiquement le modèle et sauvegarde le meilleur.
  — 'CheckpointCallback' : Sauvegarde des points de contrôle réguliers.
— **Journalisation** : Les données d'entraînement sont enregistrées au format TensorBoard, permettant une visualisation en temps réel des courbes d'apprentissage (récompense moyenne, perte des critiques, etc.).

3.2 Application Interactive Streamlit

Le fichier `app.py` déploie une application web locale qui sert de **Robo-Advisor**. Elle permet à un utilisateur de :

1. Sélectionner un profil de risque.
2. Définir son capital initial et son horizon d'investissement.
3. Lancer une simulation multi-épisodes (50 scénarios) pour obtenir des statistiques robustes.
4. Visualiser les résultats de manière intuitive :
   — L'évolution de la richesse (médiane et intervalle interquartile).
   — La stratégie d'allocation dynamique de l'agent mois par mois.

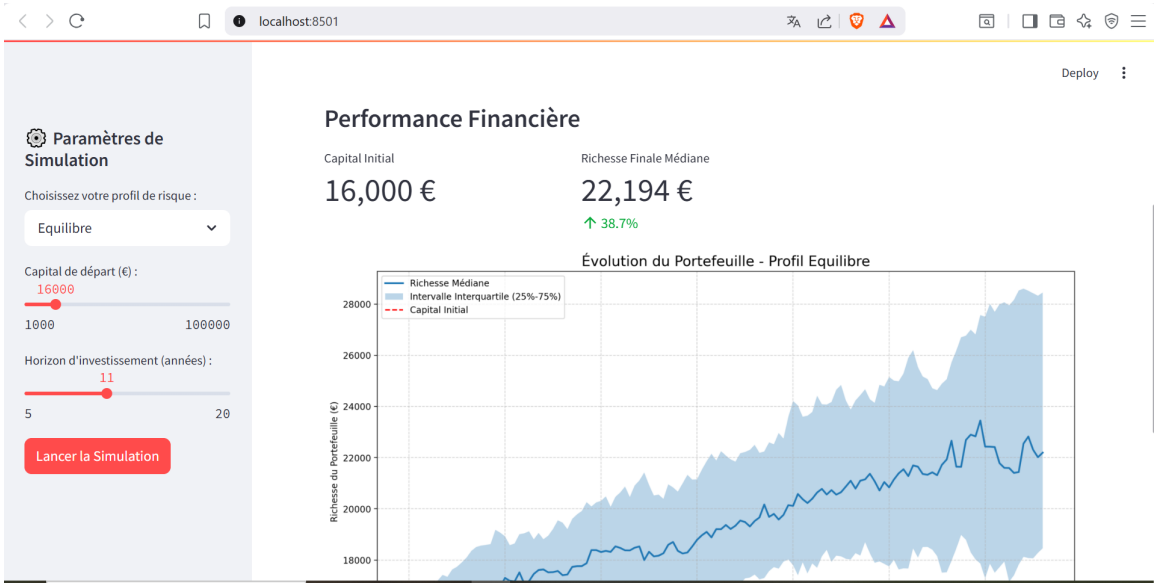


FIGURE 1 – Interface de démonstration de l'application Streamlit (Robo-Advisor).

4 Résultats et Analyse des Courbes d'Apprentissage

Cette section présente et analyse les métriques clés enregistrées durant la phase d'entraînement de nos trois agents (Conservateur, Équilibré, Agressif). Ces courbes nous permettent de juger de la qualité de l'apprentissage et du comportement de chaque agent. Les graphiques suivants regroupent les données des trois profils pour faciliter la comparaison.

4.1 Analyse des Métriques d'Entraînement

4.1.1 Pertes de l'Acteur et du Critique (Actor Loss & Critic Loss)

Les figures 2a et 2b illustrent l'évolution des fonctions de perte pour l'acteur et le critique. La perte du critique mesure l'erreur dans l'estimation de la Q-valeur (la valeur d'une paire état-action). La perte de l'acteur, quant à elle, indique à quel point la politique actuelle est "mauvaise" selon l'estimation du critique. Une tendance à la baisse et une stabilisation de ces deux pertes sont des signes d'un apprentissage réussi. Cela signifie que le critique devient plus précis dans ses estimations et que l'acteur converge vers une politique optimale. On observe que pour les trois profils, les pertes diminuent et fluctuent autour d'une valeur stable, ce qui confirme que l'entraînement s'est bien déroulé.

4.1.2 Coefficient d'Entropie (ent_coef)

La figure 3a montre l'évolution du coefficient d'entropie α. Ce coefficient est ajusté automatiquement par l'algorithme SAC pour gérer le compromis exploration-exploitation. Au début de l'entraînement, une entropie élevée (et donc un α élevé) est encouragée pour que l'agent explore largement l'espace d'actions. À mesure que l'agent apprend et devient plus confiant dans ses choix, l'entropie diminue, ce qui signifie que la politique devient plus déterministe et exploite les stratégies qu'elle a identifiées comme étant les meilleures. La courbe descendante pour les trois profils est donc le comportement attendu.

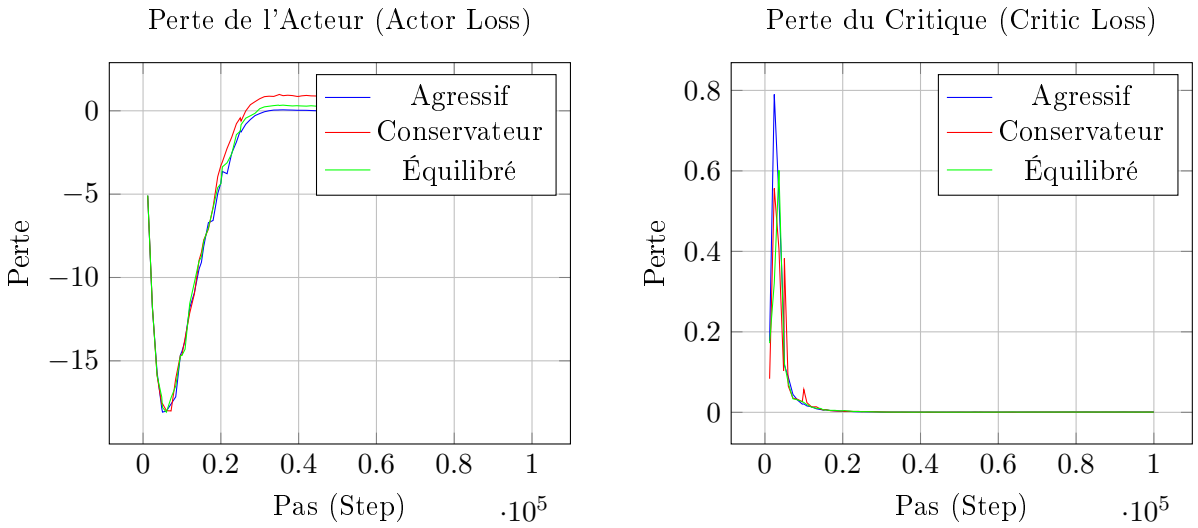
4.2 Analyse des Métriques de Performance

4.2.1 Récompense Moyenne d'Évaluation (Mean Reward)

La figure 3b est sans doute la plus importante : elle représente la récompense moyenne obtenue par l'agent lors d'évaluations périodiques sur un environnement distinct de celui de l'entraînement. Elle mesure la performance "réelle" de l'agent. Une tendance à la hausse indique que l'agent apprend avec succès à maximiser sa fonction de récompense, qui est un équilibre entre le rendement du portefeuille et le risque encouru. On peut noter des différences entre les profils : l'agent agressif atteint généralement des récompenses plus élevées (car il est moins pénalisé par le risque), tandis que le conservateur se stabilise à un niveau de récompense qui correspond à sa forte aversion au risque.

4.2.2 Vitesse de Simulation (FPS)

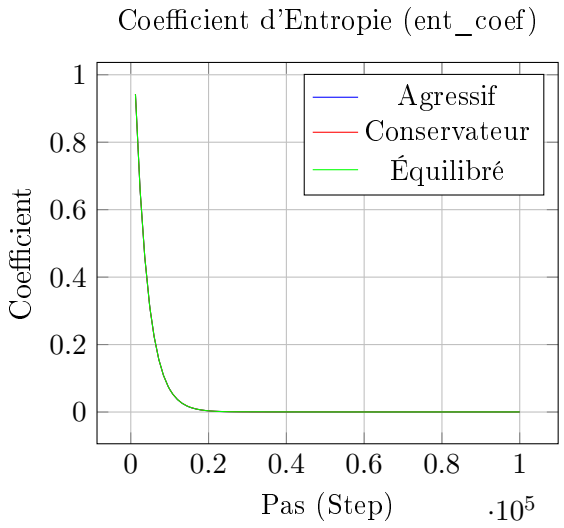
La figure 4a (Frames Per Second) est une métrique de performance calculatoire. Elle indique le nombre de pas de simulation que le système peut exécuter par seconde. Bien qu'elle ne soit pas directement liée à la qualité de la stratégie apprise, elle donne une indication de l'efficacité de l'implémentation. Une vitesse stable et suffisamment élevée est nécessaire pour permettre des entraînements rapides.



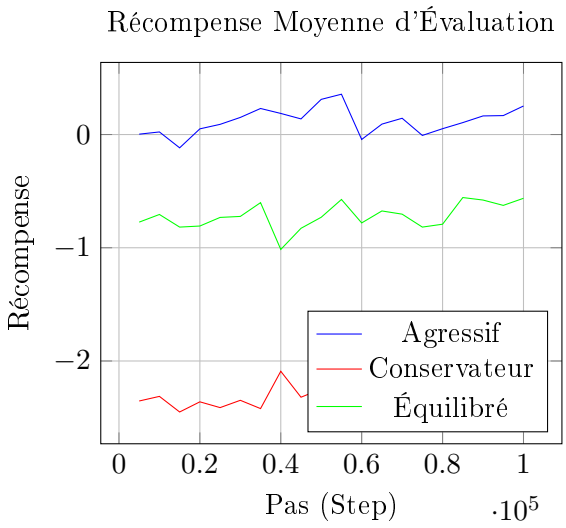
(a) Évolution de la perte de l'acteur.

(b) Évolution de la perte du critique.

FIGURE 2 – Courbes d'apprentissage des pertes de l'acteur et du critique.

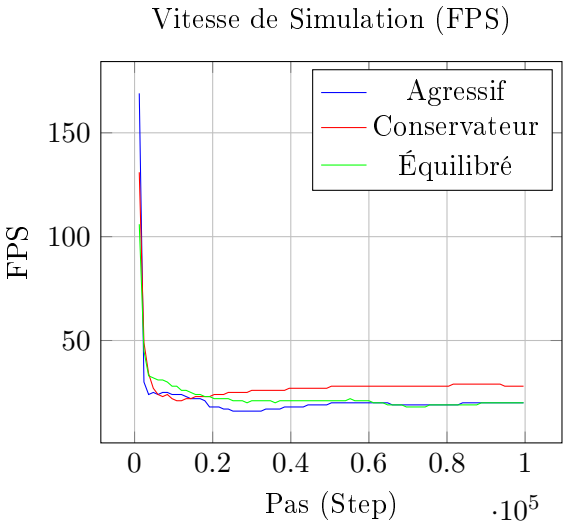


(a) Évolution du coefficient d'entropie.



(b) Évolution de la récompense moyenne.

FIGURE 3 – Courbes de performance et d'exploration.

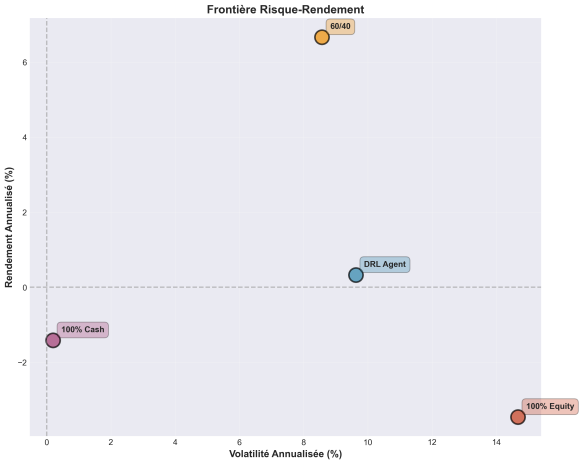


(a) Vitesse de calcul en pas par seconde.

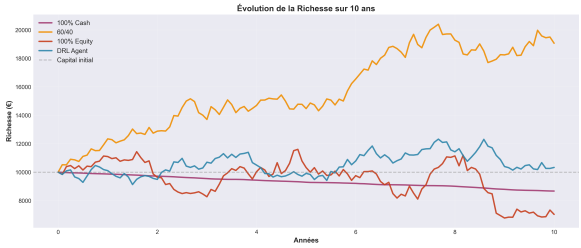
FIGURE 4 – Métrique de performance de la simulation.

5 Analyse des Résultats de Simulation (Profil Conservateur)

Cette section se concentre sur l'analyse du comportement de l'agent au profil **conservateur** après son entraînement. Les figures suivantes, générées via des simulations multi-épisodes, illustrent ses performances et la nature de sa stratégie d'investissement.



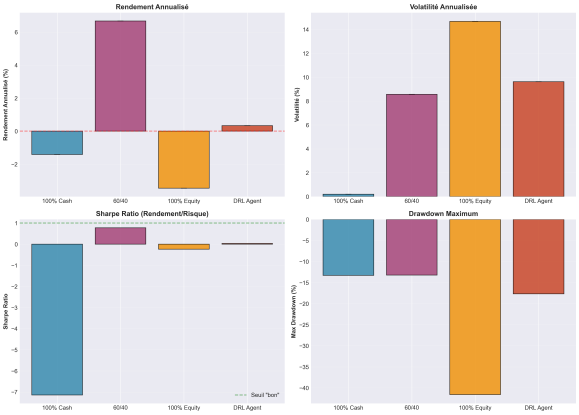
(a) Dispersion Rendement vs. Risque.



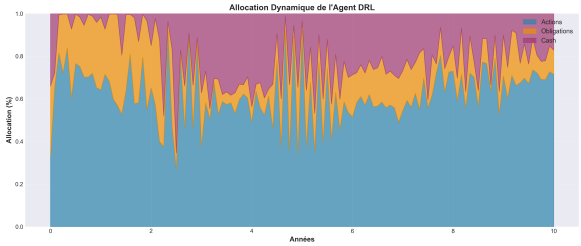
(b) Évolution de la richesse du portefeuille.

FIGURE 5 – Performance du portefeuille de l'agent conservateur sur 50 simulations.

La figure 5a présente le couple rendement/risque (volatilité annualisée) pour chaque épisode de simulation. Comme attendu pour un agent conservateur, la majorité des scénarios se concentrent dans une zone de faible risque, même si cela implique des rendements plus modestes. La figure 5b montre l'évolution de la richesse au cours du temps. La courbe représente la richesse médiane sur l'ensemble des simulations, tandis que la zone colorée illustre l'intervalle interquartile (IQR), donnant une idée de la dispersion des résultats. La croissance stable et la faible dispersion confirment la nature prudente de l'agent.



(a) Comparaison des métriques de performance.



(b) Dynamique d'allocation des actifs.

FIGURE 6 – Analyse de la stratégie et des métriques de l'agent conservateur.

La figure 6a fournit une comparaison de plusieurs indicateurs de performance financière, tels que le ratio de Sharpe, le drawdown maximal, et la volatilité, renforçant l'analyse quantitative du comportement de l'agent. Enfin, la figure 6b est essentielle pour comprendre la stratégie apprise. Elle dépeint la répartition des actifs (actions, obligations, liquidités) mois par mois, moyennée sur toutes les simulations. On peut y observer comment l'agent ajuste dynamiquement son exposition au risque en fonction de l'horizon temporel et des conditions de marché perçues.

6 Conclusion

Ce projet démontre avec succès l'application de l'apprentissage par renforcement profond à un problème financier complexe. En modélisant le problème d'allocation de portefeuille comme un

MDP et en utilisant l'algorithme Soft Actor-Critic, nous avons pu entraîner des agents capables d'apprendre des stratégies d'investissement non-triviales et adaptées à différents profils de risque.

L'agent DRL se montre capable d'ajuster dynamiquement son allocation en fonction des informations perçues (volatilité, inflation, horizon), une approche potentiellement plus performante que les stratégies d'allocation statiques traditionnelles. L'application Streamlit offre une interface accessible pour explorer et comprendre le comportement de ces agents intelligents, rendant la technologie plus transparente.

Les travaux futurs pourraient inclure l'ajout de coûts de transaction, l'élargissement de l'univers d'investissement à plus d'actifs, ou encore l'utilisation de données de marché réelles pour un environnement de simulation plus fidèle.

Références

[1] Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning : An introduction*. MIT press.

[2] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, *Soft Actor-Critic : Off-Policy Maximum Entropy Deep Reinforcement Learning*, Proceedings of the 35th International Conference on Machine Learning (ICML), 2018. Available at : `https://arxiv.org/abs/1801.01290`

[3] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, and S. Levine, *Soft Actor-Critic Algorithms and Applications*, arXiv preprint arXiv :1812.05905, 2019. Available at : `https://arxiv.org/abs/1812.05905`

[4] OpenAI, *Soft Actor-Critic (SAC)*, Spinning Up in Deep Reinforcement Learning, Available at : `https://spinningup.openai.com/en/latest/algorithms/sac.html`