Examen de Machine Learning 23-24

1. **Complétion des parties manquantes** :

python

```
# Annexe 1
!pip install stable-baselines3

import gym  # (1)
from stable_baselines3 import PPO  # (2)
from stable_baselines3.common.evaluation import evaluate_policy  # (3)

%load_ext tensorboard  # (4)

env = gym.make("CartPole-v1")  # (5)
model = PPO("MlpPolicy", env, verbose=1,
tensorboard_log="./cartpole_tensorboard/")  # (6)
model.learn(total_timesteps=100000)

# Évaluer le modèle
evaluate_policy(model, env, n_eval_episodes=10, render=False)  # (7),
(8), (9), (10)

# Sauvegarder le modèle en l'appelant PPO_model
model.save("PPO_model")  # (11)

# Supprimer le modèle
del model  # (12)

# Charger le modèle sauvegardé
model = PPO.load("PPO_model", env=env)  # (13), (14)

# Utiliser le modèle
obs = env.reset()  # (15)
while True:
    action, _states = model.predict(obs)  # (16), (17)
    obs, rewards, dones, info = env.step(action)  # (18), (19), (20)
    env.render()
```

2. **Faut-il entraîner davantage le modèle ou l'arrêter ?**
   En se basant sur les métriques fournies :
   - `ep_len_mean = 500` et `ep_rew_mean = 500` : Cela signifie que le modèle atteint la performance maximale possible dans l'environnement CartPole-v1, où un épisode est limité à 500 étapes.
   - Les autres métriques (`loss`, `policy_gradient_loss`, `value_loss`) montrent des valeurs faibles et stables.

   Conclusion : Il n'est pas nécessaire d'entraîner davantage le modèle car il a déjà atteint les performances maximales possibles dans cet environnement. Continuer l'entraînement pourrait entraîner un surajustement et ne serait pas bénéfique.

Exercice 2 : Analyse des algorithmes REINFORCE et PPO

1. **Quel est le principal inconvénient du premier algorithme (REINFORCE) ?**
   Le principal inconvénient de l'algorithme REINFORCE réside dans sa **variance élevée** et son inefficacité en termes d'échantillons :
   - **Variance élevée** : Les gradients de politique estimés reposent sur des retours cumulés $Q^{\pi_\theta}(s_t, a_t)$, qui varient fortement, rendant l'entraînement instable.
   - **Aucune utilisation de données hors-politiques** : Chaque mise à jour de la politique dépend strictement des trajectoires générées par la politique actuelle. Cela entraîne un faible rendement de l'échantillonnage et une convergence lente.
   - **Instabilité** : Les grandes mises à jour des paramètres peuvent parfois dégrader la politique au lieu de l'améliorer.

2. **Indiquer les principales différences du second algorithme (PPO) par rapport au premier.**
   Les principales différences entre PPO et REINFORCE sont :
   - **Utilisation d'une fonction de clipping (PPO)** : L'objectif $L_k^{CLIP}(\theta)$ empêche les mises à jour trop importantes des probabilités de la politique, limitant ainsi l'instabilité.
   - **Utilisation du replay buffer** : PPO exploite un buffer pour réutiliser des trajectoires précédentes, améliorant ainsi l'efficacité des échantillons.
   - **Estimation de l'avantage** $\hat{A}_t^{\pi_k}$: Au lieu d'utiliser directement les retours cumulés comme dans REINFORCE, PPO emploie une estimation plus précise de l'avantage, ce qui réduit la variance des gradients.
   - **Mises à jour multiples par trajectoire** : PPO effectue plusieurs mises à jour de la politique en utilisant les mêmes données, maximisant leur utilité.

- **Gradient de politique trust-region :** PPO combine la simplicité de REINFORCE avec des contraintes sur les mises à jour (grâce au clipping), réduisant ainsi les risques d'effondrement de la politique.

Résumé :

- REINFORCE est simple mais souffre d'instabilité et d'une inefficacité en termes d'échantillons.
- PPO introduit des améliorations comme le clipping, le replay buffer, et l'estimation de l'avantage pour un entraînement plus stable et efficace.

Exercice 3 : Analyse des modèles d'architecture de réseaux convolutifs

1. **À quoi correspond l'architecture du modèle (a) ?**

   Le modèle (a) correspond à une architecture inspirée de l'**Inception module** des réseaux convolutifs. Cette architecture se distingue par :

   - L'utilisation de plusieurs **branches parallèles** avec différentes tailles de convolution (1x1, 3x3, 5x5).
   - Une branche dédiée au **max pooling** (3x3 MaxPool).
   - Une **concaténation** finale des sorties des branches pour regrouper les informations extraites à différentes échelles spatiales et caractéristiques.

   L'objectif principal de ce type d'architecture est de capturer des **caractéristiques multi-échelles** tout en optimisant l'efficacité du calcul grâce à des convolutions 1x1 pour réduire les dimensions.

2. **Calculer le nombre de paramètres de chaque modèle :**

   Pour calculer le nombre de paramètres, la formule générale est :

   $$\text{Paramètres} = (\text{Taille du filtre} \times \text{Taille du filtre} \times \text{Canaux d'entrée} + 1) \times \text{Nombre de filtres}$$

   où $+1$ représente le biais associé à chaque filtre.

   - **Modèle (a) :**
     - **1x1 Conv (64) :**

       $$(1 \times 1 \times 192 + 1) \times 64 = 12352$$

     - **1x1 Conv (96) :**

       $$(1 \times 1 \times 192 + 1) \times 96 = 18528$$

- **3x3 Conv (128) :**

$$(3 \times 3 \times 96 + 1) \times 128 = 110720$$

- **5x5 Conv (32) :**

$$(5 \times 5 \times 192 + 1) \times 32 = 153632$$

- **1x1 Conv (16) :**

$$(1 \times 1 \times 192 + 1) \times 16 = 3088$$

- **1x1 Conv (32) :**

$$(1 \times 1 \times 192 + 1) \times 32 = 6176$$

Total : $12352 + 18528 + 110720 + 153632 + 3088 + 6176 = 304496$

- **Modèle (b) :**
  - **3x3 Conv (256) :**

$$(3 \times 3 \times 192 + 1) \times 256 = 442624$$

  Total : 442624
- **Modèle (c) :**
  - **5x5 Conv (256) :**

$$(5 \times 5 \times 192 + 1) \times 256 = 1224704$$

  Total : 1224704

3. **Commentaires sur les résultats :**
    - **Modèle (a) :**
        - Complexité modérée avec un total de 304496 paramètres.
        - Permet une meilleure extraction multi-échelle des caractéristiques grâce aux convolutions de tailles différentes.
    - **Modèle (b) :**
        - Nombre de paramètres relativement élevé (442624) pour une seule branche.
        - Moins flexible car il n'explore qu'une seule taille de convolution (3x3).

- **Modèle (c) :**
    - Le plus coûteux avec $1,22 \times 10^6$ paramètres.
    - Se concentre uniquement sur de grandes convolutions (5x5), ce qui peut être inefficace pour des images avec peu de détails.

Conclusion : Le modèle (a) est le meilleur compromis entre complexité et capacité à extraire des caractéristiques multi-échelles.