Bibliothèque d'Algorithmes de Renforcement

Projet académique de Machine Learning

Ton Nom

April 26, 2025

Abstract

Ce projet propose une bibliothèque Java permettant d'appliquer différents algorithmes de Reinforcement Learning (RL) sur des environnements déterministes simples tels que SimpleRL, YouTube, et TicTacToe. Ce rapport présente l'architecture développée, les méthodes employées, les résultats obtenus, et les pistes d'amélioration futures.

Contents

1 Introduction

Le but de ce projet est de concevoir une architecture modulaire permettant de tester et comparer plusieurs algorithmes de Reinforcement Learning sur différents environnements classiques. L'objectif est d'assurer la généralité du couplage Agent Environnement, tout en permettant la personnalisation via les fonctions de récompense.

2 Objectifs du projet

- Créer une bibliothèque Java modulaire pour tester des agents RL.
- Séparer proprement l'agent, l'environnement et la fonction de récompense.
- Implémenter plusieurs agents : Epsilon-Greedy, UCB, Bandit Gradient, Value Iteration, Policy Iteration, Q-Learning.
- Tester les agents sur trois environnements différents : SimpleRL, YouTube, TicTacToe.
- Générer automatiquement des analyses et des graphes de résultats.

3 Méthodologie

3.1 Architecture du projet

Le projet suit une architecture en couches :

- **Agent** : Prend les décisions selon l'environnement et la politique.
- **Environnement** : Fournit les états et transitions accessibles.
- **RewardFunction** : Définit les récompenses spécifiques à chaque problème.
- **Runner** : Gère le chargement et l'initialisation des B Machines.
- **Analysis** : Génère des graphes et des statistiques post-entraînement.

3.2 Environnements utilisés

- **SimpleRL** : Sélection de films avec états OK/KO.
- **YouTube** : Modélisation de séquences de vidéos.
- **TicTacToe** : Jeu complet du Morpion avec alternance de joueurs.

3.3 Algorithmes implémentés

- **Bandits** : Epsilon-Greedy, UCB, Bandit Gradient.
- **MDPs** : Value Iteration, Policy Iteration, Q-Learning.

4 Résultats obtenus

4.1 Analyse simple des récompenses

Après plusieurs entraînements, les résultats montrent :

- Sur SimpleRL : convergence rapide vers l'action optimale.
- Sur YouTube : bonne estimation des durées maximisant la récompense.
- Sur TicTacToe : apprentissage progressif de stratégies menant à la victoire.

4.2 Génération de graphes

Des graphes automatiques sont générés pour :

- Récompenses cumulées
- Récompenses moyennes
- Distribution des actions

4.3 Observations spécifiques TicTacToe

- Le Q-Learning améliore progressivement le taux de victoire du joueur 0.
- Value Iteration permet de calculer une politique optimale théorique.

5 Difficultés rencontrées

- Gérer le reset propre de l'environnement sans perte du StateSpace.
- Adapter la logique de training pour passer d'un nombre de pas à un nombre d'épisodes (notamment pour TicTacToe).
- Stabiliser les algorithmes de policy/value iteration dans des environnements de grande taille.

6 Utilisation du projet avec Maven

Le projet utilise **Maven** pour la compilation et l'exécution.
Voici les commandes à suivre pour utiliser l'application :

- Nettoyer le projet (supprimer les anciens fichiers compilés) :

```
mvn clean
```

- Compiler le projet :

```
mvn compile
```

- Lancer l'application principale (classe `fr.polytech.mnia.App`) :

  ```
  mvn exec:java
  ```

Remarque : Assurez-vous que votre fichier `pom.xml` contient bien la configuration suivante pour définir la classe principale à exécuter :

```
<build>
    <plugins>
        <plugin>
            <groupId>org.codehaus.mojo</groupId>
            <artifactId>exec-maven-plugin</artifactId>
            <version>3.0.0</version>
            <configuration>
                <mainClass>fr.polytech.mnia.App</mainClass>
            </configuration>
        </plugin>
    </plugins>
</build>
```

Après lancement, l'application s'exécutera dans la console et proposera :

- Le choix de l'environnement (SimpleRL, YouTube, TicTacToe)
- Le choix des agents
- La configuration des hyperparamètres
- L'entraînement et l'analyse automatiques