Dal Pian Nathan - p2210597
Aboufath Ibtissam - p2110758
5A INFO

POLYTECH LYON

Alpha Zero - Jeu d'échecs

2024/2025

Dal Pian Nathan - p2210597
Aboufath Ibtissam - p2110758
5A INFO

Introduction

Le but de ce projet est d'implémenter un système d'IA basé sur l'architecture d'AlphaZero spécifiquement pour les échecs. Nous souhaitons reproduire le même processus d'apprentissage autonome via self-play, en utilisant des techniques d'apprentissage par renforcement et une recherche basée sur Monte Carlo Tree Search (MCTS), le tout soutenu par un réseau neuronal profond pour guider la prise de décision de l'IA.

Objectifs du rapport
Ce rapport a pour objectif de décrire l'implémentation d'AlphaZero pour les échecs en expliquant :

- **Les techniques de recherche** utilisées pour explorer les coups possibles : Monte Carlo Tree Search (**MCTS**).
- Le rôle du **réseau neuronal profond** pour évaluer les positions et prédire les coups à jouer.
- **L'auto-apprentissage (self-play)**, où l'IA apprend uniquement par ses propres expériences en jouant contre elle-même.
- **Le processus d'entraînement**, de collecte de données et d'évaluation du modèle.

L'implémentation repose sur des composants interconnectés qui, ensemble, permettent à l'IA de jouer aux échecs de manière optimale. Ce rapport détaillera chaque composant, ses fonctions, et comment ces éléments interagissent pour permettre à l'IA d'apprendre et de s'améliorer de manière autonome.

AlphaZero : Principes fondamentaux

AlphaZero repose sur trois concepts clés pour apprendre à jouer à des jeux complexes :

MCTS (Monte Carlo Tree Search)

MCTS méthode qui permet à l'IA d'explorer les coups possibles en simulant plusieurs parties. Elle combine exploration et exploitation pour guider la recherche de la meilleure décision. MCTS est utilisé pour simuler des parties à partir de chaque état du jeu afin de déterminer le meilleur coup à jouer en utilisant les prédictions du modèle neuronal.

Réseau Neuronal Profond

Le réseau neuronal dans AlphaZero est constitué de deux parties :

- **La Policy Head :** qui prédit la distribution de probabilité des coups possibles à partir de l'état actuel du jeu. Elle sert à guider MCTS en indiquant quels coups sont les plus prometteurs.
- **La Value Head :** qui évalue la qualité de la position, c'est-à-dire la probabilité de victoire pour le joueur actif. Cette valeur est utilisée pour guider l'exploration de MCTS.

Dal Pian Nathan - p2210597
Aboufath Ibtissam - p2110758
5A INFO

Self-play

Le self-play est le processus par lequel l'IA joue contre elle-même pour générer des données d'apprentissage. Après chaque partie, l'IA utilise les données recueillies (positions, coups, résultats) pour améliorer ses prédictions sur les coups futurs. C'est ce processus qui permet à AlphaZero d'apprendre sans intervention humaine et de découvrir de nouvelles stratégies de jeu.

Les composants principaux

Classe *DeepNN* : Le réseau neuronal

Le réseau neuronal est structuré en trois parties principales :

1. **Extraction des caractéristiques :** Un ensemble de couches convolutives et résiduelles traite l'état brut du plateau pour extraire des caractéristiques utiles.
2. **Policy Head :** Cette composante prédit une distribution de probabilité sur les coups possibles, utilisée par MCTS pour prioriser l'exploration des nœuds.
3. **Value Head :** Cette composante évalue la qualité de la position actuelle, ce qui aide à ajuster les estimations de qualité dans l'arbre MCTS.

Détails des composantes principales

1. **Extraction des caractéristiques**

- La partie initiale du réseau traite l'entrée brute sous forme d'un tenseur de taille 8*8*101, qui encode les informations essentielles du jeu. Ce tenseur inclut les 8 dernières positions des pions de chaque joueur, représentées par 6 couches pour les pions blancs et 6 couches pour les pions noirs, soit 6*8*2 couches au total. S'ajoutent 2 couches pour indiquer si le roque est encore possible pour le joueur blanc, 2 couches pour le joueur noir, et enfin 1 couche supplémentaire pour spécifier quel joueur doit jouer. L'ensemble constitue donc une représentation complète de l'état actuel du jeu avec 101 couches au total.
- Des couches convolutives et des blocs résiduels sont utilisés pour capturer des relations complexes entre les pièces et les positions sur le plateau.

2. **Policy Head**

Cette composante prend les caractéristiques extraites et produit une distribution de probabilité (P(s,a)) sur tous les coups possibles. Chaque probabilité représente la confiance du réseau neuronal que le coup correspondant est un bon choix dans l'état actuel. Ces probabilités sont utilisées par MCTS pour prioriser les nœuds à explorer.

3. **Value Head**

Cette composante prend les mêmes caractéristiques extraites et évalue la position actuelle (V(s)).
La valeur est un score entre -1 et 1 :

- −1 : La position est une défaite certaine.
- 0 : La position est équilibrée = égalité probable.
- 1 : La position est une victoire certaine.

Ces valeurs sont remontées dans l'arbre MCTS pour ajuster les estimations des coups explorés.

Classe *MCTS* : Recherche Monte Carlo Tree Search

La recherche Monte Carlo Tree Search (MCTS) est au cœur du processus décisionnel d'AlphaZero. Elle permet d'explorer les coups possibles en simulant des parties, tout en équilibrant deux aspects :

1. **Exploration :** Tester des coups moins connus ou prometteurs pour découvrir de nouvelles stratégies.
2. **Exploitation :** Réutiliser les coups déjà identifiés comme efficaces.

Grâce à MCTS, l'IA peut sélectionner les meilleurs coups en s'appuyant sur les prédictions fournies par le réseau neuronal (P(s,a) et V(s)) et les données des simulations précédentes.

Structure de MCTS

MCTS repose sur une représentation en arbre où :

- Chaque nœud représente un état du plateau.
- Les branches représentent les coups possibles depuis cet état.
- Chaque nœud stocke des informations critiques comme :
    - N(s,a) : Nombre de fois que le nœud a été visité.
    - Q(s,a) : Qualité moyenne du coup associé au nœud.
    - P(s,a) : Probabilité initiale du coup, fournie par le réseau neuronal.

Le processus MCTS suit quatre étapes principales : sélection, expansion, simulation, et rétropropagation.

Étapes principales de MCTS

1. **Sélection**

À partir de la racine de l'arbre, on descend vers les nœuds enfants en choisissant les coups selon une combinaison de :

- Q(s,a) : Qualité moyenne du coup.
- P(s,a) : Probabilité initiale du coup.
- N(s,a) : Nombre de visites du nœud.

Cette combinaison est calculée via la formule UCT, qui équilibre exploration et exploitation.

2. **Expansion**

Lorsque MCTS atteint un nœud non exploré, il est étendu. Tous les coups légaux depuis cet état sont ajoutés comme nœuds enfants, avec leurs probabilités initiales (P(s,a)) calculées par le réseau neuronal.

3. **Simulation**

Une simulation est effectuée à partir de ce nouvel état en utilisant la **prédiction de la Value Head (V(s))** du réseau neuronal. V(s) fournit une estimation de la qualité de l'état final de la simulation.

4. **Rétropropagation**

La valeur V(s) est remontée dans l'arbre, en mettant à jour :

- N(s,a) : Incrémenté pour chaque nœud traversé.
- Q(s,a) : Ajusté pour refléter la qualité moyenne des valeurs remontées.

Informations stockées dans chaque nœud

Chaque nœud de l'arbre MCTS stocke les informations suivantes :

Plateau actuel : L'état du jeu représenté comme un objet chess.Board.

- N(s,a) (visits) : Nombre de fois que ce nœud a été visité.
- Q(s,a) (value moyenne) : Qualité moyenne du coup depuis cet état.
- P(s,a) (probabilité) : Priorité initiale du coup, donnée par le réseau neuronal.
- Parent : Référence au nœud parent pour permettre la rétropropagation.
- Enfants : Liste des nœuds enfants représentant les états atteints par les coups légaux.

Classe *Train* : Gestion de l'apprentissage

La classe Train est responsable de l'ensemble du processus d'apprentissage par auto-entraînement (self-play) d'AlphaZero. Elle orchestre les cycles où :

1. L'IA joue des parties contre elle-même pour collecter des données d'entraînement.
2. Les données générées (s,π(s,a),z) sont utilisées pour entraîner le réseau neuronal.
3. Le modèle est évalué périodiquement pour vérifier les améliorations.

Étapes principales

1. **Self-play :** Génération des données

L'IA joue des parties complètes contre elle-même en utilisant MCTS pour prendre des décisions à chaque coup. Chaque partie produit des données d'entraînement :

- s : Les états du plateau (positions).
- π(s,a) : Les distributions de probabilité normalisées des coups possibles.
- z : Le résultat final de la partie (1, −1 ou 0) qui est initialement à 0 et qui sera mis à jour à la fin de la partie.

Ces données sont stockées dans une mémoire appelée *self_play_data*.

2. **Entraînement du réseau neuronal**

Les données collectées sont utilisées pour mettre à jour les poids du réseau neuronal. Le réseau est optimisé pour :

- Réduire l'écart entre les prédictions P(s,a) et les cibles π(s,a) (policy loss) en utilisant l'entropie croisée.
- Réduire l'erreur quadratique moyenne (MSE) entre V(s) et z (value loss).

Ce qui permet d'améliorer la capacité du réseau à guider MCTS de manière plus efficace.

3. **Évaluation périodique**

Le modèle est comparé à une version précédente pour s'assurer qu'il s'améliore. Si le modèle actuel surpasse le modèle de référence (par exemple, nous avons fixé un taux de victoire > 55 %), il devient le nouveau modèle de référence.

Classe Partie : Gestion d'une partie

La classe Partie est dédiée à la gestion d'une partie complète d'échecs entre deux instances d'IA. Elle s'inscrit dans le processus de self-play et est utilisée pour orchestrer les interactions entre les différents composants :

- Elle utilise la recherche Monte Carlo Tree Search (MCTS) pour déterminer les coups à jouer.
- Elle utilise le réseau neuronal (DeepNN) pour guider les simulations MCTS via P(s,a) et V(s).
- Elle peut afficher visuellement les parties grâce à *pygame*.

Structure de la classe

La classe Partie se concentre sur trois aspects principaux :

- **Initialisation du plateau et des paramètres :** Prépare le plateau d'échecs pour la partie.
- **Gestion des coups :** Utilise MCTS pour jouer les coups.
- **Cycle de jeu complet :** Gère l'enchaînement des coups jusqu'à la fin de la partie.