National University of Computer & Emerging Sciences
Karachi Campus

Checkers AI
Project Proposal
Artificial Intelligence
Section: E

Group Members:
22k-4577 Muhammad Rouhan
22k-4299 Sarim Shah
22k-4587 Moiz ul Haq

Project Proposal

- **Introduction**

  The objective of this project is to develop an intelligent agent capable of playing the game of checkers at a competitive level. Leveraging advanced artificial intelligence techniques, the agent will employ reinforcement learning alongside traditional alpha-beta pruning algorithms to make strategic decisions. The system will feature an interactive user interface, allowing users to play against the AI and observe its decision-making process.

- **Existing System**

  Several implementations of AI-driven checkers games are available:

  - Simple-Checkers-AI: A Python-based implementation using Pygame and the easyAI library. While functional, it is noted for its simplicity and occasional performance issues.
  - Checkers-AI: Developed with HTML, CSS, and JavaScript, this project offers an interactive UI and incorporates various AI algorithms for gameplay.
  - Checkers-Reinforcement-Learning: Focuses on reinforcement learning techniques to train the AI, providing tools necessary for training and evaluation.

  These projects serve as foundational references but exhibit limitations in scalability, efficiency, and user engagement.

- **Problem Statement**

  Existing AI checkers implementations often face challenges such as:

  - Limited Learning Capabilities: Many systems rely solely on predefined algorithms without the ability to learn and adapt from gameplay experiences.
  - Performance Constraints: Some implementations experience delays and inefficiencies, particularly with deeper search depths in decision-making algorithms.
  - User Interface Limitations: A lack of intuitive and engaging interfaces can hinder user experience and accessibility.

- **Proposed Solution**

  Our project aims to address these issues by:

  - Integrating Reinforcement Learning: Implementing Q-learning to enable the AI to learn optimal strategies through self-play and adapt over time.

- Optimizing Alpha-Beta Pruning: Enhancing the efficiency of the decision-making process by implementing depth-limited search with alpha-beta pruning, reducing computational overhead.
- Developing a User-Friendly Interface: Creating an interactive and responsive UI using modern web technologies to facilitate seamless human-AI interaction.

- **Salient Features**
  - Adaptive AI Opponent: An AI that improves its gameplay strategies through continuous learning.
  - Efficient Decision-Making: Utilization of optimized search algorithms to ensure quick and strategic moves.
  - Engaging User Interface: A visually appealing and intuitive interface that enhances user experience.
  - Performance Analytics: Tools to analyze game performance, providing insights into AI decision-making and user gameplay patterns.

- **Tools & Technologies**
  - Programming Language: Python for backend logic and AI algorithms.
  - Frameworks: Pygame for game development; TensorFlow or PyTorch for implementing reinforcement learning models.
  - Operating System: Cross-platform compatibility with Windows and Linux environments.

- **References**

  - https://github.com/techwithtim/Python-Checkers-AI
  - https://github.com/sramakrishnan247/Checkers-AI
  - https://github.com/SamRagusa/Checkers-Reinforcement-Learning