AI-BASED CHECKERS GAME

ARTIFICIAL INTELLIGENCE

PROJECT REPORT

BATCH 2023

BAI – 4A

GROUP MEMBER #1:	**RAO GHULAM MOHI UDDIN**	**23K-0001**
GROUP MEMBER #2:	**SHEIKH NAVEED**	**23K-0003**
GROUP MEMBER #3:	**ARSH AL AMAN**	**23K-0078**
GROUP MEMBER #4:	**SYED MUNEEB UR REHMAN**	**23K-0038**

Course Instructor: Mr. Syed Bilal Ahsan

Lab Instructor: Ms. Ramsha Jat

DEPARTMENT OF ARTIFICIAL INTELLIGENCE

NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES – FAST

KARACHI CAMPUS

Table of Contents

Abstract:

This project implements a classic Checkers game with an AI opponent using the Minimax algorithm with alpha-beta pruning. The game features a graphical user interface (GUI) built with Pygame, allowing players to compete against an AI that evaluates board positions based on material advantage, mobility, and center control. The AI adheres to standard Checkers rules, including forced captures and king promotions, providing a challenging opponent for human players.

Introduction:

Checkers is a two-player strategy game played on an 8×8 board. The objective is to capture all opponent pieces or block them from making legal moves. This project enhances the traditional game by introducing an AI opponent capable of making intelligent decisions. The AI uses a heuristic evaluation function combined with the Minimax algorithm to determine optimal moves, ensuring a competitive gameplay experience.

Problem Statement:

- Implementing a fully functional Checkers game with enforced rules (e.g., forced captures, king promotions).
- Developing an AI opponent that can evaluate board states and make strategic decisions.
- Creating an intuitive GUI for smooth player interaction.

Project Overview:

The project consists of two main components:

1. **Backend (Game Logic & AI)**
    - Board class manages game state, piece movements, and win conditions.
    - AI_Algo class implements Minimax with alpha-beta pruning for decision-making.
    - Evaluation function considers material, mobility, and center control.
2. **Frontend (GUI)**
    - Pygame-based interface for rendering the board and handling user input.
    - Visual feedback for invalid moves, captures, and game outcomes.

Development Methodology:

1. Board Representation

- The game board is represented as an 8×8 grid.
- Pieces are stored as characters ('R' for Red, 'B' for Black, 'RK'/'BK' for kings).
- Movement validation enforces Checkers rules (diagonal moves, forced captures).

2. AI Algorithm

- **Minimax with Alpha-Beta Pruning**
  - Searches possible moves up to a depth of 3.
  - Reduces computation time by pruning irrelevant branches.
- **Heuristic Evaluation**
  - **Material Score:** Values pieces (100 for pawns, 150 for kings).
  - **Mobility Score:** Rewards pieces with more movement options.
  - **Center Control:** Encourages occupying central squares.

3. Forced Captures & Multi-Jumps

- The AI checks for mandatory captures before making a move.
- If a capture sequence exists, it prioritizes maximizing captured pieces.

4. GUI Implementation

- Pygame handles rendering, mouse input, and game state updates.
- Visual indicators for selected pieces, valid moves, and game outcomes.

Conclusion:

The project successfully delivers a playable Checkers game with a competent AI opponent. Key achievements include:

- A rule-compliant Checkers implementation.
- An AI that adapts to board dynamics using Minimax.
- A responsive GUI for seamless player interaction.

Future Plans:

1. **Improved AI**
   - Increase search depth for stronger decision-making.
   - Optimize evaluation function weights (e.g., king safety, endgame strategies).
2. **Enhanced GUI Features**
   - Animated piece movements.
   - Move hints for beginners.
3. **Multiplayer Mode**
   - Online play via networking.
   - Local two-player option.

References:

1. Russell, S., & Norvig, P. (2020). *Artificial Intelligence: A Modern Approach*. Pearson.
2. Pygame Documentation. (n.d.). Retrieved from https://www.pygame.org/docs/
3. Checkers Rules. (n.d.). *World Checkers Draughts Federation*.