Project Title: *IntelliWord: AI-Powered Multi-Mode Word Search Game*
Submitted By: Muhammad Sameer (22k-4359), Shaheer Mustafa (22k-4356), Amna Mubashir (22k-4342)
Course: AI
Instructor: Shafique Rehman
Submission Date: 11-5-2025

---

1. Executive Summary

Project Overview:
This project aimed to develop an AI-powered word search game called *IntelliWord*, which introduces innovation over traditional word search games through multiple challenging modes and AI competition. The game features four unique modes—Grid Shuffle, Word Bomb, Word Chain, and Fog of War—each with three difficulty levels. A directionally constrained Depth-First Search (DFS) AI agent was implemented as a goal-based agent to locate valid words. The AI consistently finds the correct words, with an average win rate of 70% against human players due to its precision and speed.

---

2. Introduction

Background:
Traditional word search games involve finding words in a static grid of letters. *IntelliWord* reimagines this classic game by introducing dynamic gameplay modes, AI opponents, and real-time simultaneous play. The project was selected to explore AI algorithms in a game environment requiring visual perception, pattern recognition, and strategic adaptation.

Objectives of the Project:

- Implement multiple interactive game modes with unique challenges.
- Develop a real-time AI opponent using an efficient word-searching algorithm.
- Test AI performance across increasing difficulty levels and against human players.
- Create a visually appealing, responsive GUI using Pygame.

---

3. Game Description

Original Game Rules:
In the traditional word search game, a grid of letters contains hidden words that players must find by tracing letters in sequence, typically in horizontal, vertical, or diagonal directions.

Innovations and Modifications:

- **Grid Shuffle Mode**: After each word is found, the entire grid is shuffled, increasing difficulty.
- **Word Bomb Mode**: One word is randomly designated as a bomb; players must find it before the timer runs out.
- **Word Chain Mode**: Words must be found in a given sequence.
- **Fog of War Mode**: Only a portion of the grid can be visible using reveal points, and the rest is revealed as players find words and gain more reveal points.
- Three difficulty levels (Easy, Medium, Hard) adjust grid size and word complexity.
- Real-time AI competition instead of turn-based gameplay.

---

4. AI Approach and Methodology

AI Techniques Used:
A directionally constrained Depth-First Search (DFS) algorithm was used. The AI acts as a goal-based agent, recursively searching the grid for valid words from the word list.

Algorithm and Heuristic Design:
Instead of heuristic-based approaches, we opted for a DFS with directional constraints to efficiently and correctly search all possible word paths. The algorithm is optimized to consider only valid moves (horizontal, vertical, diagonal) and backtrack when necessary. A delay was added to simulate human-like thinking speed. Once it reaches the second letter of the word , the algorithm knows the flow of the word is in a certain direction , so it goes in the same flow for the whole words, if the word is not found then it recursively comes back to the initial letter.

AI Performance Evaluation:
The AI achieved 100% word-finding accuracy. In Grid Shuffle mode, it won approximately 70% of games against human players due to superior precision. Average decision time per word was kept human-competitive by introducing artificial delay.

---

5. Game Mechanics and Rules

Modified Game Rules:

- Players must find words in various challenging conditions (shuffling grid, bomb timer, fog restrictions).
- The AI plays simultaneously with the human.
- In most modes, the game ends when all words are found.
- Scores are based on the number of correctly identified words.

Turn-based Mechanics:
There is no turn-based system. Both the AI and human play simultaneously, and the grid responds in real-time.

Winning Conditions:
The player (AI or human) with the most words found at the end of the game wins. In Word Bomb mode, failure to find the bomb word within the time results in an automatic loss.

---

6. Implementation and Development

Development Process:
We began by designing the core functionalities in Python—grid creation, word placement, word searching, and game logic for each mode. Once the core mechanics were stable, we developed the GUI using the Pygame library. Sound effects, user interactions, and visuals were gradually added and tested.

Programming Languages and Tools:

- **Programming Language**: Python
- **Libraries**: Pygame
- **Tools**: GitHub (for version control and collaboration)

Challenges Encountered:

- Selecting the optimal search algorithm: Heuristics were initially considered, but DFS with directional constraints was ultimately more effective.
- Synchronizing real-time AI and human actions required careful threading and timing.
- Implementing fog of war and real-time shuffling without breaking the UI or logic flow was complex.

---

7. Team Contributions

- **Muhammad Sameer (22k-4359)**: Core logic implementation, DFS search algorithm, game mode logic (shuffling, word chain, bomb, fog of war).
- **Shaheer Mustafa (22k-4356)**: GUI development with Pygame, sound effects integration, and incorrect word handling.
- **Amna Mubashir (22k-4342)**: Research and testing for optimal AI algorithms, Fog of War logic, and evaluation of AI performance against human players.

---

8. Results and Discussion

AI Performance:

- AI consistently found correct words with 100% accuracy.
- Win rate: 70% against human players in Grid Shuffle mode.
- The inclusion of delayed AI actions provided a fair and engaging experience.
- Dynamic modes introduced a high level of replayability and challenge, both for AI and human players.

---

9. References

- Pygame Documentation: https://www.pygame.org/docs/

- Python Official Documentation: https://docs.python.org/3/

- DFS Algorithm Reference(but we made our own custom DFS for this game (directionally constraint depth first search)): https://en.wikipedia.org/wiki/Depth-first_search