AI Wordle Solver: A Comparative Study of Artificial Intelligence Algorithms

Course: CSL304 - Artificial Intelligence (Major Project)
Date: November 2025
Group: 35

Name1 =Shashank Yadav

Name2 =Akshat Gupta

Name3 =Aman Kumar

Name4 =Ashutosh Kumar Jha

Executive Summary

This project presents a comprehensive implementation and comparative analysis of five distinct artificial intelligence algorithms applied to the Wordle puzzle-solving domain. The system demonstrates multiple AI paradigms through a production-ready web application that enables real-time algorithm comparison, performance benchmarking, and interactive visualization. The implementation features a modular architecture built on the Streamlit framework and provides detailed per-attempt analysis through an interactive dashboard. This work serves as both a practical puzzle-solving tool and an educational resource demonstrating the application of diverse AI techniques to constraint-based problem-solving.

---

1. Introduction

1.1 Project Motivation and Objectives

The Wordle puzzle, which requires identifying a five-letter word within six attempts using color-coded feedback, presents an ideal testbed for artificial intelligence algorithms. This project aims to implement, compare, and analyze five fundamentally different AI approaches to demonstrate how various computational intelligence paradigms address the same problem with differing strategies and performance characteristics.

Primary objectives include developing a multi-algorithm solver system with consistent interfaces, creating an interactive web-based platform for real-time algorithm comparison, conducting comprehensive performance benchmarking, and providing detailed visualization of solver decision-making processes.

1.2 Problem Statement and Scope

Wordle presents a constrained optimization problem where an AI agent must efficiently narrow down possible solutions using limited feedback. After each guess, the system provides three types of information: correct letters in correct positions (green), correct letters in wrong positions (yellow), and letters not present in the target word (gray). The challenge lies in selecting guesses that maximize information gain while minimizing total attempts.

This project implements five distinct solver approaches: Constraint Satisfaction Problem (CSP) solving, Knowledge-Based reasoning, Bayesian probabilistic inference, Reinforcement Learning with Q-learning, and Genetic Algorithm evolution. Each algorithm is evaluated on win rate, average attempts to solution, and computational efficiency.

---

2. Technical Architecture

2.1 System Design

The system employs a layered architecture with clear separation of concerns. The presentation layer consists of a Streamlit web interface providing game board visualization, solver selection controls, real-time statistics panels, and a detailed algorithm comparison dashboard.

The game logic layer manages core Wordle mechanics through a centralized game engine responsible for state management, feedback generation, and win/loss condition evaluation. The AI solver layer implements five distinct algorithms, all inheriting from a common BaseSolver abstract class ensuring consistent interfaces while allowing algorithm-specific implementations.

The data and utilities layer provides foundational services including word list management, input validation, feedback processing, performance metrics tracking, and comprehensive logging capabilities.

2.2 Core Implementation

The game engine maintains essential state information including the secret word, guess history, attempt count, and complete feedback history. Feedback generation employs a two-pass algorithm ensuring correct handling of duplicate letters: first marking exact position matches, then identifying present letters in wrong positions.

All solvers inherit from the BaseSolver abstract class defining required methods for generating next guesses, updating internal state based on feedback, resetting for new games, and providing statistical information. This abstraction enables consistent treatment of diverse algorithms while maintaining flexibility for algorithm-specific optimizations. A factory pattern enables dynamic solver instantiation and simplified extension with new algorithms.

---

3. Algorithm Implementations

3.1 Constraint Satisfaction Problem (CSP) Solver

The CSP solver models Wordle as a constraint satisfaction problem where each letter position has associated constraints derived from feedback. The algorithm maintains constraint sets for possible letters at each position and applies arc consistency techniques to propagate constraints across positions. After each guess, constraints are updated based on feedback: correct positions restrict to single letters, wrong positions eliminate letters from specific locations, and absent letters are removed from all positions.

This approach demonstrates strong performance with deterministic and predictable behavior, making it ideal for applications requiring logical consistency and explainable decision-making. The algorithm efficiently filters candidate words and maintains rapid execution speed.

3.2 Knowledge-Based Solver

The knowledge-based approach employs rule-based reasoning combined with statistical letter frequency analysis. The algorithm maintains distributions of letter frequencies across positions in the English language and scores candidate words based on these distributions weighted by confirmed and present letter information. After each feedback cycle, the system updates its knowledge base and recalculates word scores using inference rules.

3.3 Bayesian Probabilistic Solver

The Bayesian solver applies information theory principles to maximize expected information gain with each guess. The algorithm calculates Shannon entropy for all candidate words, considering every possible feedback pattern and its probability. Bayes' theorem updates probability distributions after each guess, with the system selecting words that most effectively partition the remaining solution space.

The mathematical foundation involves computing information gain as the difference between current entropy and expected post-feedback entropy across all possible feedback patterns. This theoretically optimal approach achieves the highest win rate with the best average attempts per solution, though computational complexity results in slightly longer processing time. This algorithm excels in scenarios where solution optimality justifies additional computational cost.

3.4 Reinforcement Learning Solver

The reinforcement learning implementation employs Q-learning with epsilon-greedy exploration strategy. The algorithm maintains a Q-table mapping state-action pairs to expected values, where states encode feedback history and actions represent word selections. The system balances exploration of untested strategies with exploitation of learned optimal policies.

Q-value updates follow the standard temporal difference learning formula, incorporating rewards that incentivize wins while penalizing excessive attempts. The learning rate and discount factor control update magnitude and future reward consideration. After training across multiple episodes, this approach demonstrates adaptive learning capabilities valuable in dynamic environments where prior training is feasible.

3.5 Genetic Algorithm Solver

The genetic algorithm applies evolutionary computation principles, maintaining a population of candidate solutions that evolve through selection, crossover, and mutation operations. Each generation scores individuals based on letter frequency and position preferences derived from feedback. Top performers are selected as parents for the next generation, with crossover combining parent characteristics and mutation introducing diversity to prevent premature convergence.

---

4. Performance Analysis

4.1 Comparative Results

Extensive testing reveals distinct performance characteristics for each algorithm. The Bayesian solver demonstrates superior overall performance with the highest win rate and lowest average attempts, though requiring more processing time. The CSP solver achieves an excellent balance between accuracy and speed with rapid execution.

The Knowledge-Based solver exhibits the fastest execution while maintaining strong win rates. Reinforcement Learning performs well after adequate training, demonstrating adaptive capabilities. The Genetic Algorithm provides unique exploratory value despite being computationally intensive.

These results demonstrate that while the Bayesian approach offers theoretical optimality, the CSP and Knowledge-Based solvers provide compelling alternatives when considering the accuracy-speed tradeoff. There are some words which only algorithms can correct but other ffails to guess due to different words selection

---

5. User Interface Features

5.1 Interactive Web Application

The Streamlit-based interface provides comprehensive functionality through multiple integrated components. The game board display renders the Wordle grid with color-coded feedback visualization, maintains guess history, and supports both manual and automated play modes. The solver selector enables runtime algorithm switching with descriptive information and configurable parameters for each approach.

5.2 Algorithm Comparison Dashboard

A sophisticated comparison dashboard enables side-by-side algorithm analysis with per-attempt candidate word tracking and scoring. Users can select any two algorithms for simultaneous comparison, input custom target words for testing, and view detailed breakdowns showing top candidate words with scores at each attempt stage.

Visualization charts reveal how each algorithm's confidence evolves as feedback accumulates. Win/fail status indicators provide immediate performance assessment. This dashboard serves educational purposes by illuminating fundamental differences in algorithm decision-making processes and strategy evolution.

---

6. Conclusions

This project successfully demonstrates the application of five fundamentally different artificial intelligence paradigms to a well-defined constraint-based puzzle-solving problem. The implementation achieves production-ready status with clean architecture, comprehensive testing, and professional documentation. Performance benchmarking provides quantitative evidence for algorithm selection decisions across diverse application scenarios.

The modular design with abstract base classes and factory patterns ensures extensibility for future algorithm additions. Comprehensive visualization through the interactive dashboard provides educational value by making abstract AI concepts concrete and observable. The system serves dual purposes as both a practical Wordle-solving tool and a pedagogical resource for understanding AI problem-solving approaches.