Michael Crevier
CS 499 Capstone
Milestone Three: Data Structures and Algorithms Enhancement

Artifact Description

The artifact I selected for further enhancement is a Python-based AI word strategy game that evolved from a Q-learning maze-solving agent I developed in CS-370: Current and Emerging Trends in Computer Science (Oct–Dec 2024). In Enhancement 1, I restructured the original maze AI into a modular, turn-based word game engine. For this second enhancement, I focused on enriching the AI's intelligence through integrated predictive modeling. The primary goal was to develop a system capable of intelligently guessing a player's word using context-aware analysis and statistical reasoning.

This enhancement involved implementing several algorithmic techniques: Markov Chains that model character transitions based on linguistic probabilities derived from a word corpus, Monte Carlo Tree Search (MCTS) for simulating and evaluating potential word guesses, word frequency modeling that ranks candidate guesses based on real-world usage, and Naive Bayes classification that leverages conditional probabilities to identify likely player choices. These models are orchestrated through a reinforcement learning Q-learning meta-layer, which dynamically assigns weights to each model's output and refines its strategy based on historical performance.

Justification for Inclusion

I selected this artifact because it demonstrates my ability to design, implement, and coordinate complex algorithms in a collaborative framework to solve a non-trivial problem. The integration of multiple AI components creates a sophisticated prediction system that showcases my algorithmic and data structure proficiency.

The components of this system highlight my algorithmic design skills. Markov Chains model character transitions based on linguistic probabilities, enabling the AI to construct likely word candidates from shared letters and develop intuition for letter placement within words. Monte Carlo Tree Search enables the AI to simulate various word guesses through randomized search trees, making it ideal for evaluating move sequences with incomplete information and balancing exploration with exploitation. Word Frequency Ranking incorporates a large English word corpus and ranks candidate guesses based on their real-world frequency and strategic value, prioritizing more common or high-value words during prediction. Naive Bayes Classification utilizes features such as letter pool distributions, shared letters, and player behavior patterns to predict likely player word choices through probability-based classification. The Q-learning Meta Layer serves as an orchestration mechanism that trains over time to determine optimal model combinations for different scenarios, adjusting weightings based on observed outcomes and game context.

This enhancement also demonstrates my ability to balance intelligence and responsiveness by integrating performance safeguards like turn-level time budgeting and fallback models. By harmonizing these techniques, I've demonstrated proficiency in designing algorithmic systems that transcend standalone logic to work synergistically through model-learning strategies.

Outcome Alignment

This enhancement directly aligns with the Algorithms and Data Structures outcome of the Computer Science program. Specifically, I demonstrated the ability to evaluate and implement appropriate algorithmic strategies for a problem space characterized by uncertain or partial information, manage computational trade-offs between performance constraints and AI depth by

introducing simulation time budgets and fallback models, and leverage real-world language data to design efficient lookup and prediction structures.

This enhancement builds directly on the object-oriented, testable, and extensible architecture developed in Enhancement 1, validating its flexibility for integrating intelligent behavior. The game engine now supports adaptive decision-making, which will be further strengthened in Enhancement 3 through persistent database learning and large-dataset training. My outcome-coverage plans remain aligned with what I outlined in Module One, ensuring a cohesive development trajectory across all enhancements.

Reflection on Enhancement Process

Enhancing this artifact presented significant algorithmic and implementation challenges, as its one of my most ambitious and largest projects to date. The primary challenge was developing an AI that feels both intelligent and fair. Since the AI lacks access to the player's full letter pool, each model needed to operate within realistic constraints—shared letters and word length—while still making predictions that felt plausible and strategic. Integrating multiple models presented another significant challenge: coordinating their outputs and reconciling conflicting predictions. I addressed this by designing the Q-learning layer not as a simple model selector, but as a dynamic weight assigner and refiner. This transformed Q-learning into a model orchestration strategy that adapts over time, making the AI responsive to evolving player behavior. Performance management was equally crucial, particularly with computationally expensive algorithms like MCTS. I implemented turn-level time budgeting and layered fallback logic to address this concern. When MCTS exceeded its allocated time window, the system defaulted to a combination of Markov predictions and word frequency rankings, ensuring the AI remained responsive even during complex evaluations. This functionality will be one of the

most positively affected by the addition of persistent data logging that will be the core of Enhancement 3, as it will gain the ability to train itself in between games instead of having to run totally on the fly within each game.

This enhancement deepened my understanding of AI strategy in dynamic game environments. Unlike the original maze project with its static environment and fixed rewards, the word strategy game introduces unpredictability and incomplete information. Designing algorithms that thrive in this context sharpened my skills in probabilistic modeling, heuristic design, and adaptive reinforcement learning. Most importantly, this enhancement confirmed the architectural decisions made in Enhancement 1, demonstrating that the modular design successfully accommodated complex algorithmic integration while maintaining system cohesion.