Reinforcement Learning

Exploring learning agents through a Pacman environment

Vin Anand, 2019

Abstract

This project explored the design and implementation of reinforcement learning algorithms for sequential decision-making under uncertainty. The objective was to construct intelligent agents capable of learning optimal behavior through interaction with an environment and through planning based on probabilistic models. Two fundamental reinforcement learning paradigms were implemented: model-based planning using value iteration and model-free learning using Q-learning. These algorithms were evaluated within simulated environments, including Gridworld and Pacman, where agents learned policies that maximized long-term cumulative reward. The project demonstrated how intelligent behavior emerges from iterative value estimation, reward-driven feedback, and exploration strategies, and provided a conceptual foundation for modern AI systems that autonomously perform multi-step tasks.

Introduction

Reinforcement learning addresses a central problem in artificial intelligence: how an agent should act when outcomes unfold over time and the consequences of decisions are uncertain. Unlike supervised learning, where correct answers are provided explicitly, reinforcement learning requires agents to discover successful strategies through interaction with an environment and feedback in the form of rewards. The challenge lies in balancing short-term gains against long-term outcomes while operating under incomplete knowledge.

The purpose of this project was to implement reinforcement learning algorithms from first principles and observe how optimal behavior can arise without explicit programming of strategies. The project was structured to contrast two complementary approaches to intelligence. The first approach assumes the environment is known and allows an agent to compute optimal behavior through planning. The second assumes no prior knowledge and requires the agent to learn purely from experience. Together, these approaches illustrate the foundations of modern decision-making systems used in robotics, recommendation systems, and increasingly in autonomous AI agents.

Problem Formulation

The environments used in the project were formalized as Markov Decision Processes (MDPs), defined by a set of states, available actions, probabilistic transitions between states, and rewards associated with those transitions. At each step, the agent selects an action, receives a reward, and transitions to a new state. The goal is to learn a policy that maximizes the expected cumulative discounted reward over time.

The inclusion of a discount factor introduces a preference between immediate and delayed rewards, forcing the agent to reason about long-term consequences. This formulation captures many real-world problems, including navigation, resource allocation, and multi-step planning tasks.

Methodology and Implementation

Model-Based Planning: Value Iteration

The first component of the project involved implementing value iteration, a dynamic programming algorithm that computes optimal behavior when the environment's dynamics are known. The algorithm repeatedly applies the Bellman optimality update, which estimates the value of each state as the maximum expected return achievable by any action.

In implementation, the agent iteratively evaluated all states and calculated expected outcomes by summing over possible future transitions weighted by their probabilities. A key methodological detail was the use of synchronous (batch) updates, ensuring that each iteration relied only on values from the previous iteration rather than partially updated results. This preserved the theoretical guarantees of convergence.

From the resulting value function, the optimal policy was derived by selecting the action that maximized expected future reward. Terminal states were treated separately to prevent propagation of artificial future value. Through repeated updates, values propagated backward from reward states, illustrating how long-term incentives shape local decisions.

This implementation demonstrated how an agent can compute optimal strategies entirely through reasoning, without interacting with the environment.

Model-Free Learning: Q-Learning

The second component shifted from planning to learning. In this setting, the agent did not know transition probabilities or reward structures in advance and instead learned through experience. A tabular Q-function was implemented to estimate the expected return of taking a given action in a given state.

Learning occurred through temporal difference updates, where estimates were incrementally adjusted toward observed outcomes. Each interaction produced a transition consisting of a state, action, reward, and next state. The Q-value update combined immediate reward with the estimated value of the best future action, allowing the agent to bootstrap knowledge from partial experience.

An epsilon-greedy exploration strategy was implemented to ensure that the agent continued to explore unfamiliar actions while gradually exploiting learned knowledge. This mechanism prevented premature convergence to suboptimal behaviors and enabled discovery of higher-reward strategies.

Over repeated episodes, the agent's behavior evolved from random exploration to structured, goal-directed decision making, demonstrating how intelligent policies can emerge solely from reward feedback.

Behavioral Analysis and Reward Design

A separate analytical component explored how modifying parameters such as discount factors, transition noise, and living rewards altered agent behavior. By adjusting these values, different policies were induced, including risk-seeking strategies, cautious long-horizon planning, and avoidance of terminal states.

This exercise highlighted a critical insight: reinforcement learning behavior is shaped primarily by incentive design rather than algorithmic complexity. Small changes in reward structure produced dramatically different behaviors, reinforcing the importance of alignment between objectives and incentives.

Results and Outcomes

The value iteration agent successfully converged toward optimal policies, demonstrating that planning algorithms can compute ideal strategies when environmental dynamics are known. Visualizations showed value estimates propagating through the environment until stable policies emerged.

The Q-learning agent exhibited a different but equally important phenomenon. Initially, actions appeared random, reflecting exploration under uncertainty. Over time, repeated experience improved value estimates, and coherent strategies emerged without explicit programming. In the Pacman environment, the agent learned to avoid ghosts and efficiently collect rewards, illustrating learning through feedback loops.

These outcomes validated core reinforcement learning principles: optimal behavior can arise from iterative estimation, and intelligence can be constructed through reward-driven adaptation rather than predefined rules.

Theoretical Significance

The project implemented several foundational reinforcement learning concepts. Dynamic programming demonstrated how optimal control problems can be solved through recursive reasoning. Temporal difference learning showed how agents can learn efficiently from incomplete experience. Exploration strategies illustrated the necessity of uncertainty-driven behavior, while reward shaping emphasized the central role of incentives in defining outcomes.

These concepts form the basis of modern reinforcement learning systems, including Deep Q Networks and reinforcement learning from human feedback (RLHF).

Relevance to Modern AI and LLM-Based Agents

Contemporary AI systems increasingly operate as agents performing multi-step tasks such as booking travel, managing workflows, or coordinating software tools. Large language models provide reasoning and language understanding, but they do not inherently optimize long-term outcomes. Reinforcement learning supplies the mechanism by which such agents learn effective decision policies.

The environments explored in this project can be viewed as simplified analogues of modern agent workflows. States correspond to partially completed tasks, actions correspond to tool usage or

decisions, and rewards correspond to successful task completion, efficiency, or user satisfaction. The same principles governing Gridworld navigation apply to optimizing sequences of API calls or interactions in real-world systems.

Future Extensions and Proposed Experiments

Building on this foundation, reinforcement learning could be applied to modern AI agents by allowing systems to learn optimal task execution strategies. One natural extension would involve training an LLM-based assistant to decide when to search for information, when to ask clarifying questions, and when to act autonomously, optimizing for task success and minimal interaction cost. Another direction would involve workflow optimization, where agents learn efficient execution sequences for complex processes such as itinerary planning or enterprise operations.

Further experimentation could incorporate user feedback as reward signals, enabling continuous improvement of agent behavior. Reinforcement learning could also be used to optimize cost-aware decision making, allowing agents to choose among tools or models based on performance, latency, and expense.

Conclusion

This project demonstrated that intelligent behavior can emerge from iterative value estimation, feedback-driven learning, and carefully designed incentives. By implementing both planning and learning paradigms, the work provided a comprehensive understanding of how agents reason about long-term consequences and adapt through experience. These principles remain central to modern artificial intelligence, particularly as systems evolve from passive predictors into autonomous agents capable of executing complex real-world tasks. The techniques explored in this project represent foundational mechanisms underlying today's AI decision systems and provide a direct conceptual bridge to the development of reinforcement learning–driven AI agents in contemporary applications.