Detailed Explanation of Eligibility Traces

The Bridge Between Monte Carlo and TD Learning

October 22, 2025

Abstract

Eligibility traces (λ) are a core mechanism in temporal-difference (TD) learning that enables efficient, online credit assignment. They combine the low-bias benefits of Monte Carlo methods with the fast, incremental updates of TD methods, offering a powerful tool for policy evaluation and control.

1 Core Concept: The Credit-Assignment Problem

Eligibility traces address the fundamental challenge in Reinforcement Learning of determining which past states or actions are responsible for a current reward or prediction error.

There are two ways to view eligibility traces. The more theoretical view, which we emphasize here, is that they are a bridge from TD to Monte Carlo methods. When TD methods are augmented with eligibility traces, they produce a family of methods spanning a spectrum that has Monte Carlo methods at one end and one-step TD methods at the other. In between are intermediate methods that are often better than either extreme method. In this sense as depticted 1 eligibility traces unify TD and Monte Carlo methods in a valuable and revealing way.

The other way to view eligibility traces is more mechanistic. From this perspective, an eligibility trace is a temporary record of the occurrence of an event, such as the visiting of a state or the taking of an action. The trace marks the memory parameters associated with the event as eligible for undergoing learning changes. When a TD error occurs, only the eligible states or actions are assigned credit or blame for the error. Thus, eligibility traces help bridge the gap between events and training information. Like TD methods themselves, eligibility traces are a basic mechanism for temporal credit assignment.

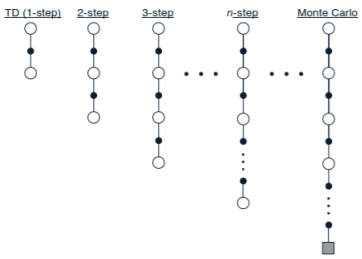

Figure 1: The spectrum ranging from the one-step backups of simple TD methods to the up-until-termination backups of Monte Carlo methods

- **TD(0):** Assigns credit only to the immediate preceding step (high bootstrapping).
- **Monte Carlo ($\lambda = 1$):** Assigns credit equally to all steps in the episode (no bootstrapping).
- **Eligibility Traces ($0 < \lambda < 1$):** Distribute the error backward, decaying the credit exponentially based on time since the visit.

2 The Eligibility Trace $\mathbf{E_t}$

The trace $\mathbf{E_t}(\mathbf{s})$ (or $\mathbf{E_t}(\mathbf{s}, \mathbf{a})$) is a temporary memory that records how **eligible** a state or action is for being updated. This eligibility is a function of:

- **Recency:** How recently the state/action was visited.
- **Frequency:** How often the state/action was visited.

2.1 The Governing Parameters

- $\boldsymbol{\gamma}$ (Discount Factor): Controls the overall value decay of future returns.
- $\boldsymbol{\lambda}$ (Trace-Decay Parameter): Controls the exponential decay rate of the trace itself.

2.2 n-step TD Prediction

What is the space of methods lying between Monte Carlo and TD methods? Consider estimating v_π from sample episodes generated using π. Monte Carlo methods perform a backup for each state based on the entire sequence of observed rewards from that state until the end of the episode. The backup of simple TD methods, on the other hand, is based on just the one next reward, using the value of the state one step later as a proxy for the remaining rewards. One kind of intermediate method, then, would perform a backup based on an intermediate number of rewards: more than one, but less than all of them until termination. This method serves as the theoretical link, using a return calculated n steps into the future.

The methods that use n-step backups are still TD methods because they still change an earlier estimate based on how it differs from a later estimate. Now the later estimate is not one step later, but n steps later. Methods in which the temporal difference extends over n steps are called *n-step* TD methods.

- **n-step Return ($G_{t:t+n}$):**

$$G_{t:t+n} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n V(S_{t+n})$$

- **Update Rule:**

$$V(S_t) \leftarrow V(S_t) + \alpha \left[G_{t:t+n} - V(S_t) \right]$$

3 TD(λ): Temporal Difference with Eligibility Traces

TD(λ) uses the $\boldsymbol{\lambda}$-return (G_t^λ), which is an exponentially weighted average of all n-step returns.

3.1 Forward View TD(λ)

This is the theoretical, non-causal view.

- **λ-Return (G_t^λ):**

$$G_t^\lambda = (1-\lambda) \sum_{n=1}^{\infty} \lambda^{n-1} G_{t:t+n}$$

- **Update Rule:**

$$V(S_t) \leftarrow V(S_t) + \alpha \left[G_t^\lambda - V(S_t) \right]$$

3.2 Backward View TD(λ) (Online Implementation)

This is the practical, causal, and efficient method that relies on the **eligibility trace ($E_t(s)$)**.

- **TD Error (δ_t):** The one-step prediction error.

$$\delta_t = R_{t+1} + \gamma V(S_{t+1}) - V(S_t)$$

- **Online Value Update:**

$$\Delta V(s) = \alpha \delta_t E_t(s) \quad \text{for all states } s$$

4 Control Methods: $Q(\lambda)$ and SARSA(λ)

Eligibility traces extend naturally to action-value (Q-function) control methods, most notably SARSA(λ).

4.1 SARSA(λ)

This is the on-policy control algorithm using action-value traces $\boldsymbol{E_t(s, a)}$.

- **TD Error (δ_t):**
$$\delta_t = R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)$$
- **Online Q-Update:**
$$\Delta Q(s, a) = \alpha \delta_t E_t(s, a) \quad \text{for all state-action pairs } (s, a)$$

5 Eligibility Trace Implementation Types

The two primary methods for updating the trace differ in how they handle repeated visits to the same state/action within an episode.

5.1 1. Accumulating Traces

The trace accumulates on every visit, potentially exceeding a value of 1. It is the standard definition used in basic TD(λ).

- **State Trace Update:**
$$E_t(s) = \gamma \lambda E_{t-1}(s) + \mathbb{1}(S_t = s)$$

5.2 2. Replacing Traces

When a state is visited ($S_t = s$), its trace is reset (or **replaced**) to 1, regardless of its previous value. This leads to better performance, especially with function approximation.

- **State Trace Update:**
$$E_t(s) = \begin{cases} 1 & \text{if } S_t = s \\ \gamma \lambda E_{t-1}(s) & \text{if } S_t \neq s \end{cases}$$

6 Control with Eligibility Traces: SARSA(λ)

SARSA(λ) extends the trace mechanism to action-value functions, $Q(s, a)$, for on-policy control.

6.1 Action Trace $\mathbf{E_t(s, a)}$

The trace is applied to the state-action pair, (s, a).

$$\text{Accumulating Trace: } E_t(s, a) = \gamma \lambda E_{t-1}(s, a) + \mathbb{1}(S_t = s, A_t = a)$$

6.2 SARSA TD Error

The error uses the Q-value for the action A_{t+1} *actually taken* in the next state S_{t+1}.

$$\delta_t = R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)$$

6.3 Online Q-Update

The Q-value of every state-action pair is updated by:

$$\Delta Q(s, a) = \alpha \delta_t E_t(s, a)$$

6.4 Online Value Update

At each time step t, the value function of *every* state s is updated simultaneously:

$$\Delta V(s) = \alpha \underbrace{\delta_t}_{\text{TD Error}} \underbrace{E_t(s)}_{\text{Eligibility Trace}}$$