On-Policy vs Off-Policy Reinforcement Learning

LLM Lab

December 2025

Contents

On-Policy vs Off-Policy Reinforcement Learning **1**
1. The Core Difference . . . 1
1.1 On-Policy RL (e.g., SARSA) . . . 1
1.2 Off-Policy RL (e.g., Q-learning) . . . 2
1.3 Quick Comparison . . . 2
2. The Update Equations . . . 2
2.1 SARSA (On-Policy) . . . 2
2.2 Q-Learning (Off-Policy) . . . 2
2.3 Side-by-Side Comparison . . . 3
3. A Concrete Example: The Cliff Walking Problem . . . 3
3.1 SARSA (On-Policy) Behavior . . . 3
3.2 Q-Learning (Off-Policy) Behavior . . . 3
3.3 Why This Matters . . . 3
4. Connection to LLM Training (RLHF/RLAIF) . . . 3
4.1 Why RLHF is Fundamentally Off-Policy . . . 3
4.2 Implications . . . 4
4.3 Connection to RL-Test . . . 4
5. Summary . . . 4
5.1 Key Differences . . . 4
5.2 When to Use Which . . . 4
6. Further Topics . . . 5
References . . . 5

On-Policy vs Off-Policy Reinforcement Learning

This tutorial provides a clear comparison of **on-policy** and **off-policy** reinforcement learning, using **SARSA** and **Q-learning** as canonical examples. We'll build intuition first, then dive into the math, and finally connect these concepts to modern LLM training.

---

1. The Core Difference

1.1 On-Policy RL (e.g., SARSA)

Learn about the policy you are currently executing.

- The actions you *take* are the actions you *learn from*
- You update values toward the **actual behavior policy**
- Safer and more conservative, but sometimes slower to converge

1.2 Off-Policy RL (e.g., Q-learning)

Learn about a different policy than the one you execute.

- You can follow an exploratory policy (e.g., -greedy)
- But learn a value function for a **greedy/optimal policy**
- More powerful, but potentially less stable

1.3 Quick Comparison

Type	Learns Value Of	Behaves According To
On-policy (SARSA)	The behavior policy	The same policy
Off-policy (Q-learning)	The optimal greedy policy	A more exploratory policy

---

2. The Update Equations

Both SARSA and Q-learning are **temporal difference (TD)** methods that update Q-values based on the difference between predicted and observed returns.

2.1 SARSA (On-Policy)

SARSA updates using the *action actually taken* next:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma Q(s',a') - Q(s,a)\right]$$

where:

Symbol	Meaning
$Q(s,a)$	Current Q-value for state s and action a
α	Learning rate
r	Reward received after taking action a in state s
γ	Discount factor
s'	Next state
a'	**Actual next action taken** (sampled from behavior policy)

Key point: The update uses a', the action *you really took* according to your current policy. This means SARSA tracks the value of your *current behavior*, including exploration.

2.2 Q-Learning (Off-Policy)

Q-learning updates using the *best possible action* next:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma \max_{a'} Q(s',a') - Q(s,a)\right]$$

Key point: The update uses $\max_{a'} Q(s',a')$, the greedy action, regardless of what action was actually taken. This means Q-learning learns the value of the *optimal policy*, even if you behaved suboptimally during exploration.

2.3 Side-by-Side Comparison

Aspect	SARSA	Q-Learning
Update target	$r + \gamma Q(s', a')$	$r + \gamma \max_{a'} Q(s', a')$
Next action	Actual action taken	Best possible action
Policy learned	Behavior policy	Optimal policy

---

3. A Concrete Example: The Cliff Walking Problem

Consider navigating a gridworld with a **dangerous shortcut** — a path along a cliff that offers high reward but risks falling off with large negative penalties.

3.1 SARSA (On-Policy) Behavior

If your behavior is -greedy (occasionally random), SARSA learns:

> "If I follow an -greedy policy, taking this path is risky because I might accidentally fall into the bad state."

Result: SARSA becomes **risk-averse** and learns to avoid the shortcut, taking the safer but longer path.

3.2 Q-Learning (Off-Policy) Behavior

Q-learning assumes:

> "I evaluate the action as if I always take the greedy optimal path — even though I sometimes explore."

Result: Q-learning becomes **risk-seeking** and learns that the shortcut is optimal in expectation, ignoring the exploration noise.

3.3 Why This Matters

Algorithm	Risk Profile	Reason
SARSA	Risk-averse	Accounts for exploration in value estimates
Q-learning	Risk-seeking	Assumes optimal behavior in value estimates

This is the classic "cliff walking" example from Sutton & Barto's RL textbook.

---

4. Connection to LLM Training (RLHF/RLAIF)

Modern LLM training with reinforcement learning has strong connections to off-policy learning.

4.1 Why RLHF is Fundamentally Off-Policy

In RLHF/RLAIF:

- The **reward model** changes over time (updated with new preferences)

- Policies change every iteration
- Exploratory rollouts are required for diverse training data
- Evaluation uses a *different* reward model (fresh RM or verifiers)
- The goal is to learn an **improved policy**, not reinforce the behavior that produced the data

This is fundamentally **off-policy**:

> **Train on data generated by old policies → Evaluate on a new, often improved, reward function.**

4.2 Implications

Aspect	Implication
No fixed validation set	Evaluation depends on a different reward function
Distribution shift	Training data comes from older policy versions
Reward hacking risk	On-policy methods could overfit to the reward model

4.3 Connection to RL-Test

This is why evaluation in RLHF uses:

> Reward from a **new RM + verifiers** (instead of validating on the reward used during RL training)

This mirrors off-policy evaluation in classical RL, where we evaluate a target policy using data collected by a different behavior policy.

---

5. Summary

5.1 Key Differences

Aspect	SARSA (On-Policy)	Q-Learning (Off-Policy)
Learns value of	Behavior policy	Optimal policy
Updates with	Actual next action a'	Greedy next action $\max_{a'}$
Exploration impact	Reflected in updates	Not reflected
Stability	More stable	Less stable
Risk behavior	Risk-averse	Risk-seeking
LLM RL (RLHF/RLAIF)?	Rarely used	Conceptually similar

5.2 When to Use Which

Scenario	Recommended
Safety-critical environments	On-policy (SARSA)
Learning optimal behavior	Off-policy (Q-learning)
Using replay buffers	Off-policy
LLM fine-tuning (RLHF)	Off-policy concepts

---

6. Further Topics

- Visual diagrams comparing SARSA vs Q-learning
- Python code implementing both algorithms
- Connection to PPO, GRPO, DPO, and post-training in modern LLMs
- Why PPO (used in RLHF) is "kind-of" on-policy but used in an off-policy setting
- How evaluation in RLHF is analogous to off-policy evaluation in classical RL

References

1. Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning: An Introduction* (2nd ed.)
2. Watkins, C. J. C. H. (1989). *Learning from Delayed Rewards* (Q-learning)
3. Rummery, G. A., & Niranjan, M. (1994). *On-line Q-learning using connectionist systems* (SARSA)