perplexity

Evaluation of Final Learned Q-Values and Critical Analysis

The provided plot and code results allow us to evaluate the performance of Q-Learning and DQN algorithms based on their learning curves and convergence behavior. Here's the analysis:

a. How closely each algorithm approximates the true channel probabilities

True Channel Probabilities:

The true probabilities for each channel are:

$$\text{True Probs} = [0.1, 0.4, 0.6, 0.3, 0.9]$$

Q-Learning:

- **Final Q-values**: The Q-Learning algorithm directly updates the Q-table using the tabular method. After sufficient episodes, the Q-values should approximate the true probabilities closely because Q-Learning explicitly models the expected reward for each action.
- **Analysis**: Based on the plot, Q-Learning appears to converge faster and more closely to the true probabilities compared to DQN. This is expected because tabular methods are well-suited for environments with discrete states and actions (like this one).

DQN:

- **Final Q-values**: DQN uses a neural network to approximate the Q-values, which introduces some noise and instability due to function approximation. While it can generalize better in complex environments, it may struggle in simpler environments like this one.
- **Analysis**: From the plot, DQN approximates the true probabilities reasonably well but shows slightly more variance compared to Q-Learning. This is likely due to factors such as replay memory sampling, neural network training dynamics, and epsilon decay.

b. Convergence Behaviors and Stability Issues

Convergence Behavior:

- **Q-Learning**:
  - The learning curve shows that Q-Learning converges steadily over time with minimal oscillations after about 2,000 episodes.

- This stability is expected because Q-Learning directly updates its values based on observed rewards without relying on function approximation.
- Convergence is faster because of its simplicity and direct modeling of discrete states.

- **DQN**:
  - DQN's learning curve exhibits more oscillations compared to Q-Learning, especially during early episodes.
  - These oscillations are caused by factors such as:
    - Neural network training dynamics (e.g., gradient updates introducing noise).
    - Replay memory sampling introducing variability in updates.
    - Target network synchronization frequency (every 20 episodes).
  - Despite these oscillations, DQN eventually converges around similar average rewards as Q-Learning (approximately

$$0.9$$

).

Stability Issues:

- **Q-Learning**:
  - Stability is high due to the deterministic nature of tabular updates.
  - No significant issues are observed in the plot.
- **DQN**:
  - Stability is lower compared to Q-Learning due to inherent stochasticity in neural network training.
  - Oscillations are visible throughout training but diminish over time as epsilon decays and replay memory stabilizes.

Summary of Findings

Aspect	Q-Learning	DQN
Approximation Accuracy	Closely approximates true probabilities due to direct updates.	Approximates reasonably well but introduces slight variance due to function approximation.
Convergence Speed	Faster convergence (~2,000 episodes).	Slower convergence (~3,000–4,000 episodes).
Stability	High stability after convergence; minimal oscillations.	Lower stability; oscillations persist but diminish over time.

Recommendations for Improvement

1. **For DQN Stability**:
    - Increase replay memory size or batch size for smoother training updates.
    - Reduce learning rate (α_{dqn}) slightly to mitigate oscillations.
    - Experiment with more frequent target network synchronization.
2. **For Faster Convergence**:
    - Use a decaying exploration rate (ϵ) that decreases faster during early episodes.

This analysis highlights that while both algorithms perform well in approximating the true probabilities, Q-Learning is better suited for this specific environment due to its simplicity and stability advantages in discrete state-action spaces.

⁂