Parnal Sinha, Amisha Singleton
ITAI 2376
A03: NN Zoo

A03 Neural Network Zoo: Reinforcement Learning in Dog Behavior

1. Introduction to Neural Networks and Reinforcement Learning

Neural networks are essential computational models which have been inspired by the anatomy of the human brain. They consist of interconnected neurons organized in layers, allowing models to process information and learn patterns within the latent hypergeometric plane. Reinforcement Learning (RL), a subset of machine learning, focuses on how agents learn optimal behavior through trial and error, guided by rewards and punishments.

2. The Zoo Concept: The Pavlovian Dog as a Reinforced Neural Network

3.

In this "Neural Network Zoo," we represent different neural networks as animals, each symbolizing a unique learning approach. The Dog is an ideal candidate for understanding Reinforcement Learning (RL) due to its ability to learn through Pavlovian conditioning and behavioral reinforcement.

Dog's Learning Process = Reinforcement Learning

1. The dog interacts with its environment (state S)
2. It takes an action A (e.g., sitting when commanded)
3. It receives a reward R (e.g., a treat) or punishment P (e.g., no treat)
4. The dog updates its learned behavior (policy π) to maximize future rewards.

This behavior follows the **Markov Decision Process (MDP)**, where learning is formalized as:

$$(S, A, P, R, \gamma)$$

where:

- S = Set of states (e.g., dog standing, sitting, lying down)
- A = Set of actions (e.g., bark, jump, sit)
- P = Probability of transitioning from one state to another based on an action
- R = Reward function (e.g., treat for sitting)
- γ= Discount factor for future rewards

3. Training a Reinforced Neural Network Dog

Here, we quantify the equations to the behavior of a dog using Pavlovian based behavior

Step-by-Step Dog Training Simulation:

1. Defining the State-Value Function:

$$V^{\pi}(s) = E^{\pi}[\sum_{t=0}^{\infty} \gamma^{t} R_{t+1} \mid S_0 = s]$$

2. Action-Value Function for Specific Behaviors:

Parnal Sinha, Amisha Singleton
ITAI 2376
A03: NN Zoo

$$Q^{\pi}(s,a) = E^{\pi}[\sum_{t=0}^{\infty} \gamma^{t} R_{t+1} \mid S_0 = s, A_0 = a]$$

3. Using Bellman Equations to Optimize Learning:

$$V^{\pi}(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \sum_{s' \in \mathcal{S}} P(s'|s,a)[R(s,a,s') + \gamma V^{\pi}(s')]$$

4. Applying Temporal Difference Learning to Training:

$$V(S_t) \leftarrow V(S_t) + \alpha[R_{t+1} + \gamma V(S_{t+1}) - V(S_t)]$$

Intuition with respect to the neural network dog:

1. The dog's behavior is evaluated using the **state-value function:** $V^{\pi}(s)$
    a. This function determines how rewarding it is for the dog to be in a certain state (e.g., sitting patiently near the owner vs. jumping on guests).
2. The dog's learned behaviors are updated based on the **Q-value function**:
    a. Example: If a dog sits on command (action *A*) in a crowded park (state *S*), it receives treats more frequently (higher Q-value). If it barks instead, it gets scolded (lower Q-value).
3. Using Bellman Equations to Optimize Learning:
    a. The learning process follows the **Bellman equation**.
    b. This equation helps the neural network adjust training based on how likely the dog is to receive a reward for a given action.
4. **Applying Temporal Difference Learning to Training:**
    a. The dog updates its behavior iteratively using **Temporal Difference (TD) learning**.
    b. If the dog correctly sits on command, the expected future reward (a snack) increases, reinforcing this action.

Analysis & Conclusion

Through leveraging the Pavlovian Dog as an analogy for Reinforcement Learning (RL), we were able to contextualize and understand the fundamental principles of how artificial intelligence agents learn through feedback mechanisms – the aspect of biology is again reiterated in this model just like it is in when thinking of neural networks. Just as Pavlov's dog associates a stimulus (bell) with a reward (food), an RL agent associates an action with a reward signal, reinforcing behaviors that maximize positive outcomes over time. This analogy provides an intuitive way to conceptualize policy optimization, value functions, and reward-based decision-making, illustrating how artificial neural networks adjust weights in response to different environmental conditions. The mathematical foundation of RL, particularly the Bellman equation and Q-learning, simplified the understanding of iterative nature of value updates and long-term planning in intelligent systems. Through structured training and consistent reinforcement, both biological and artificial learners refine their decision-making strategies to optimize future rewards.

Beyond its educational benefits, this analogy bridges the gap between theoretical machine learning models and real-world applications. Reinforcement Learning is widely used in robotics, autonomous systems, financial forecasting, and healthcare, demonstrating its versatility in solving complex decision-making problems. Understanding how policy learning and reward mechanisms shape behavior helps students appreciate the power of AI in adaptive learning environments.

Furthermore, integrating RL with neural network architectures, such as Deep Q-Networks (DQNs) and Actor-Critic models, expands its capability to handle high-dimensional state spaces, making it applicable to sophisticated AI challenges. Reinforced learning models are now becoming ubiquitous and is widely utilized in multiple applications. Through drawing parallels between classical conditioning and modern reinforcement learning, we were able to develop a deeper appreciation for both biological learning and artificial intelligence, reinforcing the significance of structured learning paradigms in developing intelligent systems.

References

1. Sutton, R. S., & Barto, A. G. (2018). Reinforcement Learning: An Introduction.
2. MLU-Explain (2025). Reinforcement Learning. Retrieved from https://mlu-explain.github.io/reinforcement-learning/
3. "Reinforcement Learning: AI Meets Pavlov's Dog." *Marvik*, 16 June 2023, https://blog.marvik.ai/2023/06/16/reinforcement-learning-ai-meets-pavlovs-dog/.
4. "A Beginner's Guide to Deep Reinforcement Learning." *Pathmind*, http://wiki.pathmind.com/deep-reinforcement-learning. Accessed 11 Feb. 2025.

Policy and Value Functions

Consider policy π → agent behavior {map states to action}

- **State Value Function ($V^\pi(s)$):** The expected cumulative reward from state s from policy π

$$V^\pi(s) = \mathbb{E}^\pi\left[\sum_{t=0}^{\infty} \gamma^t R_{t+1} \,\middle|\, S_0 = s\right]$$

Expectation under the policy π

↓

The sum of discounted future rewards is averaged over all possible trajectories of an agent based on policy π.

- **Action-Value Function $Q^\pi(s,a)$:**

The expected cumulative reward starting from state s, taking action a, following policy π.

$$Q^\pi(s,a) = \mathbb{E}^\pi\left[\sum_{t=0}^{\infty} \gamma^t R_{t+1} \,\middle|\, S_0 = s, A_0 = a\right]$$

Here, $\mathbb{E}^\pi$ accounts for <u>all</u> possible future state transitions and rewards averaged over all possible actions the agent takes <u>after</u> the first action following π.

∴, intuitively, $\mathbb{E}^\pi$ as computing the expected outcome <u>over all possible paths the agent can take,</u> given that the model chooses actions according to policy π.

Bellman Equations:

These represent <u>recursive</u> relationships for value functions [a → b]

State-Value Function:

$$V^\pi(s) = \sum_{a \in A} \pi(a|s) \sum_{s' \in S} P(s'|s,a)\left[R(s,a,s') + \gamma V^\pi(s')\right]$$

$\pi(a|s)$: Probability of selecting each possible a in s based on policy $\pi(a|s)$

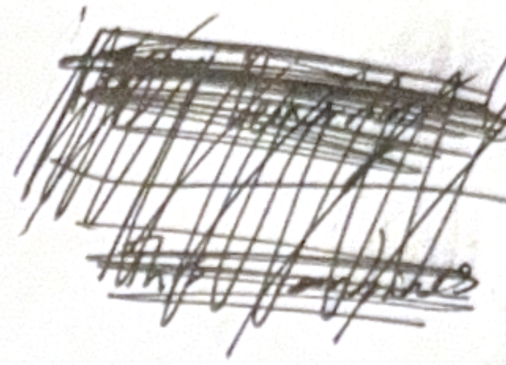
$P(s'|s,a)$: for action a, it ∴ it follows transition is not deterministic ∴ probability as an action a does not always lead to next state s'

Inside the brackets:

- $R(s,a,s')$ equates to the <u>immediate reward</u> from changing to state s'
- $\gamma V^\pi(s')$ = <u>discounted future rewards</u> ⇒ the agent in the model analyzes the future rewards ("looks ahead") at what it expects to earn from s'.
- Given policy π is probabilistic, an agent cannot always know what action it might take ∴ it cannot know where it will end up as the latent data space might be transient ("stochastic").

↳ We <u>MUST</u> sum over all possible expectations to compute an average.

Action-Valued Function:



$Q^\pi(s,a)$ indicates how good an action a in state s is through policy π.

$$Q^\pi(s,a) = \mathbb{E}^\pi\left[\sum_{t=0}^{\infty} \gamma^t R_{t+1} \mid S_0 = s, A_0 = a\right]$$

The logic is very similar to $V^\pi(s)$, however, we start from a state-action pair (s,a)

Bellman: $$Q^\pi(s,a) = \sum_{s' \in S} P(s'|s,a)\left[R(s,a,s') + \gamma \sum_{a' \in A} \pi(a'|s')\, Q^\pi(s',a')\right]$$

Uncertainty in where the agent lands after taking ~~action~~ action a.

Immediate reward for transition

a' = all possible future actions from s' via weights of policy $\pi(a',s')$.·. the agent assumes it follows π afterwards

- In chess, we evaluate every move:

However:

a) We can't predict our opponents move, so we evaluate all scenarios and consider all possibilities.

b) ~~Profit~~ You estimate chances of ~~winning~~ winning by the average of all possible next moves. (At least proficient chess players).

-> Same concept for RL agent: it must evaluate all possible outcomes when taking action a.