Learning frameworks

Supervised learning
- Assumes environment specifies correct output (targets) for each input

Unsupervised learning
- Assumes environment only provides input; learning is based on capturing the statistical structure of that input (efficient coding)

Reinforcement learning
- Assumes environment provides evaluative feedback on actions (how good or bad was the outcome) but not what the correct/best action would have been

Associative reinforcement learning
- Given input, learn to produce output (action) that maximizes immediate reward
- Modified Associative reward-penalty ($A_{R-P}$)
  \[
  \rho(a_j = 1) = \frac{1}{1 + \exp(-n_j)}
  \]
  \[
  \Delta w_{ij} = \begin{cases} 
  \rho(a_j - n_j) a_i & \text{if success} \\
  \lambda \rho((1 - a_j) - n_j) a_i & \text{if failure}
  \end{cases}
  \]

- Reinforcement is broadcast within multilayer network

Adaptive critic
- Feedback can be intermittent, probabilistic, temporally delayed

- Optimal/effective actions are not provided to learner; must be discovered
- Feedback (reinforcement signal) reflects overall consequences of action (and other things) in environment
- Feedback can be intermittent, probabilistic, temporally delayed, and dependent on things outside learner’s control
- Tension between exploration and exploitation
Sequential reinforcement learning

- Execute sequence of actions that maximizes expected sum of discounted future rewards

\[ E \left\{ r[t] + \gamma r[t+1] + \gamma^2 r[t+2] + \ldots \right\} = E \left\{ \sum_{k=0}^{\infty} \gamma^k r[t+k] \right\} \]

- Temporal difference (TD) learning
  - Unit activation at each step should equal expected summed discounted reward

\[ a[t+1]_j = E \left\{ r[t+1] + \gamma r[t+2] + \gamma^2 r[t+3] + \ldots \right\} \quad \text{if unit is "correct"} \]

\[ a[t]_j = E \left\{ r[t] + \gamma r[t+1] + \gamma^2 r[t+2] + \gamma^3 r[t+3] + \ldots \right\} \]

\[ E \left\{ \Delta r[t] \right\} = \Delta a[t]_j \]

- Change weights to reduce difference between \( E \{ r[t] \} \) and \( r[t] \) (reward prediction error)

\[ \Delta w[t]_{ij} = \rho \left( r[t] - E \left\{ r[t] \right\} \right) a_i \quad \text{delta rule} \]

Dopamine and reward prediction error (Shultz et al., 1997)

Response of neurons in substantia nigra (subcortical nucleus) during classical conditioning

- After learning, conditioned stimulus is unexpected and predicts reward (response) but reward itself is now expected (no response)

- Before learning, reward is unexpected and evokes response (reward prediction error)

- After learning, withholding expected reward causes negative prediction error (suppression)
Strengths and limitations of reinforcement learning

**Strengths**
- No need for explicit behavioral targets
- Can be applied to networks of binary stochastic units
- TD learning consistent with some physiological evidence (Schultz)
- Can use associative reinforcement learning (e.g., $A_{R-P}$) to learn actions based on prediction of reinforcement learned by TD

**Limitations**
- Learning is often very slow (not enough information)
- Application to large/continuous state spaces requires some mechanism for function approximation (e.g., multilayer back-propagation network; deep reinforcement learning)
- Associative and TD learning combined only in very simple domains (but deep learning can also be applied to state representations; e.g., auto-encoder)
  - Atari games (Mnih et al., 2013, NeurIPS); Go (Silver et al., 2017, Nature) [deep Q-learning]

Forward models

- Feedback from the world is in terms of *distal* error (observable consequences) rather than *proximal* error (motor commands)
- Would like compute proximal error from distal error (to improve motor commands to achieve goals)
- Relationship between motor commands and observable consequences involves processes in the external world (e.g., physics)
- Learn an internal (forward) model of the world which can be inverted (e.g., back-propagated through) to convert distal error to proximal error
  - Such a model can also provide online outcome prediction to detect errors during execution
Training

- **Forward model: predicted – actual**
  - Generate action randomly, predict outcome
  - Use discrepancy between predicted outcome and actual outcome as error signal

- **Inverse (action) model: desired – actual**
  - Generate action from “intention” in current context
  - Use discrepancy between generated outcome to actual outcome as error signal
  - Back-propagate error through forward model to derive error derivatives for action representation
  - Back-propagate action error to improve inverse model

- Forward and inverse models can be trained at the same time

*Figure 11. A three-joint planar arm.*