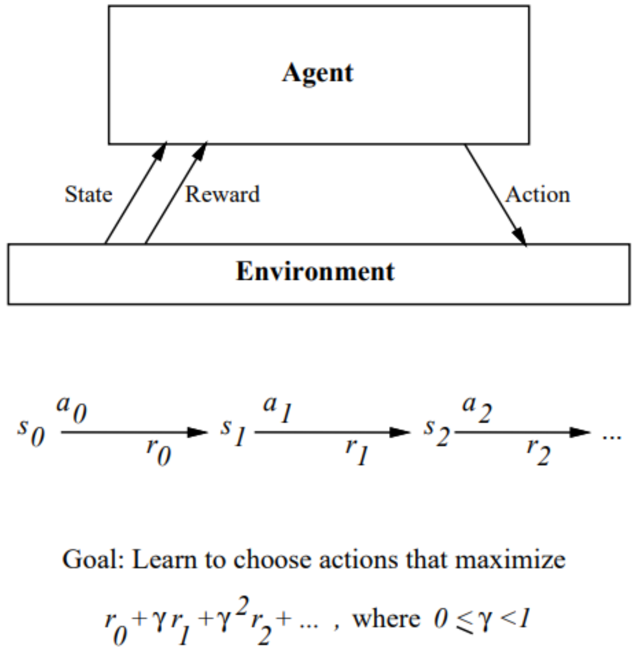
Reinforcement learning (RL) & other ML models

SUMMARY

1. Reinforcement learning

- learning through interaction with the environment
- the RL problem is the general problem of improving the behavior of an artificial agent, based on a feedback to its performance
- RL addresses the problem of how an autonomous agent (that senses and acts in its environment) can learn to choose optimal actions to achieve its goals
- **Applications**
    - learning to control autonomous and mobile robots
    - learning to optimize processes in factories
    - learning to play games
- RL combines **supervised learning** and **dynamic programming** (field of mathematics that has traditionally been used to solve problems of optimization and control)
- **Idea**
    - in RL, the agent is simply given the goal to achieve
        - the agent then learns how to achieve the goal by trial-and-error interactions with its environment
    - each time the agent performs an action in its environment, it receives a *reward* (or *reinforcement*) to indicate the desirability of the resulting state (e.g. + if the game was won, - if the game was lost, 0 in all other states)
    - the task of the agent is to learn from this indirect, delayed reward, to choose sequences of actions that produce the greatest cumulative reward



[1]

- γ is a **discount factor** indicating the importance of future rewards
  - $\gamma=0 \Rightarrow$ only the current rewards matter – Greedy approach
  - γ is usually chosen as 0.95

- a RL task can be viewed as a tuple $<S, A, \delta, r>$, where
  - S is the space of states
    - can be **<u>discrete</u>** or **continuous**
    - RL in continuous state spaces $\Rightarrow$ *Gaussian Processes*
  - A is the action space
  - δ is the transitions function between the states
  - r is the reinforcement function
  - interaction between the agent and the environment
    - at time the agent observes state $s_t \in S$ and chooses action $a_t \in A$
    - then receive the reward r_t
    - and state changes to $s_{t+1} \in S$

- **<u>Markov assumption</u>**
  - the environment in a RL task is a **Markov Decision Process** (**MDP**)
  - s_{t+1} and r_t depend only on current state and action, and not on the entire (state, action) history of the agent in the environment
    - $s_{t+1} = \delta(s_t, a_t)$
    - $r_t = r(s_t, a_t)$
  - functions δ and r may be nondeterministic
  - functions δ and r not necessarily known by the agent
    - POMDP (Partial Observable Markov Decision Processes)

- **agent's learning task**

  - learn action policy $\pi : S \rightarrow A$ that maximizes
  $$E[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots]$$
  from any starting state in S
  - here $0 \leq \gamma < 1$ is the discount factor for future rewards

  Note something new:

  - Target function is $\pi : S \rightarrow A$
  - but we have no training examples of form $\langle s, a \rangle$
  - training examples are of form $\langle \langle s, a \rangle, r \rangle$

  [1]

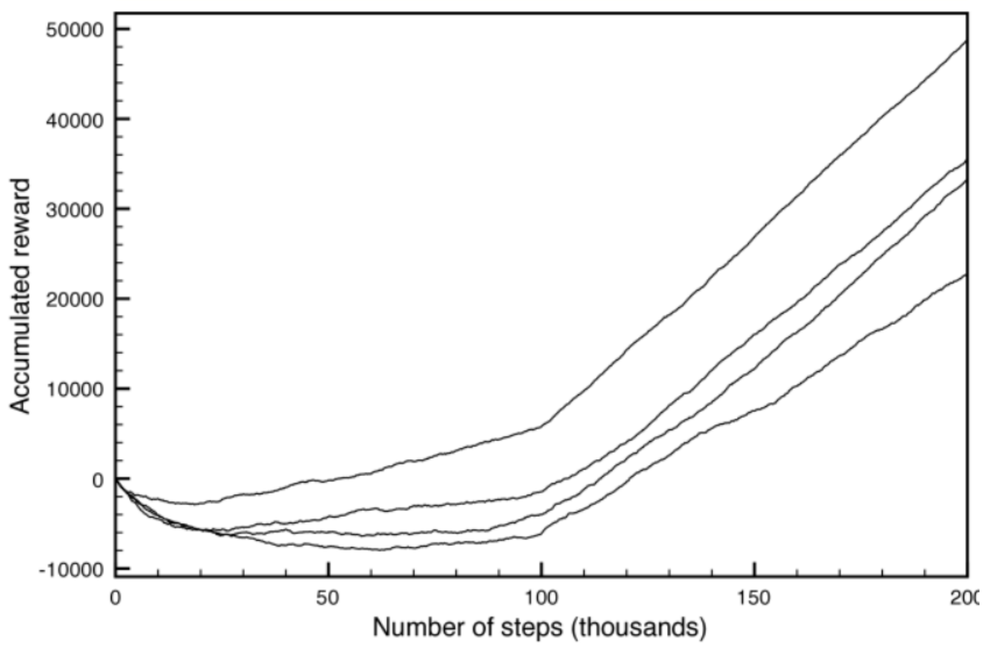
- **two designs** to consider
  - to learn a **utility function** on states (or state histories) and use it to select actions that maximize the expected utility of their outcomes
    - is model based
    - the agent must have a model of the environment
      - it must know the states to which its action will lead
  - to learn an **action-value** function giving the expected utility of taking a given action in a given state
    - is called **Q-learning**
    - is model free
      - the agent must not have a model of the environment
- the learning task can very
  - the environment can be **accessible** or **inaccessible**
    - in an accessible environment, states can be identified with percepts
    - in an inaccessible environment, the agent must maintain some internal state to try to keep track of the environment
  - the agent can begin with a knowledge of the environment and the effects of its actions, or it will have to learn this model as well as utility information
  - rewards can be received only in terminal states, or in any state
    - learning is faster if rewards are received in any state
  - rewards can be components of the actual utility (e.g. points for a ping-pong agent or dollars for a betting agent) or they can be hints as to the actual utility (e.g. "nice move")
  - the agent can be a **(1)** ***passive learner*** or an **(2)** ***active learner*.**
    - **passive learner** – simply watches the environment and tries to learn the utility of being in various states
    - **active learner** – use its problem generator to suggest explorations of unknown portions of the environment
      - trade-off between **exploration** and **exploitation**

- **applications**
  - automation of industry with RL

    - robotics
  - RL in NLP
  - RL in video games
  - Enhancing applications with RL
    - **[Horizon]()**
      - Open source RL platform - Google

Evaluation of RL algorithms

1. how good is the policy the agent finds
   - a standard approach to evaluate a policy in an episodic MDP with discounted long term reward is to run multiple independent trials with the agent using the policy under evaluation.
2. how much reward the agent receives while acting and learning
   - one way to show the performance of a reinforcement learning algorithm is to plot the cumulative reward (the sum of all rewards received so far) as a function of the number of steps.



  - one algorithm dominates another if its plot is consistently above the other.

2. RL related research topics

- Fuzzy RL
- Bayesian RL
- Hybrid ML models
  - RL+HMM (Hidden Markov Models)
  - RL+clustering (adaptive clustering)
- SVMs/DTs/ANNs for approximating the Q function in Q-learning

- Deep Reinforcement Learning
- Actor-critic methods

3. Other learning models

- Association Rules (AR) mining
  - Classification based on ARs
- Inductive Logic Programming
- Hidden Markov Models (HMMs)
  - statistical model
  - connection to Bayesian networks
- Few shot learning
  - feeding a learning model with a very small amount of training data, contrary to the normal practice of using a large amount of data.
  - One-shot learning
    - object categorization in computer vision
    - learn information about object categories from one, or only a few, training samples/images.
    - Siamese neural networks
  - Less than one shot-learning
  - Zero-shot learning
    - predict the category for classes that were not observed during training
    - computer vision, natural language processing

- Semi-supervised learning
  - SS classification, SS regression
  - **one-stage** SSL (1) vs **multi-stage** SSL (2)
    - **(1)** – integrating both labeled and unlabeled data in a single learning stage by composing both supervised and unsupervised objective functions
    - **(2)** - an initial phase of learning from unlabeled data, followed by one or more learning stages during which both labeled and unlabeled data could be used.
  - SSL taxonomy (Yang et al, 2023)
    - generative methods (GANs, VAEs), consistency regularization methods, graph-based methods, pseudo-labeling methods and hybrid methods
    - Generative, graph-based models
  - Deep SSL
    - Vision Transformers **(ViT)** for SS image classification
  - Contrastive learning
    - Form of SSL
    - deep learning technique for unsupervised representation learning
    - learn a representation of data such that
      - similar instances are close together in the representation space, while
      - dissimilar instances are far apart
  - applications
    - search engines (e.g., Google) use SSL to label and rank web pages in their search results
    - image and audio analysis

[SLIDES]

- RL slides (T. Mitchell) [1]

[READING]

- Reinforcement_learning (T. Mitchell) [1]
- Reinforcement_learning (Sutton and Barto) [2]
- Reinforcement_learning (Russel and Norvig) [3]

Bibliography

[1] Mitchell, T., *Machine Learning*, McGraw Hill, 1997 (available at www.cs.ubbcluj.ro/~gabis/ml/ml-books)

[2] Sutton, R.S., Barto, A.G., *Reinforcement learning*, The MIT Press Cambridge, Massachusetts, London, England, 1998 (http://incompleteideas.net/book/the-book.html)

[3] Stuart J. Russell and Peter Norvig, *Artificial Intelligence - A Modern Approach*, Prentice Hall, 1995