Risk-averse Distributional Reinforcement Learning

A CVaR optimization approach

Silvestr Stanko[1]

[1]Department of Computer Science
Czech Technical University

February 7, 2018

Outline

Motivation

Figure: Robotics

Figure: Finance

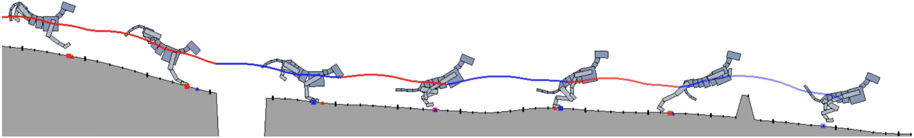


Figure: AI safety

Ultimate goals

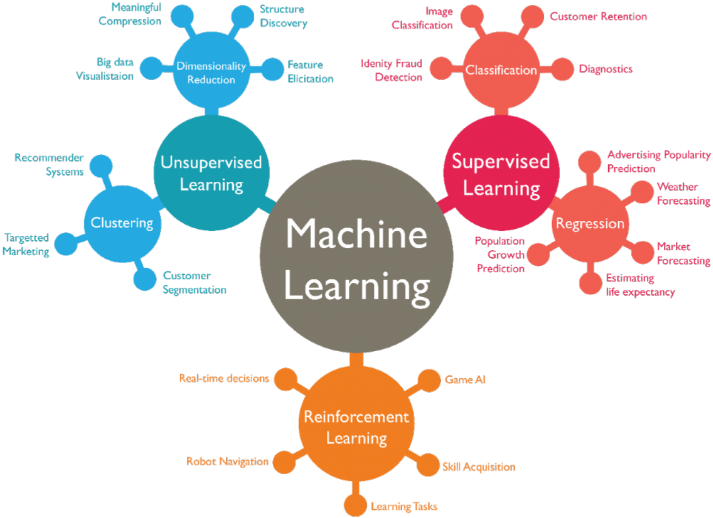
General AI

- Learn from experience
- Learning *tabula rasa*
- Beyond purpose-specific AI
- Better than human-level performance

Safe AI

- Avoiding catastrophic events
- Robust to environment changes or adversaries

Machine Learning



Recent successes

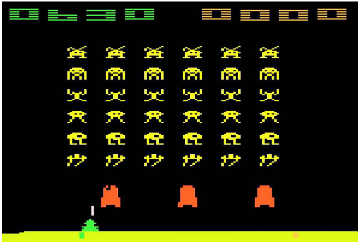


Figure: Atari games

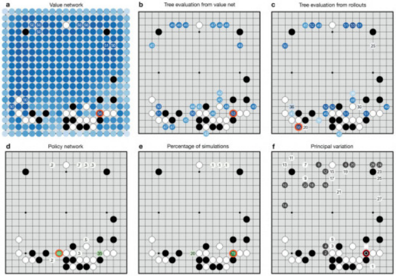
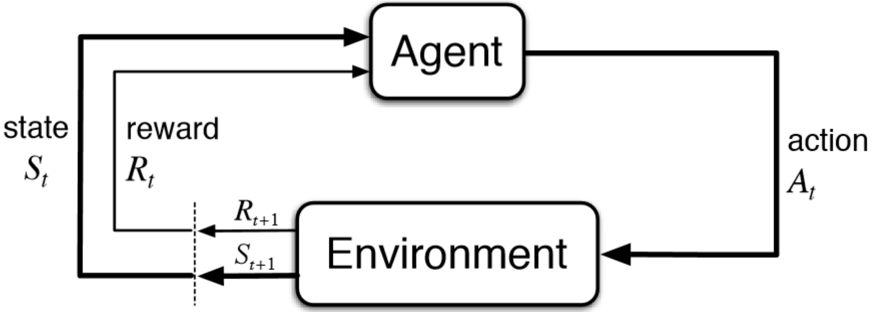


Figure: Go

Reinforcement Learning



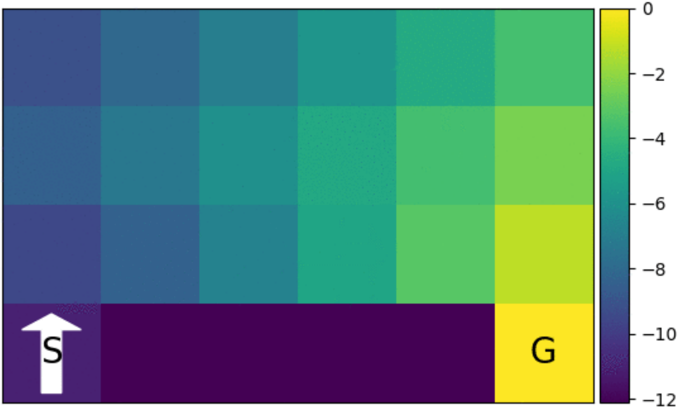
Markov Decision Processes

Definition

An MDP is a 5-tuple $\mathcal{M} = (\mathcal{X}, \mathcal{A}, R, P, \gamma)$, where

- $\mathcal{X}$ is the state space
- $\mathcal{A}$ is the action space
- $R(x, a)$ is a random variable representing the reward generated by being in state x and selecting action a
- $P(\cdot|x, a)$ is the transition probability distribution
- $\gamma \in [0, 1)$ is a discount factor

Markov Decision Process - example



Reinforcement Learning - Goal

Definition

$Z^\pi(x_t)$ Is a random variable representing the discounted reward along a trajectory generated by the MDP by following the policy π, starting at state x_t.

$$Z^\pi(x_t) = \sum_{t=0}^{\infty} \gamma^t R(x_t, \pi(x_t))$$

Reinforcement Learning goals

Our goal is to find a globally optimal policy π^*

$$\pi^* = \arg\max_\pi \mathbb{E} Z^\pi(x_0)$$

Potential problems

- Solutions must avoid catastrophic events and be **safe**
- RL is sample inefficient $\rightarrow$ expensive training
- Solutions must be **robust** to small model changes

Solution

Instead of maximizing the expected reward, focus on other criteria that take into account the **risk** of the potential reward.

Risk

Definition

Risk is the potential of gaining or losing something of value.

Risk-averse: disinclined or reluctant to take risks

Risk-neutral: indifferent to or balanced with respect to risk.

Risk-seeking: inclined or eager to take risks

Example

Choose between recieving:

1. \$100 in 100% cases
2. \$200 in 50% cases and \$0 in 50% cases
3. \$10,000 in 1% cases and \$0 in 99% cases

Measuring Risk

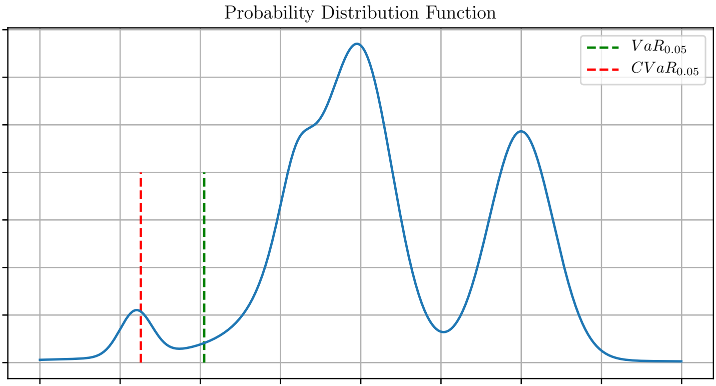
Value-at-Risk (VaR)

- Easy to understand
- Historically the most used risk-measure
- Undesirable computational properties
- Does not differentiate between large and catastrophic losses

Conditional Value-at-Risk (CVaR)

- Coherent risk measure
- Basel Committee on Banking Supervision $\rightarrow$ CVaR
- Equivalent to robustness

Value-at-Risk, Conditional Value-at-Risk



Risk-averse Reinforcement Learning - goals

Definition

$Z^\pi(x_t)$ Is a random variable representing the discounted reward along a trajectory generated by the MDP by following the policy π, starting at state x_t.

$$Z^\pi(x_t) = \sum_{t=0}^{\infty} \gamma^t R(x_t, \pi(x_t))$$

Reinforcement Learning with CVaR

For a given α, our goal is to find a globally optimal policy π^*

$$\pi^* = \arg\max_\pi CVaR_\alpha^\pi(Z^\pi(x_0))$$

Risk-averse Reinforcement Learning - example

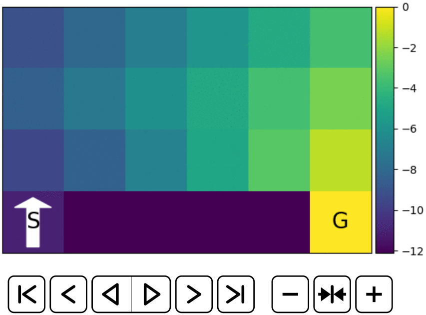


Figure: Greedy agent

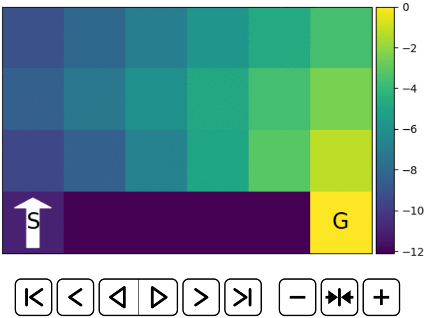


Figure: Risk-averse agent

Value Iteration

Definition

Value function $V(x)$ is the expected return when starting in state x and following the optimal policy π^* thereafter.

Value Iteration

Initialize $V_0(x)$ for each state (arbitrary value, e.g. 0).
Update each state:

$$V_{k+1}(x) = \max_a \left[R(x,a) + \gamma \sum_{x'} p(x'|x,a) V_k(x') \right]$$

Repeat.

The algorithm converges to the optimal policy π^*: $\lim_{k\to\infty} V_k(x) = V(x)$

Value Iteration with CVaR

Theorem (CVaR decomposition)

For any $t \geq 0$, denote by $Z = (Z_{t+1}, Z_{t+2}, \dots)$ the reward sequence from time $t+1$ onward. The conditional CVaR under policy π obeys the following decomposition:

$$CVaR_\alpha\left(Z^\pi(x,a)\right) = \min_{\xi \in \mathcal{U}_{CVaR}(\alpha, P(\cdot|x,a))} \sum_{x'} p(x'|x,a)\xi(x') CVaR_{\xi(x')\alpha}\left(Z^\pi(x')\right)$$

Theorem (CVaR Value Iteration)

The following Bellman operator is a contraction:

$$\mathbf{T}V(x,y) = \max_a \left[R(x,a) + \gamma \min_\xi \sum_{x'} p(x'|x,a)\xi(x') V\left(x', y\xi(x')\right)\right]$$

CVaR Value Iteration

Theorem (CVaR Value Iteration)

The following Bellman operator is a contraction:

$$\mathbf{T}V(x, y) = \max_a \left[R(x, a) + \gamma \min_\xi \sum_{x'} p(x'|x, a)\xi(x')V\left(x', y\xi(x')\right) \right]$$

The operator $\mathbf{T}$ describes the following relationship:

$$\mathbf{T}CVaR_y(Z(x)) = \max_a \left[R(x, a) + \gamma CVaR_y(Z(x, a)) \right]$$

Linear interpolation

Computing operator $\mathbf{T}$ s intractable, as the state-space is continuous. A solution would be to approximate the operator with linear interpolation.

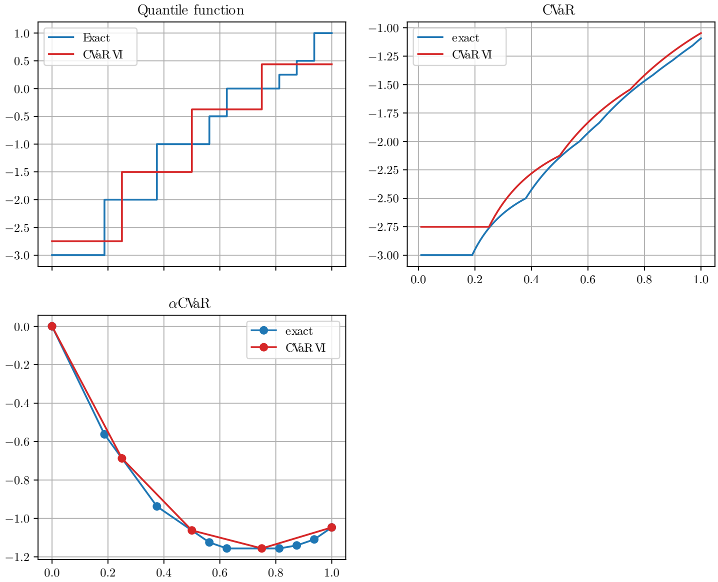
Theorem

The function $\alpha CVaR_\alpha$ is convex. The operator $\mathbf{T}_\mathcal{I}$ is a contraction.

$$\mathcal{I}_x[V](y) = y_i V(x, y_i) + \frac{y_{i+1} V(x, y_{i+1}) - y_i V(x, y_i)}{y_{i+1} - y_i}(y - y_i)$$

$$\mathbf{T}_\mathcal{I} V(x, y) = \max_a \left[R(x, a) + \gamma \min_\xi \sum_{x'} p(x'|x, a) \frac{\mathcal{I}_{x'}[V](y\xi(x'))}{y} \right]$$

This iteration can be formulated and solved as a linear program.


Quantile function
Exact
CVaR VI
CVaR
exact
CVaR VI
αCVaR
exact
CVaR VI

αCVaR$_\alpha$ duality

Lemma

Any discrete distribution has a piece-wise linear $\alpha CVaR_\alpha$ function. Similarly, any a piece-wise linear $\alpha CVaR_\alpha$ function can be seen as representing a certain discrete distribution.

$\alpha\mathrm{CVaR}_\alpha \Leftarrow \mathrm{VaR}$

$$\frac{\partial}{\partial\alpha}\alpha\mathrm{CVaR}_\alpha(Z) = \frac{\partial}{\partial\alpha}\int_0^\alpha VaR_\beta(Z)d\beta = VaR_\alpha(Z)$$

$\alpha\mathrm{CVaR}_\alpha \Rightarrow \mathrm{VaR}$

$$\alpha\mathrm{CVaR}_\alpha(Z) = \int_0^\alpha VaR_\beta(Z)\mathrm{d}\beta$$

Linear-time Computation

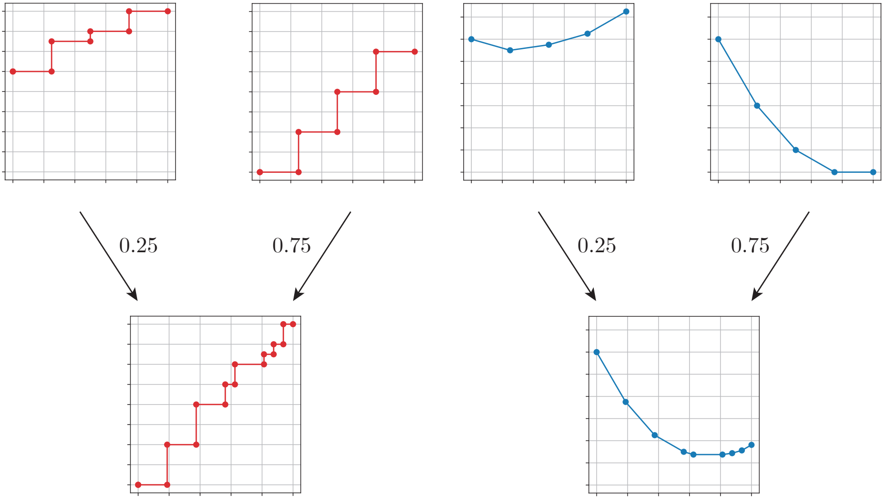
Theorem

Solution to minimization problem present in the CVaR Value Iteration can be computed by setting

$$\xi(x') = \frac{F_{x'}(F_x^{-1}(\alpha))}{\alpha}$$

The computational complexity is $O(n \cdot m)$ where n is the number of transition states and m is the number of atoms.

Next state CVaR computation



VaR-based Policy Improvement

Theorem

Let π be a fixed policy, $\alpha \in (0, 1]$. By following policy π' from the following algorithm, we will improve $CVaR_\alpha(Z)$ in expectation:

$$CVaR_\alpha(Z^\pi) \leq CVaR_\alpha(Z^{\pi'})$$

input α, x_0, γ
$a = \arg\max_a CVaR_\alpha(Z(x_0, a))$
$s = VaR_\alpha(Z(x_0, a))$
$x_t, r_t = \text{envTransition}(x_0, a)$
while x_t is not terminal **do**
  $s = \dfrac{s - r_t}{\gamma}$
  $a = \arg\max_a \mathbb{E}\left[(Z(x_t, a) - s)^-\right]$
  $x_t, r_t = \text{envTransition}(x_t, a)$
end while

TODO

- CVaR Q-learning
  - (?) Use Wasserstein distance with quantile improvement
  - (?) Extend the VaR-based algorithm
  - (?) Combine with quantile regression
- Experiments
  - Value Iteration + Q-learning
  - Deep Q-learning