RL (10/3/2025).

Agent ⇒ Perceive + Act

Environment

Major in RL (schemes).

Policy Iteration

Value Iteration --- Both are in DP setting

Policy Iteration $\xrightarrow{\text{Derivative}}$ GPI (Generalized Policy Iteration) → inexact (approximate).

Value Iteration $\xrightarrow{\text{Derivative}}$ Q^*-learning.

$* \rightarrow$ optimality criteria : g, b, γ, total.

1st optimality criteria → total.

In avg-rew setting, we don't care about Q_g. Thus we just use Q_b

Which is more intuitive?

Remember "no free lunch theorem", ~~so~~ so sometimes PI is ~~more~~ better, so is VI.

Back to inexact policy evaluation. → value.

↳ It uses function approximator (like in supervised learning).

characteristics

Parametric

non-parametric.

↳ has fixed & finite number of params.

classified into:

our focus:

$\boxed{\text{linear}}$

non-linear; e.g. NN.

Notes: NN can be linear too!

What's being approximated: Value

$\hat{V}(s;w) = w^T \cdot f(s) \approx V(s)$ → In matrix form:

$\vec{V}(w) = F \cdot w.$

parameter vector

$w \in W = \mathbb{R}^{\dim(w)}$

state-feature vector function.

$f : S \to \mathbb{R}^n.$

$$\begin{bmatrix} V(s^0;w) \\ V(s^1;w) \\ \vdots \\ \vdots \end{bmatrix} = \begin{bmatrix} f(s^0)^T \\ f(s^1)^T \\ \vdots \\ \vdots \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_{\dim(w)-1} \end{bmatrix}$$

$n = \dim(w)$ in linear setting.

What's next after we define $\hat{v}$?

↳ learn w! → we need to define error function, of course to minimize the error.

(training).

Yields:

optimal $w^* = \underset{w \in \mathbb{R}^d}{\arg\min}\ \underbrace{\overset{\text{error}}{e_x}(w)}_{\text{for this week}}$.

Update via SGD ← e_{ms} (mean square)

e_{PB} (Projected Bellman).

① e_{ms}: weighted (we weight states with $\underbrace{p^{\star}(s)}_{\text{stationary state dist}}$).

we need it bcs some states may not be visited!

$\star \neq *$ (stationary, optimal)

Stationary state dist: likelihood the agent stays in s

$$e_{ms}(w) = \sum_{s \in S} p^{\star}(s)\left[v(s) - \hat{v}(s;w)\right]^2 = \mathbb{E}_{S \sim p^{\star}}\left[v(S) - \hat{v}(S;w)\right]^2$$

We use $p^{\star}(s)$ instead of uniform weighting, i.e multiply by $1/|S|$

For finding exact optimal policy:

~~we need~~

we may need weighted vector-norm:

$$\|\vec{v} - \vec{v}(w)\|^2_{p^*}$$

$$= \left[v - \hat{v}(w)\right]^T \cdot \underline{D_{p^*}} \left[v - \hat{v}(w)\right].$$

↳ A Diagonal Square Matrix.

e.g. $\begin{bmatrix} p^*(s) & 0 & 0 & \cdots \\ 0 & p^*(s) & & \vdots \\ 0 & & \ddots & \\ \vdots & \cdots & & \end{bmatrix}$

During Learning (Training):

↳ update $\hat{w}$, using gradient $\underline{\text{descent}}$.

$$\hat{w} \leftarrow \hat{w} - \alpha \cdot \nabla_{w} e_{ms}$$

hence ~~....~~ we use minus. if ascent → plus.

where

$$\nabla e_{ms} = \nabla \mathbb{E}_{p^{\star}}\left[\left(v(s) - \hat{v}(s;w)\right)^2\right]$$

exchange the order of ∇ vs $\mathbb{E}$

$$= \mathbb{E}_{p^{\star}}\left[\nabla\left(v(s) - \hat{v}(s;w)\right)^2\right].$$

$$= \mathbb{E}_{p^{\star}}\left[2 \cdot \left(v(s) - \hat{v}(s;w)\right) \cdot \left(-\nabla_{w}\hat{v}(s;w)\right)\right]$$

↓ Introduce $\left(-\frac{1}{2}\right)$ as multiplier in both sides!

$$-\frac{1}{2}\nabla e_{ms} = \mathbb{E}_{p^{\star}}\left[\left(\underbrace{v(s)}_{\text{true}} - \underbrace{\hat{v}(s;w)}_{\text{approx.}}\right) \cdot \underbrace{\nabla_{w}\hat{v}(s;w)}_{= f(s) \text{ in linear case!}}\right]$$

c.f. SL, ~~....~~

① there are no training data, so $\boxed{v(s) \text{ is unknown}}$! → thus approximate true value, e.g. with TD, MC, etc.

② in RL, data isn't i.i.d, i.e. Markovian, and collected by the agent itself!

For ①; How to approx. the true value?

$$\approx \mathbb{E}\left[\left\{\underbrace{r(S,A)-\hat{g}+\hat{v}(S';w)}_{\text{approx. } v(s) \text{ based on BEE} \Rightarrow \text{involve } \hat{v}(w) \text{ hence bootstrap}}\right\}-\hat{v}(S;w)\Big\}\cdot f(S)\right]$$

~~we~~ approx operator as we use $g, \hat{v}$

use sampling of (s,a,r',s').

$$\approx \left\{\underbrace{r(s,a)-\hat{g}+\hat{v}(s';w)-\hat{v}(s;w)\Big\}f(s)}_{\text{sampling approximation.}}\right\}$$

term

↳ bootstrap = using estimation ~~to esti~~ of a value to estimate another one.

RL - Week 6 (12-3-2025).

Implementations in RL:

1.) SL: approx policy evaluation — caveats: ① not i.i.d. ② true val unknown

2.) UL: Clustering states (aggregating, e.g Jabodetabek)

- State representation / feature extraction.

Continuing on error: Projected Bellman Error

- In contrast with ℓ_{ms}, ℓ_{PB} doesn't involve true value $\rightarrow$ more stable.

Bellman Operator.

From BEE: correlating value of current with next state.

$$\underbrace{v(s) = r(s,a) - g + \mathbb{E}[v(s')]}$$

TD = RHS − LHS = 0 ... theoritically...

TEMPOral Difference, i.e. (t+1) − (t).

matrix (vector) form:

$$\vec{v} = \vec{r} - g\mathbb{1} + P\vec{v} \iff (I-P)v = r - g\mathbb{1}$$

singular!

P: 1-step transition matrix

FAKULTAS ILMU KOMPUTER

$\vec{v} = \mathbb{B}[\vec{v}]$

↳ Bellman (evaluator) operator. on $\vec{v}$

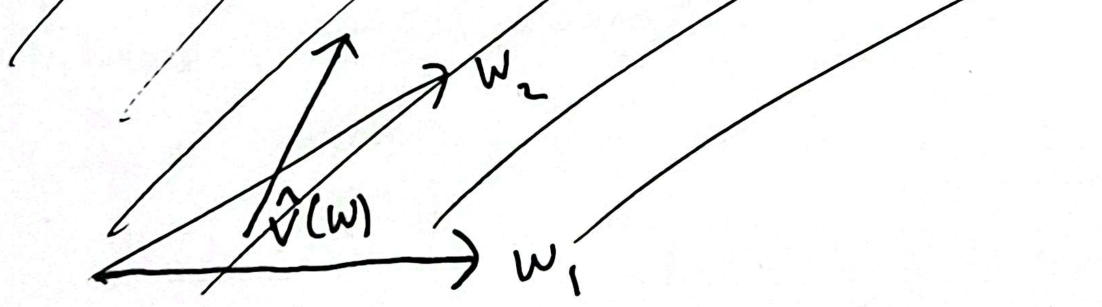
$$e_{PB} = \| \underbrace{\hat{v}(w) - \underbrace{\mathbb{P}}_{\text{Projection operator.}} \cdot \mathbb{B}\, \hat{v}(w)}_{\Delta V} \|^2_{p^\star} \quad \text{--- norm vector.}$$

$$= (\Delta V)^T \cdot D_{p^\star} \cdot \Delta V$$

$$= \sum_{s \in S} p^\star(s) \cdot (\Delta V(s))^2$$

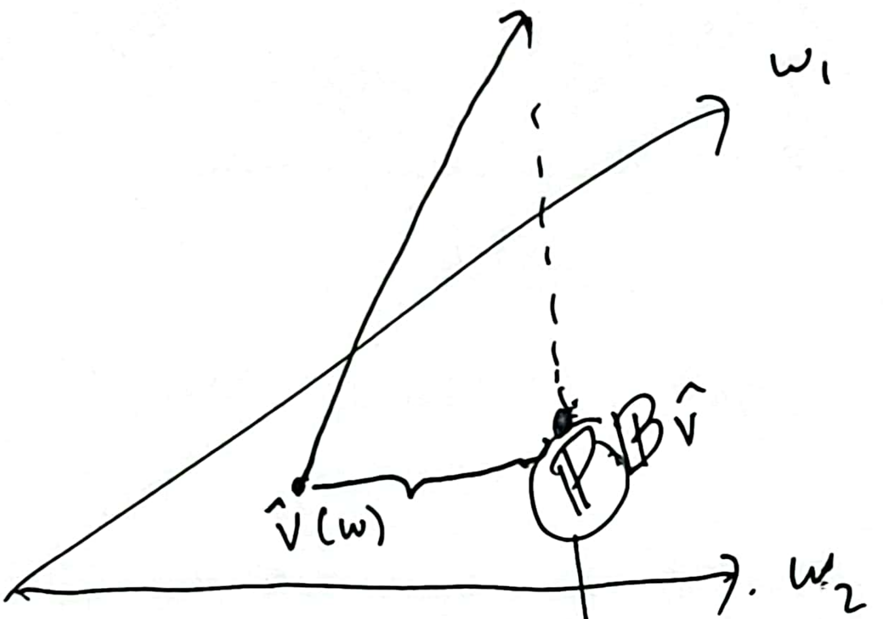
$$= \mathbb{E}_{p^\star}\left[(\Delta V(s))^{\cancel{2}}\right]^2$$

Let $\vec{w} = [w_1 \ w_2]^T$, $w_1, w_2 \in \mathbb{R}$.



Let $w = [w_1 \; w_2]^T ; w_1, w_2 \in \mathbb{R}$.

$\mathbb{B}[\hat{v}(w)] \to$ is not in parameter space! $\to$ params w can't represent $\mathbb{B}[\hat{v}_w]$



to bring back $\hat{v}(w)$ after $\mathbb{B}[\hat{v}(w)]$!

The projection operator $\mathbb{P}$ satisfies:

$$\mathbb{P}(v) = F \tilde{w}$$

true val? feature matrix.

later on will be cancel out!

where:

$$\tilde{w} = \underset{w \in W}{\arg\min} \left\{ \| \underbrace{F \cdot w}_{\hat{v}} - v \|^2_{p^\star} \right\}.$$

~~later on, the~~

we can show that:

$$\boxed{\mathbb{P} = F(F^T D_{p^\star} F)^{-1} F^T D_{p^\star}}$$

Now, we should be able to get:

$$\boxed{w^{\star} = \underset{w}{\operatorname{argmin}}\ \ell_{PB}(w)}$$

How?

① Take grad of ℓ_{PB}

② Set the grad to 0 as in local minima, grad is 0.

We can find $w^{\star}$ with closed form solution

$w^{\star} = X^{-1} \cdot y$ → Note: in DP setting.

where

$$X = \sum_s p^{\star}(s) \cdot \sum_{s'} p(s'|s) \left[f(s) \{ f(s) - f(s') \}^T \right]$$

$$= \sum p^{\star}(s) f(s) \left[\left(f(s) - \sum_{s'} p(s'|s) f(s') \right)^T \right]$$

$$= F^T D_{p^{\star}} (I - P) \cdot F$$

FAKULTAS ILMU KOMPUTER

and

$$y = \sum_s p^{\star}(s) \left[(r(s) - g\mathbb{1})^T \cdot f(s) \right]$$

$$= F^T D_{p^{\star}} (r - g\mathbb{1})$$

In RL settings, we can't find w^* using $X^{-1}y$ as before. Thus, we'll use LSTD, which is the approximation version of $X^{-1}y$.

Least Square Temporal Diff.

$$\boxed{\hat{w}^* = \hat{X}^{-1}\hat{y}}$$

available because X and y are sampling-friendly, e.g. using expectation, i.e ~~P^*~~ weight with p^*,

$$\hat{X} = \frac{1}{n}\sum_{i=1}^{n} f(s_i)\left[f(s_i) - f(s_i')\right]^T$$

$$\hat{y} = \frac{1}{n}\sum \left(r(s_i) - g\right) \cdot f(s_i).$$

Note:
n is number of samples. ~~used~~

Based on sample mean,

$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

which is unbiased!

Problem with LSTD:

$\hat{w} = \hat{X}^{-1} \hat{y}$, but sometimes $\hat{X}$ is not invertible.

Hence introduce ε to add into it!

$$\hat{X} \leftarrow \hat{X} + \varepsilon \mathbf{I}.$$

Are there training & testing in RL?

In RL, there are no separation between training vs testing!

~~Training~~: before convergence / after convergence.

If we really want to distinguish: Basic setting: In the same env

~~Even the environment for traing~~

~~But~~ maybe not in more challenging setting,

e.g. ① Non-stationary env

② Different but similar env.

Is tabular setting a special case of parametric function approximator?

Yes, where $f(s)$ uses one-hot encoding and $\dim(w) = |S|$ e.g.,

	$s = s^0$?	$s = s^1$?	...
$f(s^0)$	1	0	0 ...
$f(s^1)$	0	1	0 ...