Towards a unified view of supervised learning and reinforcement learning

Mohammad Norouzi

Goolge Brain

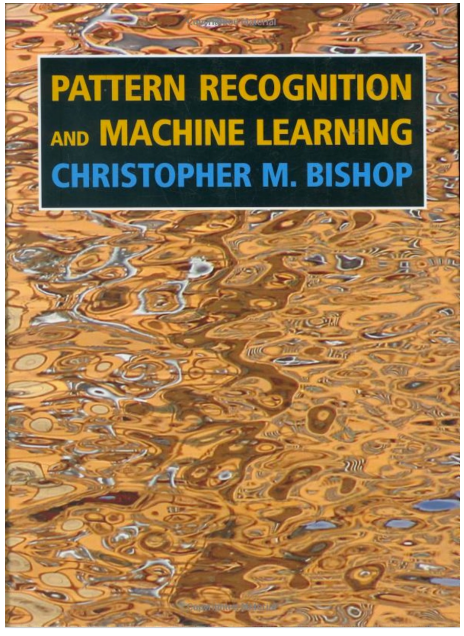
Joint work with Ofir Nachum and Dale Schuurmans

April 19, 2017

Based on three joint papers with various subsets of

Ofir Nachum, Dale Schuurmans,
Kelvin Xu, Samy Bengio, Yonghui Wu,
Zhifeng Chen, Navdeep Jaitly, Mike Schuster

- RML: Reward Augmented Maximum Likelihood for Neural Structured Prediction, NIPS 2016
- UREX: Improving Policy Gradient by Exploring Under-appreciated Rewards, ICLR 2016
- PCL: Bridging the Gap Between Value and Policy Based Reinforcement Learning, Preprint 2017


PATTERN RECOGNITION
AND MACHINE LEARNING
CHRISTOPHER M. BISHOP


Reinforcement
Learning
An Introduction
Richard S. Sutton and Andrew G. Barto

Supervised learning *v.s.* reinforcement learning

- What is the *biggest* difference?
  - The problem domains are different, but they both learn to map input/states to output/actions
  - To me, the *key* difference is the *objective* function

Supervised learning *v.s.* reinforcement learning

- Supervised learning (this talk)
  - Structured output prediction
  - Optimizing conditional log-likelihood
- Reinforcement Learning (this talk)
  - Discrete actions, deterministic transition, finite horizon
  - Optimizing expected reward (**+entropy**)
- The *"entropy of the policy"* is key to connect the dots.

Image captioning: supervised learning *v.s.* reinforcement learning

Learn a mapping from an image $\mathbf{x}$ to a sequence of words $\mathbf{a}$

$\mathbf{x}$ $\qquad$ $\mathbf{a} \equiv [a_1, \ldots, a_T]$

$\longrightarrow$

A dog lying next to a cute cat on a white bed.

- Supervised learning: lots of input-ouput pairs are available.
- Reinforcement learning: a bunch of raters provide $r(\mathbf{a} \mid \mathbf{x})$.
- Real world is a *hybrid* of the two paradigms!

Roadmap

- Learn a mapping ($\mathbf{x} \rightarrow \mathbf{a}$)
  from inputs ($\mathbf{x} \equiv [x_1, \ldots, x_T]$)
  to output/actions ($\mathbf{a} \equiv [a_1, \ldots, a_T]$)
  to maximize a reward $r(\mathbf{a} \mid \mathbf{x})$.
- Model/policy $\pi_\theta(\mathbf{a} \mid \mathbf{x}) = \prod_i \pi_\theta(a_i \mid \mathbf{x}_{<i}, \mathbf{a}_{<i})$
- For brevity, let's drop the conditioning on $\mathbf{x}$.

Roadmap

- Learn a mapping ($\mathbf{x} \to \mathbf{a}$)
  from inputs ($\mathbf{x} \equiv [x_1, \ldots, x_T]$)
  to output/actions ($\mathbf{a} \equiv [a_1, \ldots, a_T]$)
  to maximize a reward $r(\mathbf{a})$.
- Model/policy $\pi_\theta(\mathbf{a}) = \prod_i \pi_\theta(a_i \mid \mathbf{a}_{<i})$
- Optimal policy $\pi^*(\mathbf{a})$ is given by $\pi^*(\mathbf{a}) = \mathbb{1}[\mathbf{a} = \operatorname{argmax}_a r(\mathbf{a})]$
- We introduce a *soft optimal policy* $\pi^*_\tau(\mathbf{a})$:

$$\pi^*_\tau(\mathbf{a}) = \frac{1}{Z} \exp(r(\mathbf{a}) / \tau)$$

  - I am going to color π^*_τ blue in the rest of the talk
  - Our goal is to find the blue guy so $\pi_\theta \approx \pi^*_\tau$, but **how**?
  - $D_{\mathrm{KL}}\left(\pi^*_\tau \,\middle\|\, \pi_\theta\right) \approx$ conditional log-likelihood at $\tau = 0$
  - $D_{\mathrm{KL}}\left(\pi_\theta \,\middle\|\, \pi^*_\tau\right) \approx$ expected reward at $\tau = 0$

Roadmap

- We introduce $\pi^*_\tau(\mathbf{a})$: *soft optimal policy*

$$\pi^*_\tau(\mathbf{a}) = \frac{1}{Z} \exp(r(\mathbf{a}) / \tau)$$

  - $D_{\mathrm{KL}}(\pi^*_\tau \parallel \pi_\theta) \approx$ conditional log-likelihood at $\tau = 0$
  - $D_{\mathrm{KL}}(\pi_\theta \parallel \pi^*_\tau) \approx$ expected reward at $\tau = 0$

- Study the non-zero temperature
  - MENT [Peng & Willimas]: expected reward $+ \tau \times$ entropy
  - (1) RAML: conditional log-likelihood with $\tau > 0$

- (2) UREX: combining the two directions of KL to benefit from *mode seeking* $D_{\mathrm{KL}}(\pi_\theta \parallel \pi^*_\tau)$ & *mode covering* $D_{\mathrm{KL}}(\pi^*_\tau \parallel \pi_\theta)$

- (3) Softmax Bellman operator: entropy-regularized expected reward with partial rewards (bridge value & policy based RL)

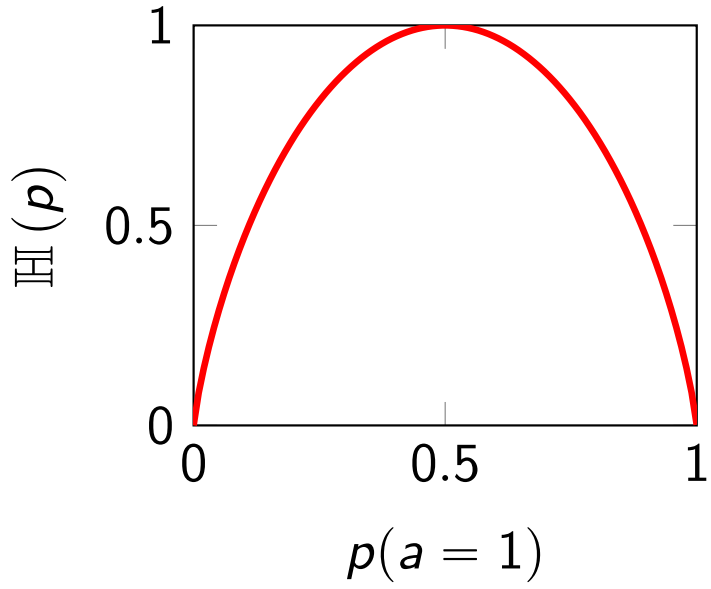
(0) Background

Entropy

Measures uncertainty/information content of a distribution $p(\mathbf{a})$

$$\begin{aligned}\mathbb{H}(p) &= -\sum_{\mathbf{a}\in\mathcal{A}} p(\mathbf{a}) \log p(\mathbf{a}) \\ &= \mathbb{E}_{\mathbf{a}\sim p}[-\log p(\mathbf{a})]\end{aligned}$$

- Concave on the simplex
- Maximum of $\log|\mathcal{A}|$ at uniform.
- Minimum of 0 at one-hot.



KL divergence or relative entropy

- KL divergence between distributions $p(\mathbf{a})$ and $q(\mathbf{a})$,

$$\begin{aligned} D_{\mathrm{KL}}\big(p \parallel q\big) &= \sum_{\mathbf{a}\in\mathcal{A}} p(\mathbf{a})[\log p(\mathbf{a}) - \log q(\mathbf{a})] \\ &= \mathbb{E}_{\mathbf{a}\sim p}[\log p(\mathbf{a}) - \log q(\mathbf{a})] \end{aligned}$$

- D_{KL} is nonnegative, asymmetric, zero iff $p = q$

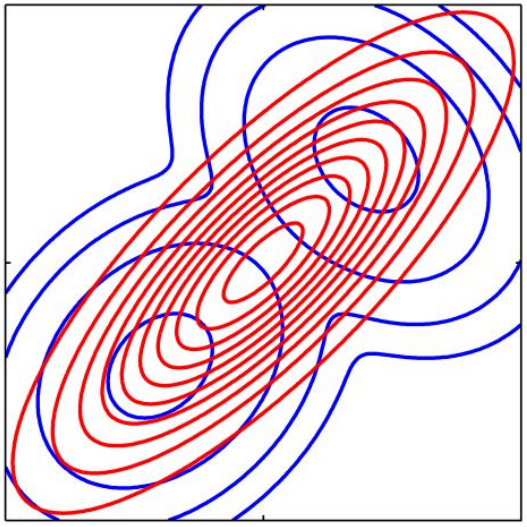
$$-D_{\mathrm{KL}}\big(p \parallel q\big) = \mathbb{H}\left(p\right) + \mathbb{E}_{\mathbf{a}\sim p}[\log q(\mathbf{a})]$$

- $-D_{\mathrm{KL}}\big(p \parallel q'/Z\big) = \mathbb{H}\left(p\right) + \mathbb{E}_{\mathbf{a}\sim p}[\log q'(\mathbf{a})] - \log Z$

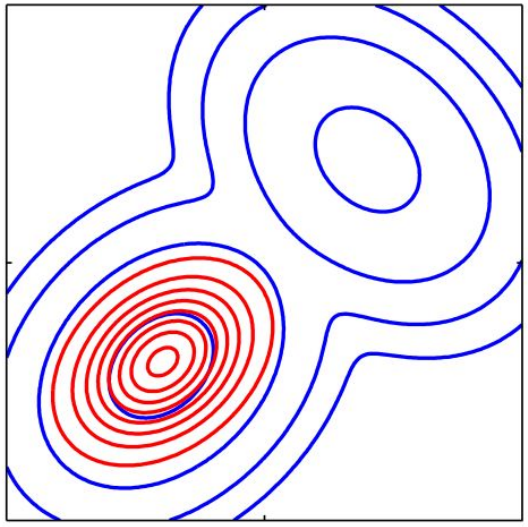
KL divergence – mode seeking *v.s.* mode covering

$$-D_{\mathrm{KL}}(p \parallel q) = \mathbb{H}(p) + \mathbb{E}_{\mathbf{a}\sim p}[\log q(\mathbf{a})]$$

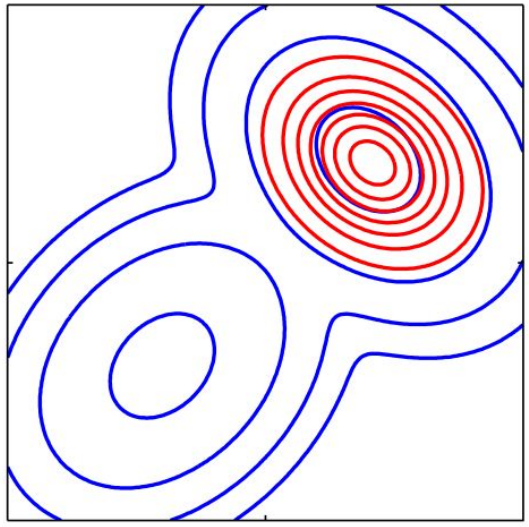
[Image courtesy of Bishop's book.]



(a) $D_{\mathrm{KL}}(p \parallel q)$
mode seeking



(c) $D_{\mathrm{KL}}(p \parallel q)$
mode covering



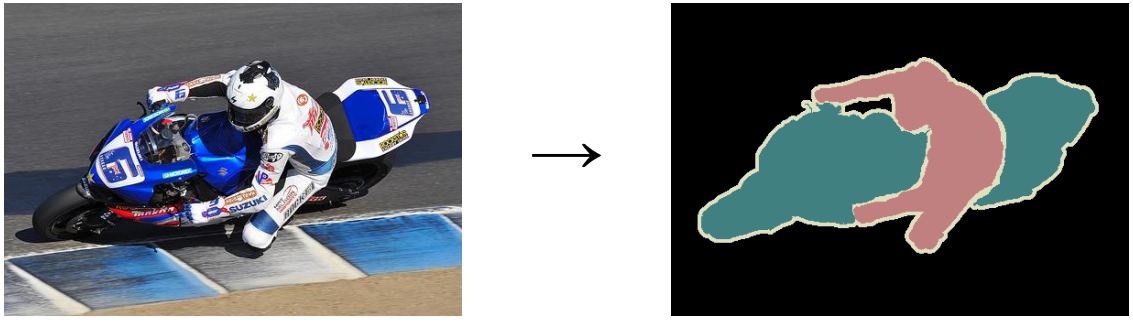
(b) $D_{\mathrm{KL}}(p \parallel q)$
mode covering

(1) **RAML**: **R**eward **A**ugmented **M**aximum **L**ikelihood

Learning from *good* mistakes.

Structured output prediction

- Image segmentation



Structured output prediction

- Machine translation

As diets change, people get bigger but plane seating has not radically changed.

$\longrightarrow$

Avec les changements dans les habitudes alimentaires, les gens grossissent, mais les sièges dans les avions n'ont pas radicalement changé.

Structured output prediction

- Image captioning

$\longrightarrow$

A dog lying next to a cute cat on a white bed.

Structured output prediction

Example tasks:

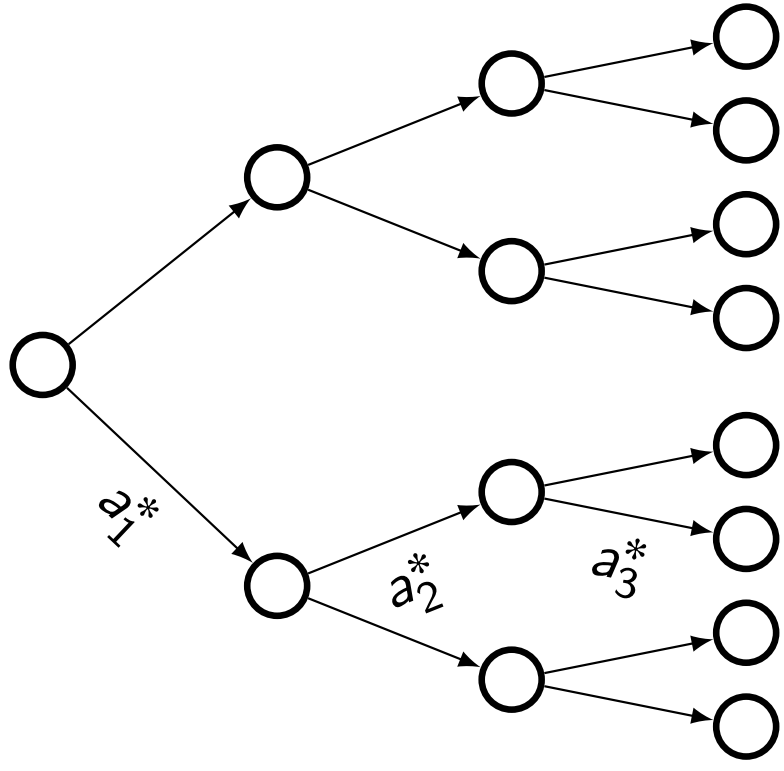
- Image (semantic) segmentation
- Machine translation
- Image captioning

Characteristics:

- Usually lots of input-output pairs.
- Outputs comprise multi-variate correlated (discrete) variables.
- Given a complete output, a reward signal is computed:
  - Intersection over Union
  - BLEU and ROUG scores
  - Human evaluation

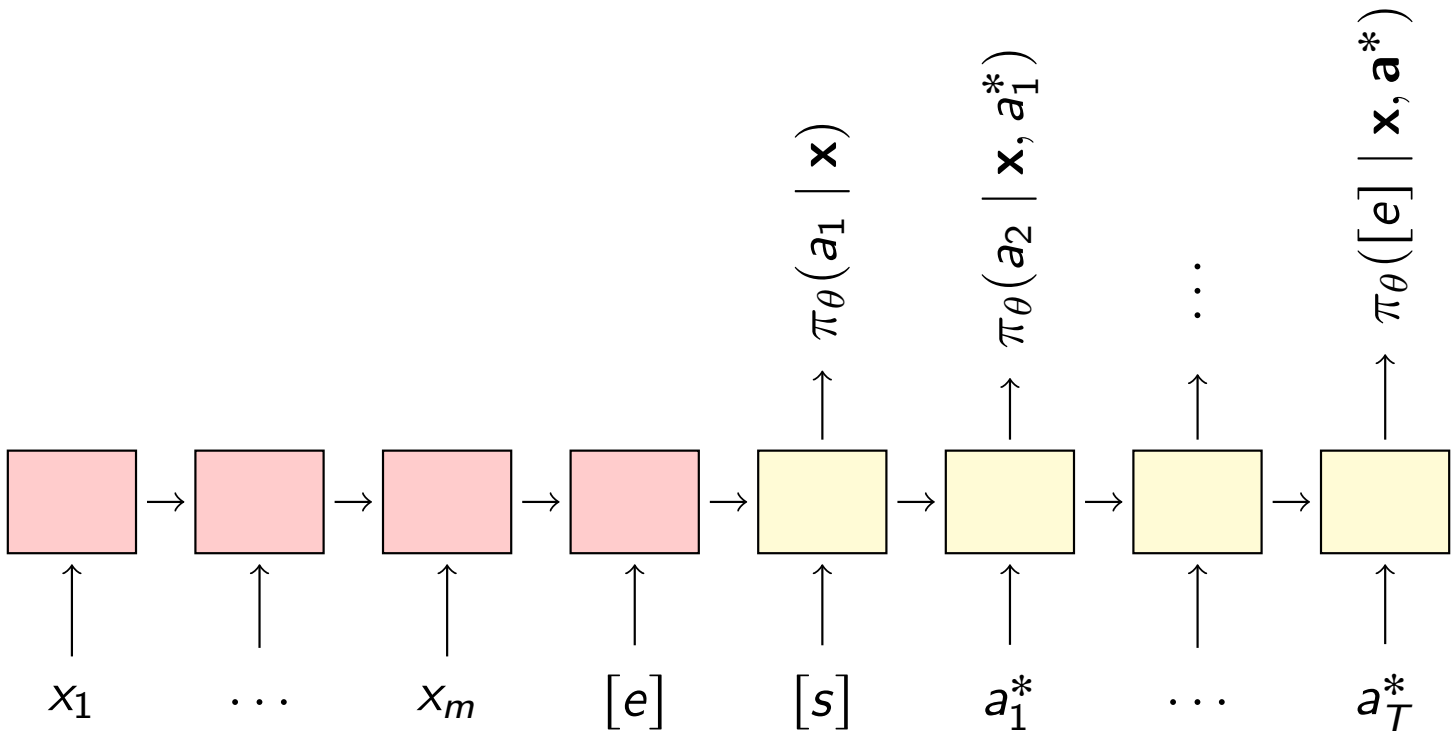
Structured prediction: dominant appraoch nowadays

- Output = sequence of decisions, $\mathbf{a} \equiv [a_1, \ldots, a_T]$
- Define $\pi_\theta(\mathbf{a} \mid \mathbf{x}) = \prod_i \pi_\theta(a_i \mid \mathbf{x}, \mathbf{a}_{<i})$
- Ignore rewards and maximize $\sum_{(\mathbf{x},\mathbf{a}^*)\in\mathcal{D}} \log \pi_\theta(\mathbf{a}^* \mid \mathbf{x})$



Structured prediction: dominant appraoch nowadays

- At inference, beam search finds $\widehat{\mathbf{a}}(\mathbf{x}) \approx \operatorname{argmax}_{\mathbf{a}} \pi_\theta(\mathbf{a} \mid \mathbf{x})$
- Prediction quality is measured by $\sum_{(\mathbf{x},\mathbf{a}^*)} r(\widehat{\mathbf{a}}(\mathbf{x}), \mathbf{a}^*)$



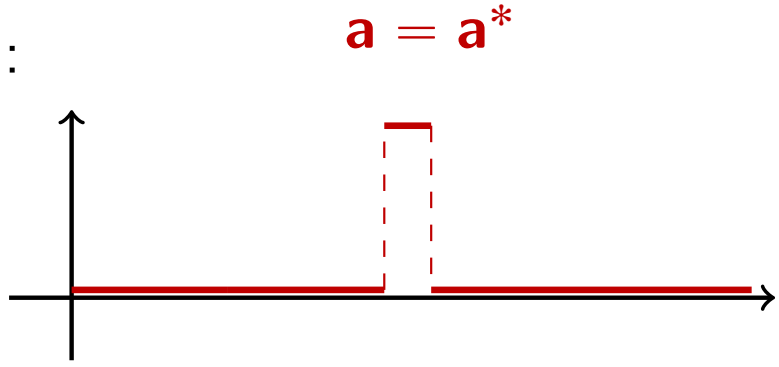
Conditional log-likelihood for a single example

- Drop the conditioning of $\mathbf{a}$ on $\mathbf{x}$ for brevity.

$$\begin{aligned} \mathcal{O}_{\mathrm{CLL}}(\boldsymbol{\theta}) &= \log \pi_\theta(\mathbf{a}^*) \\ &= -D_{\mathrm{KL}}\big(\mathbb{1}[\mathbf{a} = \mathbf{a}^*] \,\big\|\, \pi_\theta(\mathbf{a})\big) \end{aligned}$$

- There is no notion of reward (*e.g.* BLEU score, edit distance).
- All of the negative outputs $\mathbf{a} \neq \mathbf{a}^*$ are equally penalized.

Optimal $\pi^*(\mathbf{a})$:



Expected reward for a single example

Expected reward (+entropy)

$$\mathcal{O}_{\mathrm{MENT}}(\boldsymbol{\theta}, \tau) \quad = \quad \mathbb{E}_{\mathbf{a}\sim\pi_\theta}[r(\mathbf{a})/\tau] \; + \; \mathbb{H}\left(\pi_\theta\right)$$

- To optimize $\mathcal{O}_{\mathrm{MENT}}$, one uses REINFORCE to compute $\nabla_{\boldsymbol{\theta}}\mathcal{O}_{\mathrm{MENT}}$ by sampling from $\pi_\theta(\mathbf{a})$, *e.g.* [*Ranzato* et al.] .
- The gradients are high variance. The training is slow.
- This approach ignores supervision (after initialization).
- One needs to bootstrap training from an CLL-trained model.

Key observation

Recall the soft optimal policy

$$\pi^*_\tau(\mathbf{a}) \;=\; \frac{1}{Z}\exp\left\{r(\mathbf{a})\,/\,\tau\right\}$$

One can re-express $\mathcal{O}_{\mathrm{MENT}}$ as:

$$\begin{aligned}
\mathcal{O}_{\mathrm{MENT}}(\boldsymbol{\theta},\tau) &= \mathbb{E}_{\mathbf{a}\sim\pi_\theta}[r(\mathbf{a})/\tau] \;+\; \mathbb{H}\left(\pi_\theta\right) \\
&= \mathbb{E}_{\mathbf{a}\sim\pi_\theta}[r(\mathbf{a})/\tau - \log Z] + \log Z \;+\; \mathbb{H}\left(\pi_\theta\right) \\
&= \mathbb{E}_{\mathbf{a}\sim\pi_\theta}[\log \pi^*_\tau(\mathbf{a})] + \mathbb{H}\left(\pi_\theta\right) \;+\; \log Z \\
&= -D_{\mathrm{KL}}\left(\pi_\theta \;\|\; \pi^*_\tau\right) + \log Z
\end{aligned}$$

The soft optimal policy π^*_τ is a global maximum $\mathcal{O}_{\mathrm{MENT}}$.

RAML

How about optimizing for π_τ^* more directly by $D_{\mathrm{KL}}\big(\pi_\tau^* \,\big\|\, \pi_\theta\big)$?

$$\begin{aligned}\mathcal{O}_{\mathrm{RAML}}(\boldsymbol{\theta}, \tau) &= \mathbb{E}_{\mathbf{a}\sim\pi_\tau^*} \log \pi_\theta(\mathbf{a} \mid \mathbf{x}) \\ &= -D_{\mathrm{KL}}\big(\pi_\tau^*(\mathbf{a}) \,\big\|\, \pi_\theta(\mathbf{a})\big) - \mathbb{H}\left(\pi_\tau^*\right)\end{aligned}$$

- Similar to CLL, in the direction of KL.
- Similar to MENT, in the optimal policy.
- A notion of reward is captured in $\pi_\tau^*(\mathbf{a}) \propto \exp\{r(\mathbf{a}, \mathbf{a}^*)/\tau\}$.
- The temperature τ controls the concentration of π_τ^*.
  As $\tau \to 0$, then $\pi_\tau^*(\mathbf{a}) \to \mathbb{1}[\mathbf{a} = \mathbf{a}^*]$ and $\mathbb{H}\left(\pi_\tau^*\right) \to 0$.
- $\mathcal{O}_{\mathrm{RAML}}$ is convex in $\boldsymbol{\theta}$ for log-linear models.

RAML optimization

Training with RAML is efficient and easy to implement.

- Given a training pair $(\mathbf{x}^{(i)}, \mathbf{a}^{*(i)})$, first sample $\widetilde{\mathbf{a}} \sim \pi^*_\tau(\mathbf{a} \mid \mathbf{a}^{*(i)})$, then optimize $\log \pi_\theta(\widetilde{\mathbf{a}} \mid \mathbf{x}^{(i)})$.

$$\nabla_{\boldsymbol{\theta}} \mathcal{O}_{\mathrm{RAML}}(\boldsymbol{\theta}, \tau) \;=\; \sum_{(\mathbf{x}, \mathbf{a}^*)} E_{\widetilde{\mathbf{a}} \sim \pi^*_\tau(\mathbf{a} \mid \mathbf{a}^{*(i)})} \big[\nabla_{\boldsymbol{\theta}} \log \pi_\theta(\widetilde{\mathbf{a}} \mid \mathbf{x}) \big] \,.$$

- We just sample one augmentation $\widetilde{\mathbf{a}}$ per input $\mathbf{x}$ per iteration.
- A form of data augmentation based on target rewards.
- No bootstrapping needed. Much harder to over-fit than CLL.
- By contrast, REINFORCE ($\tau = 0$) samples from π_θ:

$$\nabla_{\boldsymbol{\theta}} \mathcal{O}_{\mathrm{ER}}(\boldsymbol{\theta}) \;=\; \sum_{(\mathbf{x}, \mathbf{a}^*)} E_{\mathbf{a} \sim \pi_\theta(\mathbf{a} \mid \mathbf{x})} \big[\nabla_{\boldsymbol{\theta}} \log \pi_\theta(\mathbf{a} \mid \mathbf{x}) \cdot r(\mathbf{a}, \mathbf{a}^*) \big]$$

Sampling from soft optimal policy

Stratified sampling: first select a particular reward value, then sample a target with that reward.

- If reward $=$ – Hamming distance, $r(\mathbf{a}, \mathbf{a}^*) = -D_{\mathrm{H}}(\mathbf{a}, \mathbf{a}^*)$, one can draw exact samples from $\pi^*_\tau(\mathbf{a} \mid \mathbf{a}^*)$.

  $$\text{if} \quad \mathcal{A} \equiv \{1, \ldots, v\}^m, \qquad \text{then} \quad r(\mathbf{a}, \mathbf{a}^*) \in \{0, \ldots, -m\}$$

  It is easy to count $\{\mathbf{a} \in \mathcal{A} \mid r(\mathbf{a}, \mathbf{a}^*) = k\}$: $\binom{m}{k}(v-1)^k$.
  Summing over k, one can compute the normalization factor.

- For edit distance, an approximate sampler is proposed.

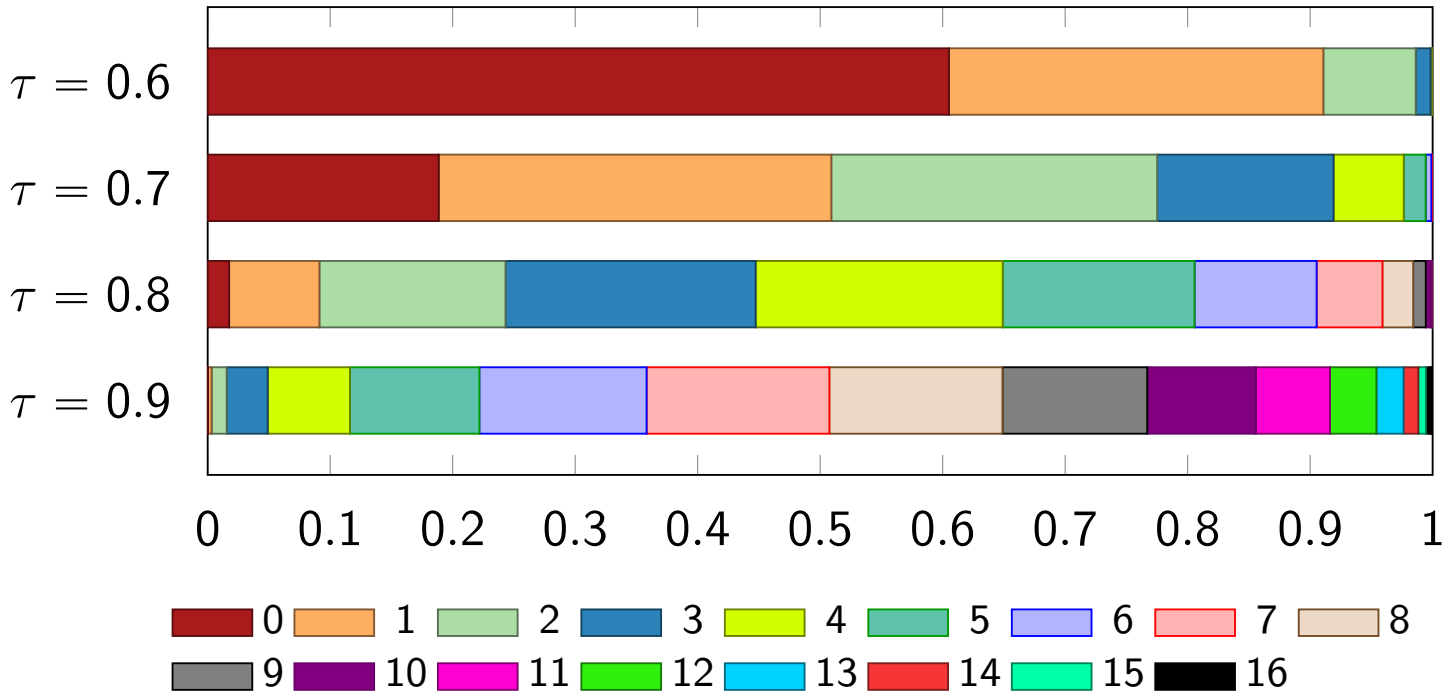
Generally, one can resort to MCMC and importance sampling.
Samples from $\pi^*_\tau(\mathbf{a} \mid \mathbf{a}^*)$ can be pre-computed and stored.

TIMIT Speech Recognition

Phone error rates (PER) for different methods on TIMIT dev & test sets. Average (min, max) PER for 4 training runs are reported.

Method	Dev set	Test set
CLL baseline	20.87 (−0.2, +0.3)	22.18 (−0.4, +0.2)
RAML, $\tau = 0.60$	19.92 (−0.6, +0.3)	21.65 (−0.5, +0.4)
RAML, $\tau = 0.65$	19.64 (−0.2, +0.5)	21.28 (−0.6, +0.4)
RAML, $\tau = 0.70$	18.97 (−0.1, +0.1)	21.28 (−0.5, +0.4)
RAML, $\tau = 0.75$	18.44 (−0.4, +0.4)	20.15 (−0.4, +0.4)
RAML, $\tau = 0.80$	18.27 (−0.2, +0.1)	19.97 (−0.1, +0.2)
RAML, $\tau = 0.85$	18.10 (−0.4, +0.3)	19.97 (−0.3, +0.2)
RAML, $\tau = 0.90$	**18.00** (−0.4, +0.3)	**19.89** (−0.4, +0.7)
RAML, $\tau = 0.95$	18.46 (−0.1, +0.1)	20.12 (−0.2, +0.1)
RAML, $\tau = 1.00$	18.78 (−0.6, +0.8)	20.41 (−0.2, +0.5)

Fraction of number of edits for a sequence of length 20



At $\tau = 0.9$, augmentations with 5 to 9 edits are sampled with a probability > 0.1.

Machine Translation (WMT WMT'14 En→Fr)

Tokenized BLEU score

Method	Average BLEU	Best BLEU
ML baseline	36.50	36.87
RAML, $\tau = 0.75$	36.62	36.91
RAML, $\tau = 0.80$	36.80	37.11
RAML, $\boldsymbol{\tau = 0.85}$	**36.91**	**37.23**
RAML, $\tau = 0.90$	36.69	37.07
RAML, $\tau = 0.95$	36.57	36.94

RAML at different τ considerably outperforms CLL.

Related work

- ◇ [*Szegedy* et al., CVPR'16] Rethinking the Inception
  Label smoothing is a special case of our method
- ◇ [Volkovs, Larochelle, Zemel, ArXiv'11] Loss-sensitive Training of Probabilistic Conditional Random Fields, applies the same ideas to CRF for ranking. No connection to RL.

Alternative methods requiring sampling or inference at training:

- ◇ [*S. Bengio* et al., NIPS'15] Schedule sampling
- ◇ [*Ranzato* et al., ICLR'16] Sequence level training
- ◇ [*Wiseman & Rush*, EMNLP'16] Beam search optimization

How about RL

Is RAML applicable to RL with unknown reward landscapes?

(2) **UREX**: **U**nde-appreciated **R**eward **EX**ploration

Calibrate rewards/τ with log-plicies

Motivation

- Common forms of exploration in RL are undirected.
- We need more effective exploration in high-dimensional action spaces with sparse delayed reward.

Key idea

- Recall soft optimal policy $\pi_\tau^*(\mathbf{a}) \propto \exp(r(\mathbf{a})/\tau)$
- Augment the expected reward objective with $D_{\mathrm{KL}}\left(\pi_\tau^* \,\|\, \pi_\theta\right)$ to encourage *mode covering* behavior

$$\mathcal{O}_{\mathrm{UREX}}(\boldsymbol{\theta}, \tau) \;=\; \mathbb{E}_{\mathbf{a}\sim\pi_\theta}[r(\mathbf{a})/\tau] \;+\; \mathbb{E}_{\mathbf{a}\sim\pi_\tau^*}[\log \pi_\theta(\mathbf{a})]$$

- To sample $\mathbf{a} \sim \pi_\tau^*$, first draw $\mathbf{a} \sim \pi_\theta$, then reweight by π_τ^*/π_θ. (importance sampling)
- let's start by reviewing the baselines

REINFORCE [Williams'87]

Standard policy-based approach to maximize expected reward:

$$\begin{aligned} \mathcal{O}_{\mathrm{ER}}(\boldsymbol{\theta}) &= \mathbb{E}_{\mathbf{a}\sim\pi_\theta}[r(\mathbf{a})] \\ \nabla\mathcal{O}_{\mathrm{ER}}(\boldsymbol{\theta}) &= \mathbb{E}_{\mathbf{a}\sim\pi_\theta}[r(\mathbf{a})\nabla\log\pi_\theta(\mathbf{a})] \end{aligned}$$

- Draw K *i.i.d.* action sequence samples: $\mathbf{a}^{(k)} \sim \pi_\theta(\mathbf{a})$ for each $1 \leqslant k \leqslant K$
- Estimate the gradient via

$$\nabla\mathcal{O}_{\mathrm{ER}} = \frac{1}{K}\sum_{k=1}^{K}(r(\mathbf{a}^{(k)}) - b)\,\nabla\log\pi_\theta(\mathbf{a}^{(k)})$$

- Use a baseline b to reduce variance, *e.g.* sample mean reward.
- This fails even on simple problems due to lack of *exploration*.

MENT [Peng & Williams'91]

Augment the objective with entropy regularization

$$\begin{aligned}\mathcal{O}_{\mathrm{MENT}}(\boldsymbol{\theta}, \tau) &= \mathbb{E}_{\mathbf{a}\sim\pi_\theta}[r(\mathbf{a})/\tau] + \mathbb{H}(\pi_\theta) \\ \nabla\mathcal{O}_{\mathrm{MENT}}(\boldsymbol{\theta}, \tau) &= \mathbb{E}_{\mathbf{a}\sim\pi_\theta}[(r(\mathbf{a})/\tau - \log\pi_\theta(\mathbf{a}) - 1)\,\nabla\log\pi_\theta(\mathbf{a})]\end{aligned}$$

- Estimate the gradient using K on-policy samples

$$\nabla\mathcal{O}_{\mathrm{MENT}} = \frac{1}{K}\sum_{k=1}^{K}(r(\mathbf{a}^{(k)})/\tau \underbrace{-\log\pi_\theta(\mathbf{a}^{(k)})}_{\text{entropy bonus}} - b)\,\nabla\log\pi_\theta(\mathbf{a}^{(k)})$$

- MENT does better that REINFORCE, but still fails on problems with a large action space.
- We need something better!

UREX [Nachum *et al.*, ICLR'16]

Augment the objective with *mode covering* KL

$$\mathcal{O}_{\mathrm{UREX}}(\boldsymbol{\theta}, \tau) \quad = \quad \mathbb{E}_{\mathbf{a}\sim\pi_\theta}[r(\mathbf{a})/\tau] \; + \; \mathbb{E}_{\mathbf{a}\sim\pi^*_\tau}[\log \pi_\theta(\mathbf{a})]$$

- Draw K *i.i.d.* on-policy samples $\{\mathbf{a}^{(k)}\}_{k=1}^K$ from $\pi_\theta(\mathbf{a})$.
- Compute self-normalized importance weights

$$w^{(k)} = \exp\left\{r(\mathbf{a}^{(k)})/\tau - \log \pi_\theta(\mathbf{a}^{(k)})\right\} , \qquad \widetilde{w}^{(k)} = \frac{w^{(k)}}{\sum_{i=1}^K w^{(i)}} .$$

- Estimate the gradient as

$$\nabla \mathcal{O}_{\mathrm{UREX}} = \sum_{k=1}^K \Big[\frac{1}{K}(r(\mathbf{a}^{(k)})/\tau - b) + \underbrace{\widetilde{w}^{(k)}}_{\text{UREX bonus}}\Big] \nabla \log \pi_\theta(\mathbf{a}^{(k)})$$

- The most *under-appreciated* action sequence has largest $\widetilde{w}^{(k)}$.

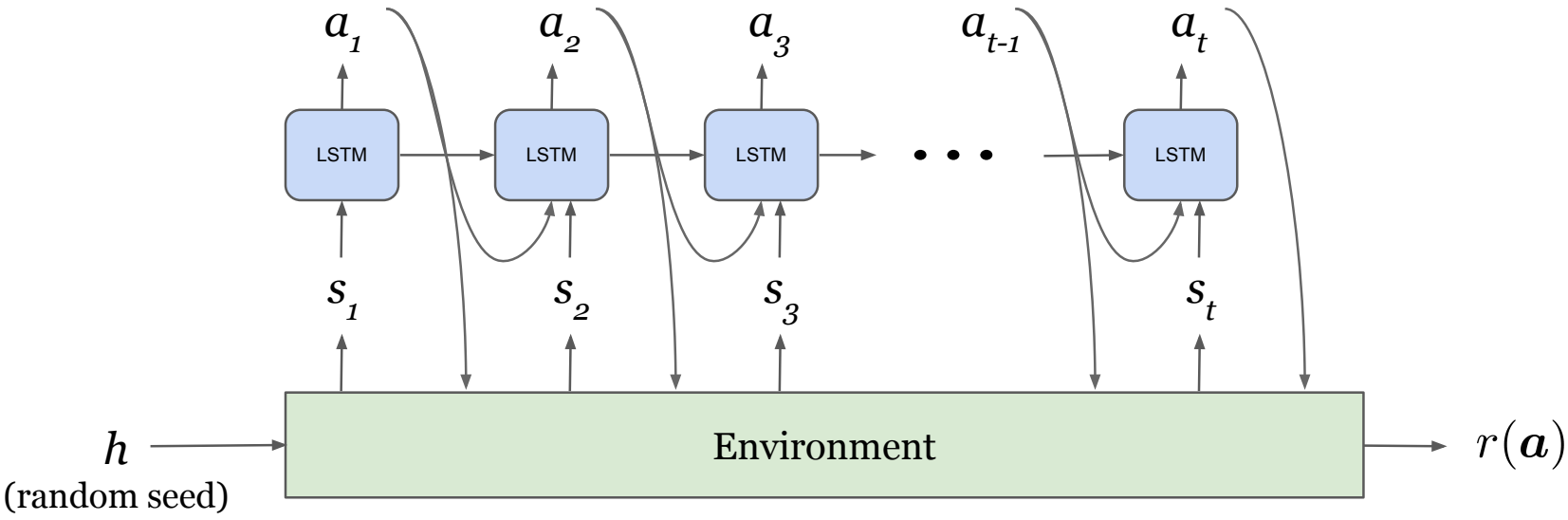
Characteristics of UREX

- UREX encourages exploration in areas where rewards are under estimated by the current log-policy, *i.e.* $w^{(k)}$ measures the difference between $r(\mathbf{a}^{(k)})/\tau$ and $\log \pi_\theta(\mathbf{a}^{(k)})$.

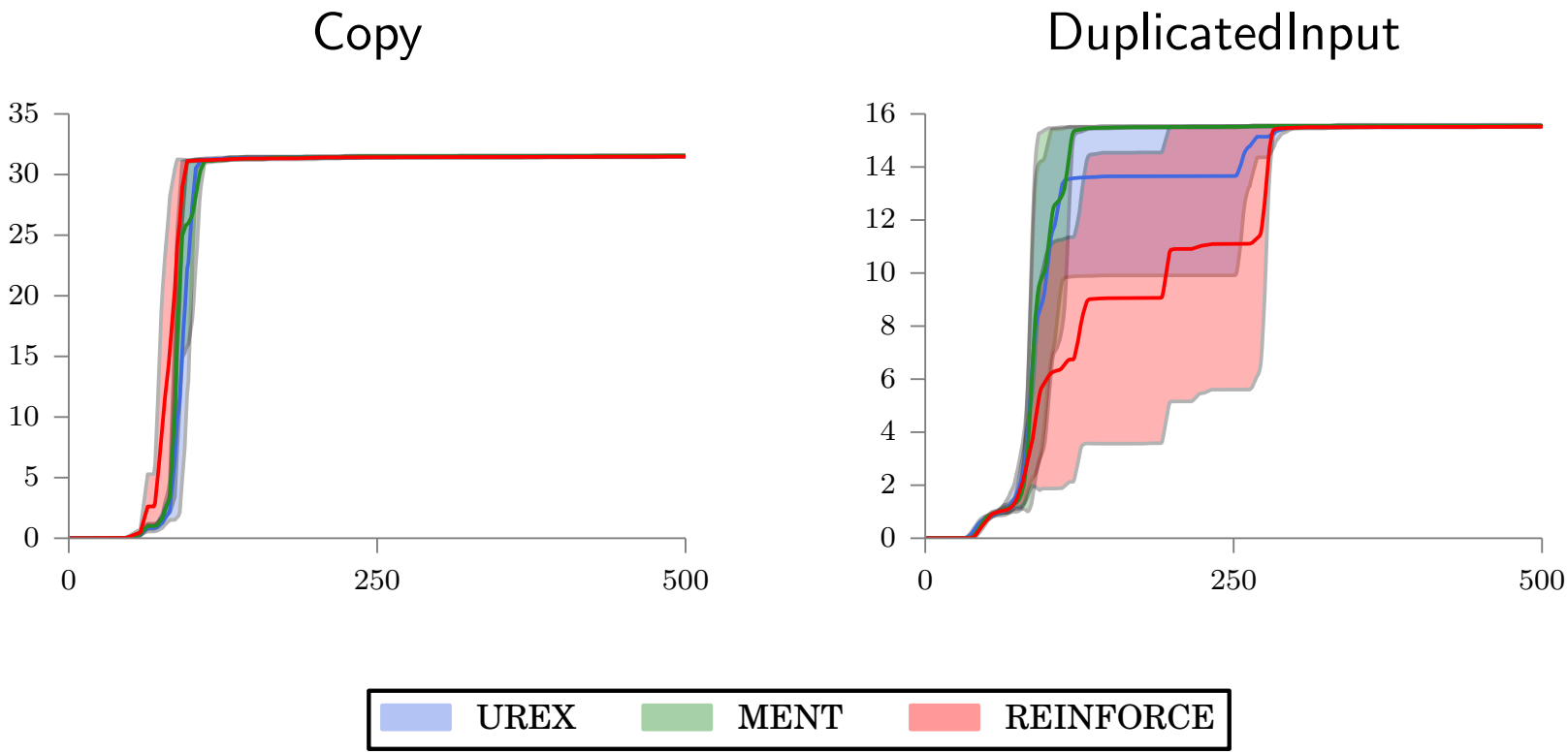
$$w^{(k)} = \exp\left\{r(\mathbf{a}^{(k)})/\tau - \log \pi_\theta(\mathbf{a}^{(k)})\right\} , \qquad \widetilde{w}^{(k)} = \frac{w^{(k)}}{\sum_{i=1}^{K} w^{(i)}}$$

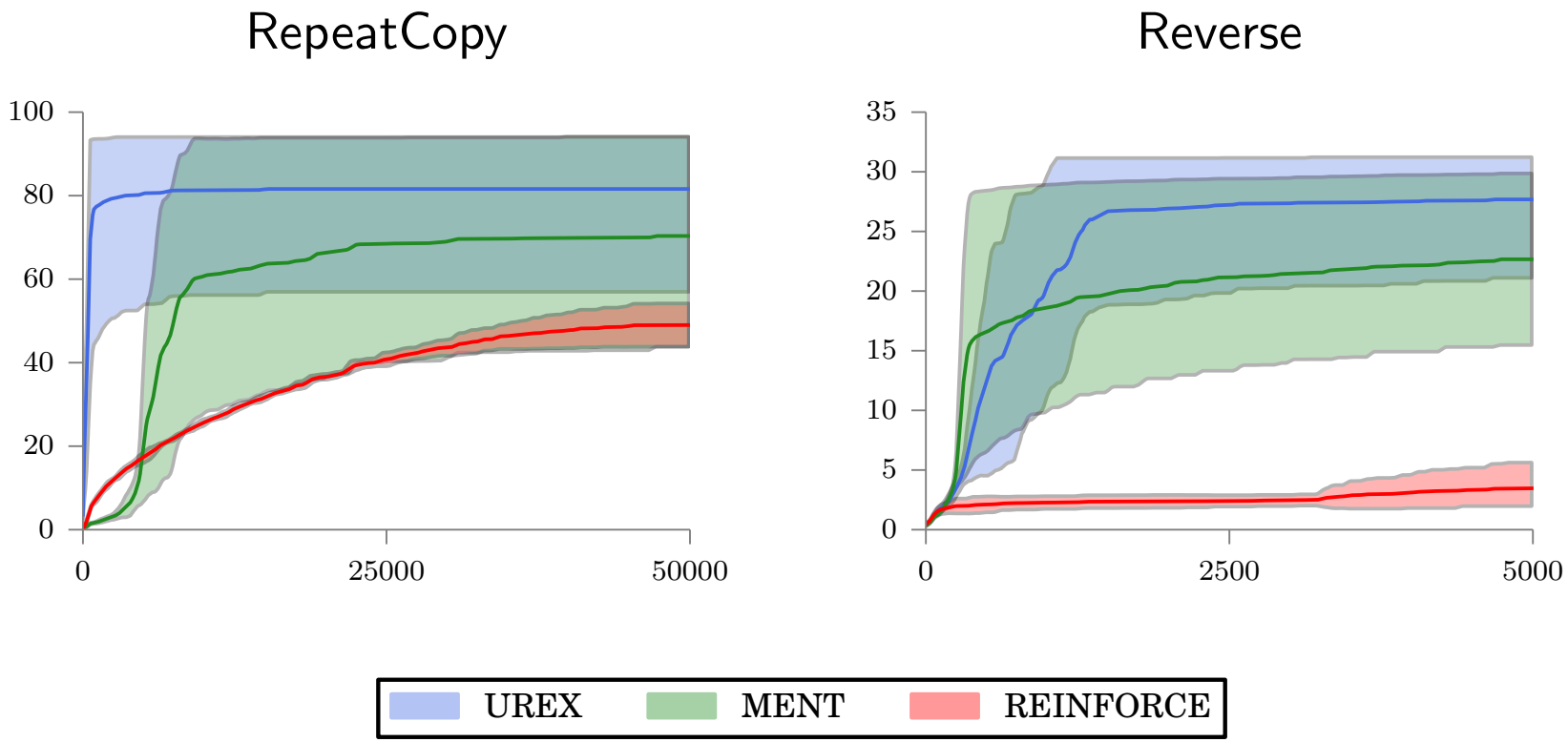
- The most **under-appreciated** action sequences among K samples has the largest UREX bonus $\widetilde{w}^{(k)}$.
- One needs multiple samples to normalize importance weights.
- Simple and easy to implement.

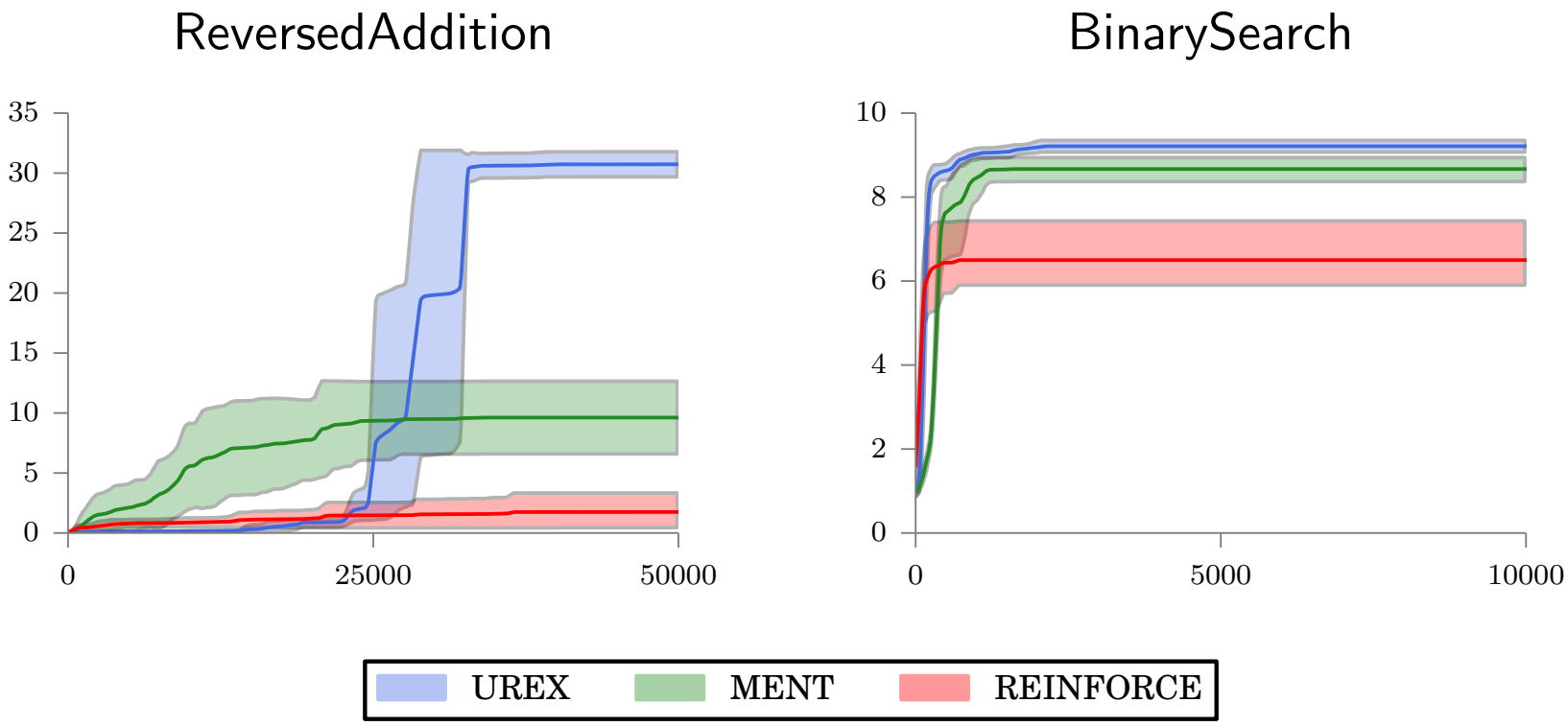
RNN policy



$$\mathbf{a} = [a_1, a_2, \ldots, a_t] \qquad \pi_\theta(\mathbf{a} \mid \mathbf{s}, h) = \prod_{i=1}^{t} \pi_\theta(a_i \mid \mathbf{a}_{<i}, \mathbf{s}_{<i}, h)$$


Copy
DuplicatedInput
35
30
25
20
15
10
5
0
16
14
12
10
8
6
4
2
0
0
250
500
UREX
MENT
REINFORCE


RepeatCopy
Reverse
100
80
60
40
20
0
0
25000
50000
35
30
25
20
15
10
5
0
0
2500
5000
UREX
MENT
REINFORCE


ReversedAddition
BinarySearch
35
30
25
20
15
10
5
0
0
25000
50000
10
8
6
4
2
0
0
5000
10000
UREX
MENT
REINFORCE

Hyper-parameters

- Learning rate $\eta \in \{0.1, 0.01, 0.001\}$
- Gradient clipping L2 norm $c \in \{1, 10, 40, 100\}$
- Temp. $\tau \in \{0, 0.005, 0.01, 0.1\}$, always $\tau = 0.1$ for UREX

	REINFORCE / MENT				UREX
	$\tau = 0.0$	$\tau = 0.005$	$\tau = 0.01$	$\tau = 0.1$	$\tau = 0.1$
Copy	85.0	88.3	**90.0**	3.3	75.0
DuplicatedInput	68.3	73.3	73.3	0.0	**100.0**
RepeatCopy	0.0	0.0	11.6	0.0	**18.3**
Reverse	0.0	0.0	3.3	10.0	**16.6**
ReversedAddition	0.0	0.0	1.6	0.0	**30.0**
BinarySearch	0.0	0.0	1.6	0.0	**20.0**

Results

- The RL agents only observe total reward at the end of episode.
- UREX reliably solves reversion and multi-digit addition.
- UREX $\geqslant$ MENT $\geqslant$ REINFORCE.

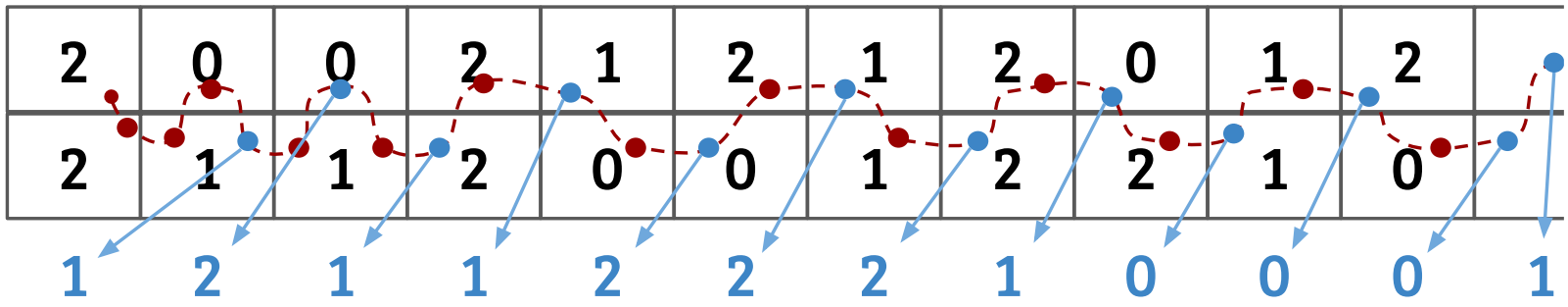
	Expected reward		
	REINFORCE	MENT	UREX
Copy	31.2	31.2	31.2
DuplicatedInput	15.4	15.4	15.4
RepeatCopy	48.7	69.2	**81.1**
Reverse	3.0	21.9	**27.2**
ReversedAddition	1.4	8.7	**30.2**
BinarySearch	6.4	8.6	**9.1**

Results

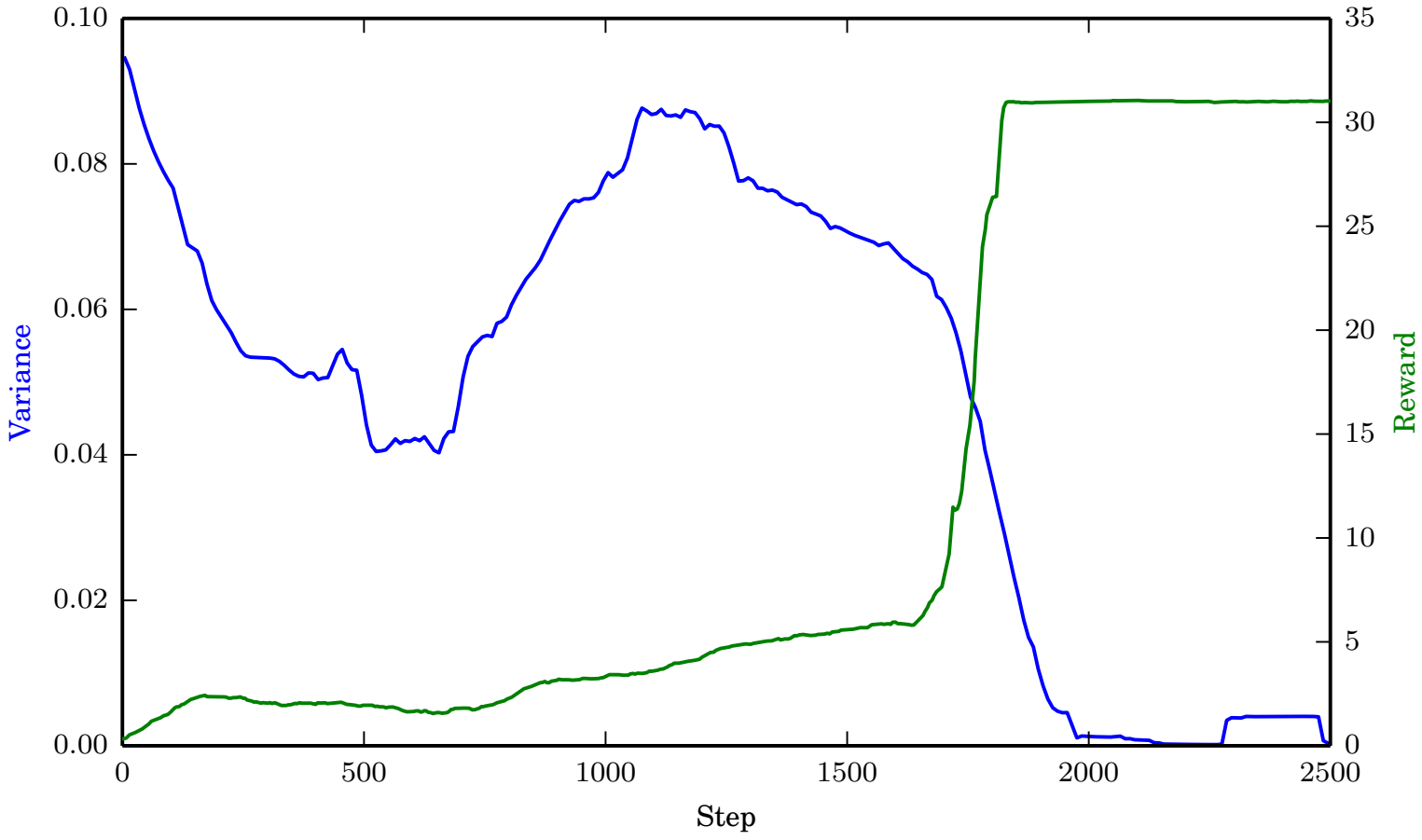
- The RL agents only observe total reward at the end of episode.
- UREX reliably solves reversion and multi-digit addition.
- UREX $\geqslant$ MENT $\geqslant$ REINFORCE.

	Num. of successful attempts out of 5		
	REINFORCE	MENT	UREX
Copy	5	5	5
DuplicatedInput	5	5	5
RepeatCopy	0	3	**4**
Reverse	0	2	**4**
ReversedAddition	0	1	**5**
BinarySearch	0	1	**4**

ReversedAddition execuation trace



Variance of importance weights *v.s.* expected reward



(3) Bridging the gap between value and policy based RL (quick overview)

Entropy is the bridge again!

Induction

- Suppose we are at state s_0
  facing n possible actions $\{a_1, \ldots, a_n\}$
  with immediate rewards of $\{r_1, \ldots, r_n\}$
  successor states of $\{s_1, \ldots, s_n\}$
  with sate values $\{v_1, \ldots, v_n\}$.
- Induce the current state value v_0

Expected reward

$$O_{\mathrm{ER}}(\pi) \;=\; \sum_{i=1}^{n} \pi(a_i)(r_i + \gamma v_i^{\circ})$$

- Suppose we are at state s_0
  facing n possible actions $\{a_1, \ldots, a_n\}$
  with immediate rewards of $\{r_1, \ldots, r_n\}$
  successor states of $\{s_1, \ldots, s_n\}$
  with O_{ER}-optimal sate values $\{v_1^{\circ}, \ldots, v_n^{\circ}\}$.
- $\pi^{\circ}(a) = \mathbb{1}[a = a_{i*}]$ where $i^* = \mathrm{argmax}_i(r_i + \gamma v_i^{\circ})$
- Induce the current state value v_0

$$v_0^{\circ} \;=\; O_{\mathrm{ER}}(\pi^{\circ}) \;=\; \max_i (r_i + \gamma v_i^{\circ}).$$

Entropy regularized expected reward

$$O_{\text{MENT}}(\pi, \tau) = \sum_{i=1}^{n} \pi(a_i)(r_i + \gamma v_i^* - \tau \log \pi(a_i)),$$

- Suppose we are at state s_0
  facing n possible actions $\{a_1, \ldots, a_n\}$
  with immediate rewards of $\{r_1, \ldots, r_n\}$
  successor states of $\{s_1, \ldots, s_n\}$
  with O_{MENT}-optimal sate values $\{v_1^*, \ldots, v_n^*\}$.
- $\pi^*(a_i) \propto \exp\{(r_i + \gamma v_i^*)/\tau\}$
- Induce the current state value v_0

$$v_0^* = O_{\text{MENT}}(\pi^*, \tau) = \tau \log \sum_{i=1}^{n} \exp\{(r_i + \gamma v_i^*)/\tau\}$$

Softmax Bellman operator

$$v_0^* = O_{\text{MENT}}(\pi^*, \tau) = \tau \log \sum_{i=1}^{n} \exp\{(r_i + \gamma v_i^*)/\tau\}$$

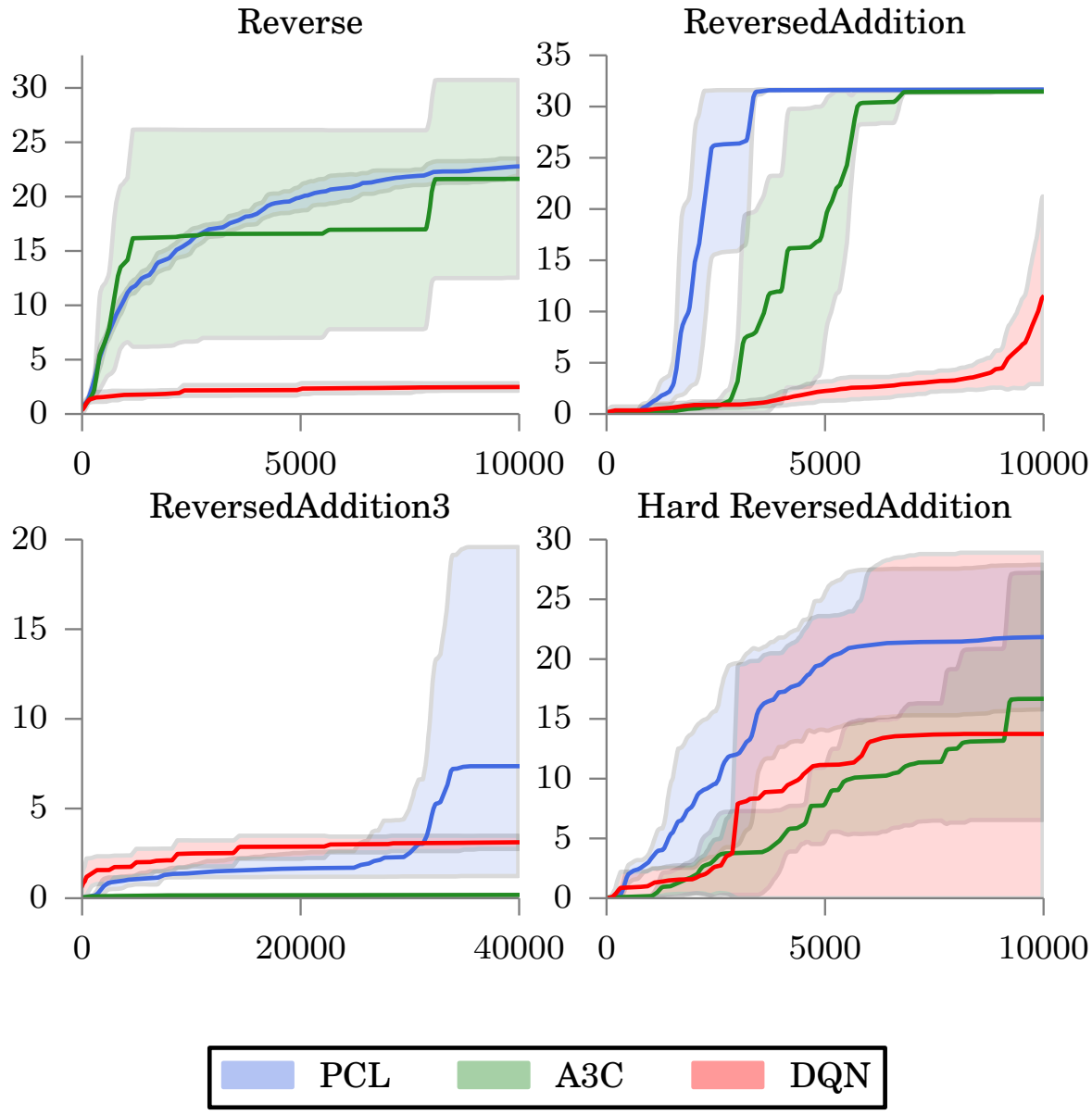
$$\pi^*(a_i) = \frac{\exp\{(r_i + \gamma v_i^*)/\tau\}}{\exp\{v_0^*/\tau\}}$$

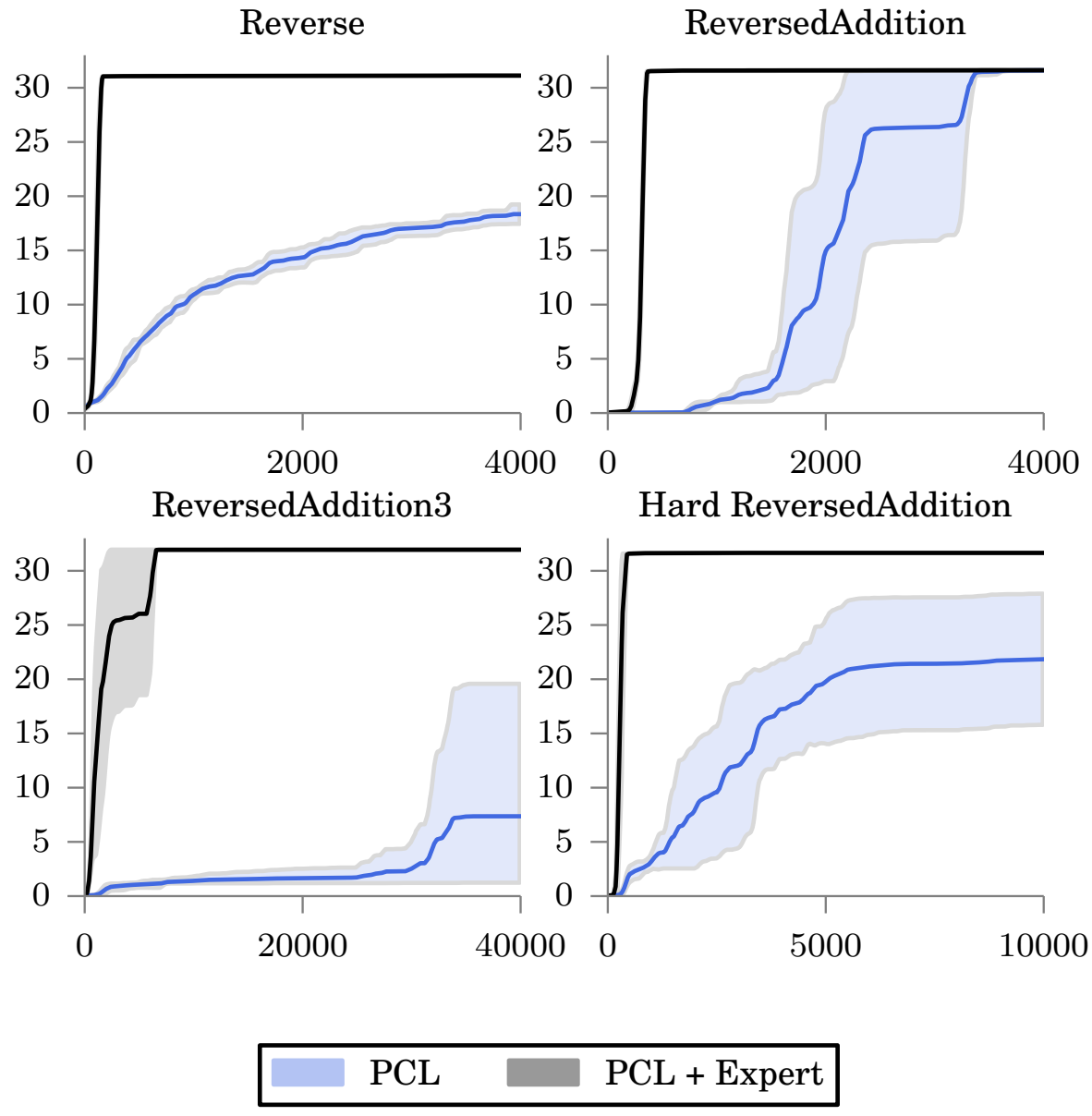
Softmax temporal consistency applicable to on-policy and off-policy samples

$$v_0^* = r_i + \gamma v_i^* - \tau \log \pi^*(a_i)$$

We propse path-consistency learning (PCL) to minimize

$$\sum_{0,i} (v_0 - r_i + \gamma v_i - \tau \log \pi(a_i))^2$$


Reverse
ReversedAddition
ReversedAddition3
Hard ReversedAddition
0
5
10
15
20
25
30
35
0
5000
10000
20000
40000
PCL
A3C
DQN


Reverse
ReversedAddition
ReversedAddition3
Hard ReversedAddition
30
25
20
15
10
5
0
0
2000
4000
20000
40000
5000
10000
PCL
PCL + Expert

Summary

- We introduce $\pi^*_\tau(\mathbf{a})$: *soft optimal policy*

$$\pi^*_\tau(\mathbf{a}) = \frac{1}{Z} \exp(r(\mathbf{a}) / \tau)$$

  - $D_{\mathrm{KL}}(\pi^*_\tau \parallel \pi_\theta) \approx$ conditional log-likelihood at $\tau = 0$
  - $D_{\mathrm{KL}}(\pi_\theta \parallel \pi^*_\tau) \approx$ expected reward at $\tau = 0$

- Study the non-zero temperature
  - MENT [Peng & Willimas]: expected reward $+ \tau \times$ entropy
  - (1) RAML: conditional log-likelihood with $\tau > 0$

- (2) UREX: combining the two directions of KL to benefit from *mode seeking* $D_{\mathrm{KL}}(\pi_\theta \parallel \pi^*_\tau)$ & *mode covering* $D_{\mathrm{KL}}(\pi^*_\tau \parallel \pi_\theta)$

- (3) Softmax Bellman operator: entropy-regularized expected reward with partial rewards (bridge value & policy based RL)

Future directions

- Continuous control.
- UREX for decomposable rewards.
- Incorporating trust region methods.
- Study the connection with simulated annealing.
- Exploit off-policy samples & expert trajectories more.

More question?

(4) Thank you!