

- today
- learning to search
  - SEARNN
  - SPENS

## Learning to search (L2S)

SEARNN Hal Daumé's phd thesis

$$h_w : X \rightarrow \mathcal{Y}$$

$$h_w(x) = \underset{y \in \mathcal{Y}}{\operatorname{argmax}} s(x, y; w) \quad ] \text{ parameterized search procedure}$$

special case of SEARNN: learning to do greedy search.

split  $y$  in ordered list of decisions

$$(y_1, y_2, \dots, y_T)$$

learn a classifier  $\pi_w(\text{feature}(\hat{y}_1, \dots, \hat{y}_{t-1}; x)) = \hat{y}_t$   
classification "decoding policy"

## L2S framework

from  $(x^{(i)}, y^{(i)})_{i=1}^n$  and  $l(\cdot, \cdot)$

learn a good classifier/policy  $\pi_w$  s.t.  $\hat{y}_t = \pi_w(y/\hat{y}_1, \dots, \hat{y}_{t-1}; x)$

$\hat{h}_w(x) \triangleq (\hat{y}_1, \dots, \hat{y}_T)$  "greedy decoder"

s.t.  $l(y^{(i)}, \hat{h}_w(x^{(i)}))$  is small

④ central idea: "reduction" where reduces structured pred. learning problem to problem of cost sensitive classification learning of  $\pi_w$

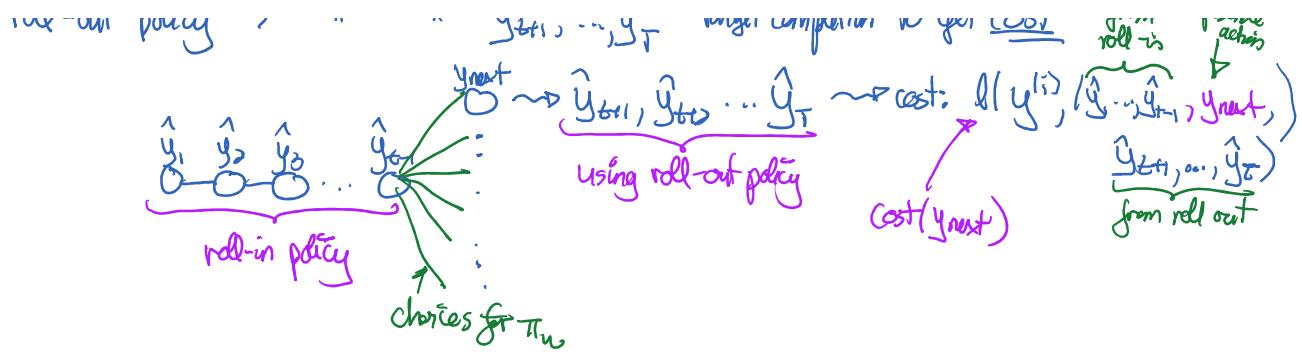
method: generate training data for classifier  $\pi_w$

$$\underbrace{(y^{(i)}_{\text{context}}, x^{(i)}, \text{cost}(y^{(i)}_{\text{next}}))}_{\text{prefix sequence to make next prediction}}$$

prefix sequence to make next prediction

"roll-in" policy  $\rightarrow$  determines how  $\hat{y}_{t+1}$ , context

"roll-out" policy  $\rightarrow$  "target completion" to get cost from roll-in possible actions  
 $y^{(i)} \rightarrow \hat{u}_{t+1}, \hat{u}_2, \dots, \hat{u}_T \rightarrow \text{cost}: l(y^{(i)}, (\hat{u}_{t+1}, \hat{u}_2, \dots, \hat{u}_T, y^{(i)}))$



"reference policy", ideally,  $\pi_{\text{ref}}(\hat{y}_1, \dots, \hat{y}_{t-1}, \hat{y}_{\text{next}})$

$$= \underset{Y_{\text{completion}}}{\operatorname{arg\min}} \ell(y^{(1)}, (\hat{y}_1, \dots, \hat{y}_{t-1}, Y_{\text{next}}, Y_{\text{completion}}))$$

intractable (NP hard) in general

in practice: use heuristic to approximate it

but sometimes, can compute exactly

e.g.  $\pi_{\text{ref}}$  for Hamming loss; is just copy ground truth

LOLS → "locally optimal learning to search"

JCML 2015 "L2S better than your teacher"

roll-in	roll-out	(approximate) reference	mixture $\frac{1}{2}\pi_{\text{ref}} + \frac{1}{2}\pi_w$	learned $\pi_w$
use ground truth "teacher forcing"		(1)		inconsistent
learned (use $\pi_w$ )		consistent not locally optimal	consistent and "locally optimal"	reinforcement learning

Figures and annotations:

- Left: A directed acyclic graph (DAG) with nodes a, b, c, d, e, f, g. Edges: a → c, a → d, b → d, b → e, c → f, d → f, d → g, e → g. A path from a to g is highlighted in red.
- Middle: A plot showing loss vs. step. The loss decreases from 100 at step 0 to 0 at step 5. A green arrow points to step 1, labeled "is cannot do better than teacher".
- Right: A note about reinforcement learning: "can learn a policy Is optimal up to one step derivation?"
- Top right: A note about a distribution D on  $X \times \Sigma$ : "if  $\pi_w$  does well on cost sensitive loss but how does poorly for structured prediction".
- Bottom right: "not using enough information" e.g. ground truth pieces.

Example of approximate  $\pi_{\text{ref}}$ :  $\ell$  is bleu score

consider all possible suffixes from ground truth and pick the best bleu score one

SEARNN : apply L2S to RNN training

[Leblond & al. ICLR 2018] i.e.  $\pi_w(y_{1:t-1}, x) \rightarrow$  RNN cell

$p(y_{next} | y_{1:t-1}, x)$  of a (decoder) RNN

If you use  $-\log p(\hat{y}_{target} | y_{1:t-1}, x)$  as a cost surrogate  
and  $\hat{y}_{target} = \text{ground truth}$

then L2S with  $\pi_{ref}$  roll-in is standard MLE

### \* cost-sensitive surrogate losses choices

a) structured SVM :  $\max_{y^*} [\underbrace{\Delta(y^*)}_{\triangleq c(y^*)} + s(y^*)] - s(y_{target})$

$$y_{target} \stackrel{\triangleq}{=} \underset{y^*}{\operatorname{argmin}} c(y^*)$$

b) "target log-loss" :  $-\log p_w(y_{target} | y_{1:t-1}, x)$

↳ differences with MLE :

- address exposure bias using learned roll-in
- make use of structured loss  $\triangleq \dots$  to predict  $\hat{y}_{target}$  vs. ground truth in MLE

### Structured prediction energy networks (SPENs)

ICML 2016

$$E(x, y; w) \quad (-s(x, y; w))$$

• relax  $y \in \{0, 1\}^T \rightsquigarrow [0, 1]^T$

$h_w(x) = \text{a few steps of projected GD on } E(x, y; w) \quad (\text{approximate prediction procedure})$   
w.r.t.  $\underline{y}$

+ rounding

2016 paper:

SSVM loss  $\rightsquigarrow$  "subgradient" method on  $w$

large loss:  $\max_{y \in [0, 1]^T} (l(y^{(i)}, \bar{y}) - E(x^{(i)}, \bar{y}; w)) + E(x^{(i)}, y^{(i)}; w)$   
requires an extension

$y_{t+1}^{(k)}$       requires an extension  
 approximate "subgradient" is       $-\nabla_w E(x^{(k)}, \hat{y}_{t+1}^{(k)}; w) + \nabla_w E(x^{(k)}, y^{(k)}; w)$   
 because  $\hat{y}$  is approximate max  
 e.g. see Clarke subdifferential  
 for generalization on non-convex fct.

2017 paper: end-to-end learning:

$$h_w(x) = y_0 - \sum_{t=1}^T m_t \frac{\partial}{\partial y} E(x, y_t; w)$$

recursively defined

$$y_1 = y_0 - \eta_1 \frac{\partial}{\partial y} E(x, y_0; w)$$

$$y_2 = y_1 - \eta_2 \frac{\partial}{\partial y} E(x, y_1; w) \text{ etc...}$$

"unrolled optimization"