Learning to search (L2S)

SEARN Hal Daume's PhD thesis

$\text{hw : } X \rightarrow \mathcal{Y}$

$\text{hw}(x) = \arg \max_{y \in \mathcal{Y}} s(x, y; w)$

Special case: learning to do already search

Split $y$ in ordered list of decisions

$(y_1, \ldots, y_T)$

Learn a classifier

$\tilde{T}(\text{feature}(y_1, \ldots, y_{t-1}, x)) = \hat{y}_t$

Classification decoding "policy"

L2S framework
from \((x^{(i)}, y^{(i)})_{i=1}^{n}\) and \(l(\cdot, \cdot)\)

learn a good classifier/policy \(T_w\) s.t. \(\hat{y}_t = T_w(l(y^t, \ldots, \hat{y}^{t-1}, x))\)

\[ h_w(x) = (\hat{y}_1, \ldots, \hat{y}_t) \] "greedy action"

s.t. \(l(y^{(i)}, h_w(x^{(i)}))\) is good

\(\text{Central idea: } "\text{reduction": }\) reduces structured prediction learning to problem of cost sensitive classification learning for \(T_w\)

method: generate training data for classifier \(T_w\)

\[ (\hat{y}^{(i)}, x^{(i)}, \text{cost}(y_{next})) \]

prefix sequence

"roll-in" policy \(\rightarrow\) determines how get \(\hat{y}_{t+1}\) context

"roll-out" policy \(\rightarrow\) \(y_{t+1}, \ldots, y_T\) "target cepthesin" to get cost

\[ \text{cost}(y_{next})\]
"reference policy"

\[ \text{Ideally, } T_{\text{ref}}(Y_1, \ldots, Y_t, Y_{\text{neg}}) = \min_{Y_{\text{completion}}} E(Y_t | Y_1, \ldots, Y_{t-1}, Y_{\text{neg}}, Y_{\text{completion}}) \]

intractable (NP-hard) in general

in practice heuristics be approximate if

but sometimes can compute exactly

\( T_{\text{ref}} \) for Hamming loss is just copy ground truth

LOLS "locally optimal learning by search"

ICML 2015 "LOLS better than your teacher"

<table>
<thead>
<tr>
<th>roll-in</th>
<th>roll-out</th>
<th>reference</th>
<th>learned</th>
<th>learned Tw</th>
</tr>
</thead>
<tbody>
<tr>
<td>T_{\text{ref}}</td>
<td>T_{\text{ref}}</td>
<td>consistent and &quot;locally optimal&quot;</td>
<td>inconsistent</td>
<td>(not using enough information)</td>
</tr>
<tr>
<td>reference policy</td>
<td>reference policy</td>
<td>consistent and not locally optimal</td>
<td>consistent</td>
<td>B.L.s</td>
</tr>
<tr>
<td>learned</td>
<td>learned</td>
<td>consistent</td>
<td>inconsistent</td>
<td>(not using enough information)</td>
</tr>
</tbody>
</table>

\( 15 \times 23 \)  

\[ (1) \quad \text{ie, cannot do better than teacher} \]

\[ (1) \quad \text{ie, there exists a dist. } D \text{ on } X \times Y \]

\[ \text{s.t. } \text{Tw does well on cost sensitive class,} \]

\[ \text{but } \text{it does poorly for structural prediction} \]

\[ 10 \quad 100 \]
SEARNN: apply LAS to RNN training

\[ \pi_W(y_{1:t-1}, x) \rightarrow \text{RNN cell} \]

\[ p(y_{t+1} | y_{1:t-1}, x) \] of a RNN

If use -log \( p(y_{t+1} | y_{1:t-1}, x) \) as cost surrogate

and \( y_{\text{target}} = \text{ground truth} \)

then LAS with ref roll-in is standard NLE

\( \Box \) cost-sensitive surrogate choices:

a) Structured SVM: \( \max_{y'} \left[ \Delta(y') + S(y) \right] - S(y_{\text{target}}) \)

\( \leq c(y') - c(y_{\text{target}}) \)

\( y_{\text{target}} \geq \text{argmin} c(y') \)
b) "target log-loss" : $- \log p(y_{\text{target}} | y_{t-1}, x)$  

- differences with MLE:
  - address the exposure bias using learned roll-in
  - make use of structured loss $\ell(...)$ to pair $y_{\text{target}}$ vs. MLE

**Structured prediction energy networks (SPENs)**

ICML 2016

$$E(x, y, \omega) = -s(x, y, \omega)$$

- relax $y \in \{0, 1\}^T \rightarrow [0, 1]^T$
  
  $\text{h}_w(x) = \text{a few steps of Gradient descent on } E(x, y, \omega)$

  \begin{align*}
  \text{with respect to } & y \\
  \text{(approximate production procedure)}
  \end{align*}

2016 paper: SSVM loss $\rightarrow$ "subgradient" method on $\omega$
huge loss: \[
    \max_{\bar{y} \in \mathcal{L}^T} \left\{ \ell (y^{(i)}, \bar{y}) - E(x^{(i)} \mid \bar{y}; \omega) \right\} + E(x^{(i)} y^{(i)}; \omega)
\]

requires an extension

approximate "subgradient" is \( \nabla \omega E(x^{(i)}, \bar{y}; \omega) \rightarrow E(x^{(i)} y^{(i)}; \omega) \)

\( \text{e.g. Clarke subdifferential} \)

\[
    h_{\omega}(z) = U_0 - \sum_{t=1}^{T} \int dy \ E(x, y_t; \omega)
\]

"unrolled optimization"