L2S

Wednesday, September 14, 2022 1:31 PM

Running Ex: POS Tagging

Given:
- Data: $D = \{(x^{(i)}, y^{(i)})\}_{i=1}^{N}$

$$x^{(i)} = \underset{x_1^{(i)}}{\text{flies}}\ \underset{x_2^{(i)}}{\text{like}}\ \underset{x_3^{(i)}}{\text{a}}\ \underset{x_4^{(i)}}{\text{plant}}$$

$$y^{(i)} = \underset{y_1^{(i)}}{N}\ \underset{y_2^{(i)}}{V}\ \underset{y_3^{(i)}}{D}\ \underset{y_4^{(i)}}{N}$$

$\vec{x}^{(i)} \in \mathcal{X}$ ← input space $y^{(i)} \in \mathcal{Y}_{\vec{x}^{(i)}}$ ← output space (specific to $\vec{x}^{(i)}$)

- Loss: $\ell(y^*, \hat{y}) : \mathcal{Y}_{\vec{x}} \times \mathcal{Y}_{\vec{x}} \to \mathbb{R}$ ← possible outputs for given input

- Hypothesis Space: $\mathcal{H}$ s.t. $\forall h \in \mathcal{H},\ h(\vec{x}) \in \mathcal{Y}_{\vec{x}}\ \forall \vec{x} \in \mathcal{X}$

Goal: Minimize Empirical Risk

$$\hat{h} = \underset{h \in \mathcal{H}}{\text{argmin}} \frac{1}{N} \sum_{i=1}^{N} \ell(y^{(i)}, h(\vec{x}^{(i)}))$$

$$\hat{\theta} = \underset{\theta}{\text{argmin}} \frac{1}{N} \sum_{i=1}^{N} \ell(y^{(i)}, h_\theta(\vec{x}^{(i)}))$$

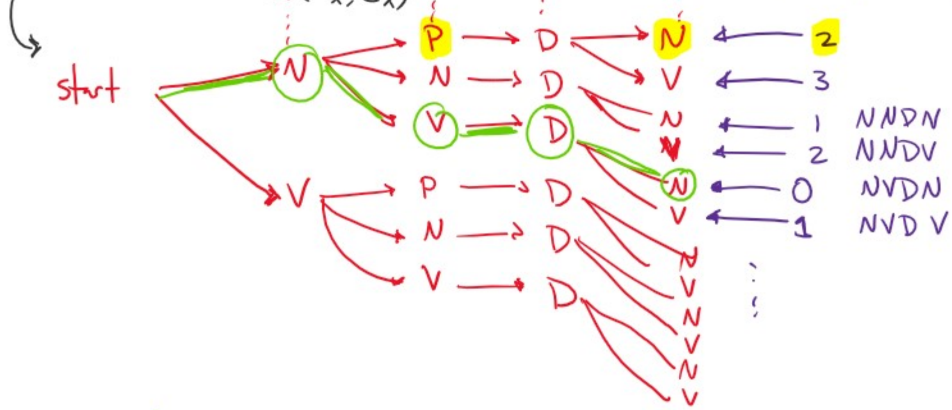
Structured Prediction as Search

$$x_{1:T} = \underset{x_1}{\text{flies}}\ \underset{x_2}{\text{like}}\ \underset{x_3}{\text{a}}\ \underset{x_4}{\text{plant}}$$

Hamming Loss

$\mathcal{Y}_{x_{1:T}}$ = Induced search space for $\vec{x}$ is a D.A.G. $G_x = (V_x, E_x)$

$$\ell(y^*_{1:T}, \hat{y}_{1:T}) = \sum_{t=1}^{T} \mathbb{1}(y^*_t \neq \hat{y}_t)$$



$$y^*_{1:T} = N \quad V \quad D \quad N$$

Def: a trajectory is a path through the search space which is a sequence of output labels $\hat{y}_{1:T}$

Def: a trajectory is a path through the search space which is a sequence of output labels $\hat{y}_{1:T}$

Def: each state $s_t \in V_x$ corresponds to a partial trajectory starting at initial state s_0

$\hat{y}_{1:t}$ for $1 \leq t \leq T$

Def: a policy maps from (observation sequence $x_{1:T}$ and a partial trajectory $y_{1:t}$) a state s_t to a an action a_t (i.e. next label y_{t+1})

Def: a action a_t is a next label y_t

Def: optimal completion cost $c^*_{x,y} : V_x \rightarrow \mathbb{R}$

$$c^*_{x,y}(s_t) = \min_{s' \in \Upsilon_{s'}} \text{loss}(s')$$

where Υ_s is the set of reachable leaves from s_t

Def: an optimal policy is some π^* induced by a labeled training example (x,y) s.t.

$$\pi^*(s_t) = \underset{s' \in N_{s_t}}{\text{argmin}} \; c^*(s')$$

neighbors of s_t

i.e. choose neighbor w/ lowest c^* value

Goal of "Learning to Search": find a good model policy

$$\hat{y}_{1:T} \sim \hat{\pi}_{x^{(i)}}(\text{start})$$

$$\hat{\pi} = \underset{\pi}{\text{argmin}} \sum_{i=1}^{N} \text{loss}\left(y^{(i)}, \hat{y}_{1:T} \sim \pi_{\vec{x}^{(i)}}(\text{start})\right)$$

How? Define a scoring function over states $f_{\theta,x} : V_x \rightarrow \mathbb{R}$

Hopefully $f_{\theta x} \approx c^*_{x,y}$, but we don't know y at test

Def: a greedy policy for scoring function $f_{\theta x}$ is

Def: a ~~greedy policy~~ for scoring function $f_{\theta,x}$ is

$$\pi_\theta(s_t) = \underset{s' \in N_{s_t}}{\arg\max}\ f_{\theta,x}(s')$$

Ex: scoring functions (1) Linear Model (2) NN (3) RNN-LM (4) seq2seq

e.g. probabilities, or log-probabilities