**Overview**

**Motivation:**
- Beam search is commonly used for structured prediction, e.g., speech recognition, machine translation, syntactic parsing, ...  
- Key shortcomings of existing learning algorithms:  
  a. Unaware of beam search  
  b. Not exposed to its own mistakes

**Contributions:**
1. Imitation learning algorithm for learning beam search policies that addresses both issues.  
2. Meta-algorithm that suggests new beam-aware algorithms and captures existing ones.  
3. Regret guarantees for new and existing algorithms inspired by the analysis of DAGger.

**Key Idea:**
Beam trajectories are collected with the learned model at train time, exposing the model to non-optimal beams resulting from its own mistakes, allowing the model to learn how to score neighbors of these beams.

---

**Background**

Learning to search for structured prediction:
- Recast *structured prediction* as *sequential prediction*.
- Example: speech recognition
  - *leaf nodes*: transcription of full sentence
  - *internal nodes*: partial transcription
  - *cost function*: word error rate

Figure 1: Example search space $G = (V,E)$

- *gold sequence* is (000)
- *leaf nodes* annotated with Hamming cost
- *internal nodes* annotated with cost of best reachable leaf

**Surrogate losses**

- *log loss (neighbors)*
- *oracle log loss (neighbors)*
- *reset cost-sensitive margin (last)*

---

**Data collection strategies**

How to collect a beam trajectory $b_0, \ldots, b_k$ used to induce local beam losses?

- **oracle** use policy $\pi^*$ induced by $c^* : V \rightarrow \mathbb{R}$.
- **stop** use $\pi(x, \theta)$; if $c(b, \theta) > 0$, stop the beam trajectory at $k$.
- **reset** use $\pi(x, \theta)$; if $c(b, \theta) > 0$, reset to a beam with gold sequence.
- **continue** always use policy $\pi(x, \theta)$.

Figure 2: Induced beam search space $C_2 = (V_2, E_2)$, for beam size $k = 2$

- each state is now a beam  
- highlighted beams can reach gold sequence

---

**Surrogate losses**

- Log loss (neighbors): $\ell(x, c) = -s_{\pi(x)} + \log \sum_{\hat{v}} \exp(s_{\hat{v}})$.
- Perceptron (first): $\ell(x, c) = \max(0, s_{\pi(x)} - s_{\hat{v}})$.
- Cost-sensitive margin (last): $\ell(x, c) = (c_{\pi(x)} - c_{\pi(x)}) \max(0, 1 + s_{\pi(x)} - s_{\hat{v}})$.
- Upper bound: $\ell(x, c) = \max(0, s_{\hat{v}} - 0)$ where $\delta_i = (c_{\pi(x)} - c_{\pi(x)}) - s_{\hat{v}}$ for $i \in [k-1, \ldots, 0]$. This loss is a convex upper bound to the expected beam transition cost, $E_{\pi(x)}(c(b, b')) : \theta \rightarrow \mathbb{R}$, where $b'$ results by transitioning with scores $s \in \mathbb{R}^k$.

---

**Regret Guarantees**

**Thm. 1:** no-regret guarantees when no-regret algorithm uses explicit loss expectations for beam search policy

Let $l(\pi, \theta) = \sum_{b \in B} \frac{1}{m} \sum_{i=1}^m \ell(x_i, b_{i+1} \downarrow \theta)$. If $\theta_1, \ldots, \theta_m$ is chosen by a deterministic no-regret online learning algorithm, then

$$\sum_{b \in B} \frac{1}{m} \sum_{i=1}^m \ell(x_i, b_{i+1} \downarrow \theta) \leq \min_{\theta} \sum_{b \in B} \frac{1}{m} \sum_{i=1}^m \ell(x_i, b_{i+1} \downarrow \theta) + \gamma_m,$$

with $\gamma_m \rightarrow 0$ when $m \rightarrow \infty$.

**Thm. 2:** no-regret high probability bounds with only access to empirical expectations

Let $\hat{l}(\pi, \theta) = \sum_{b \in B} \frac{1}{m} \sum_{i=1}^m \ell(x_i, b_{i+1})$ generated by sampling $(x_i, y_i)$ from $D$ and sampling $b_{i+1}$ with $\pi(\theta_i)$. Let $\sum_{b \in B} \frac{1}{m} \sum_{i=1}^m \ell(x_i, b_{i+1} \downarrow \theta)$ be as in Thm. 1, then

$$\mathbb{P}\left(\sum_{b \in B} \frac{1}{m} \sum_{i=1}^m \ell(x_i, b_{i+1} \downarrow \theta) \leq \min_{\theta} \sum_{b \in B} \frac{1}{m} \sum_{i=1}^m \ell(x_i, b_{i+1} \downarrow \theta) + \delta\right) \geq 1 - \delta,$$

where $\delta \in (0, 1)$ and $\hat{l}(\theta, \delta) = m^{-1} \sum_{i=1}^m \ell(x_i, b_{i+1} \downarrow \theta) + \eta(\delta, m)$.

**Thm. 3:** regret guarantees for stop and reset data collection policies

See paper for details!