Learned Prioritization for Trading Off Speed and Accuracy

Jiarong Jiang\textsuperscript{1} \quad Adam Teichert\textsuperscript{2} \quad Hal Daumé III\textsuperscript{1} \\
\quad Jason Eisner\textsuperscript{2} \\

\textsuperscript{1}University of Maryland, College Park \\
\textsuperscript{2}Johns Hopkins University \\

ICML workshop on Inferning: Interactions between Inference and Learning
Fast and accurate structured prediction
Introduction

- Fast and accurate structured prediction
- Manual exploration of speed/accuracy tradeoff
  - Prioritization heuristics
    - A* [Klein and Manning, 2003]
    - Hierarchical A* [Pauls and Klein, 2010]
  - Pruning heuristics
    - Coarse-to-fine pruning [Charniak et al., 2006; Petrov and Klein, 2007]
    - Classifier-based pruning [Roark and Hollingshead, 2008]
Introduction

- Fast and accurate structured prediction
- Manual exploration of speed/accuracy tradeoff
  - Prioritization heuristics
    - A* [Klein and Manning, 2003]
    - Hierarchical A* [Pauls and Klein, 2010]
  - Pruning heuristics
    - Coarse-to-fine pruning [Charniak et al., 2006; Petrov and Klein, 2007]
    - Classifier-based pruning [Roark and Hollingshead, 2008]
- Goal: learn a heuristic for your input distribution, grammar, and speed/accuracy needs
Introduction

- Fast and accurate structured prediction
- Manual exploration of speed/accuracy tradeoff
  - Prioritization heuristics
    - A* [Klein and Manning, 2003]
    - Hierarchical A* [Pauls and Klein, 2010]
  - Pruning heuristics
    - Coarse-to-fine pruning [Charniak et al., 2006; Petrov and Klein, 2007]
    - Classifier-based pruning [Roark and Hollingshead, 2008]
- Goal: learn a heuristic for your input distribution, grammar, and speed/accuracy needs
- Objective measure

\[ \text{quality} = \text{accuracy} - \lambda \times \text{time} \]
Agenda-based Parsing

0  Time   1  flies   2  like   3  an   4  arrow  5
N                 V                  P               DET             N
NP 
PP 
NP VP 
S
S
S
NP 
VP 
NP
Agenda-based Parsing

**GRAMMAR**
1. S -> NP VP
2. S -> Vst NP
3. S -> S PP
4. VP -> VP PP
5. VP -> V NP
6. NP -> DET N
7. NP -> NP PP
8. NP -> NP NP
9. PP -> P NP

**AGENDA**

0: Time
1: flies
2: like
3: an
4: arrow

Jiang, Teichert, Daumé, Eisner (UMD, JHU)
Agenda-based Parsing

GRAMMAR

1. S -> NP VP
2. S -> Vst NP
3. S -> S PP
4. VP -> VP PP
5. VP -> V NP
6. NP -> DET N
7. NP -> NP PP
8. NP -> NP NP
9. PP -> P NP

AGENDA

10 3NP 5

0  Time  1  flies  2  like  3  an  4  arrow  5
Priority-based Inference

Agenda-based Parsing

**GRAMMAR**
1. S -> NP VP
2. S -> S PP
3. NP -> DET N
4. NP -> NP PP
5. NP -> NP NP
6. PP -> P NP

**AGENDA**

0  Time  1  flies  2  like  3  an  4  arrow  5

Jiang, Teichert, Daumé, Eisner (UMD, JHU)
Priority-based Inference

Agenda-based Parsing

GRAMMAR
1  S -> NP VP
2  S -> S PP
1  VP -> VP PP
2  VP-> V NP
1  NP -> DET N
2  NP -> NP PP
3  NP -> NP NP
0  PP -> P NP

AGENDA

0  Time 1  flies 2  like 3  an 4  arrow 5
Agenda-based Parsing

GRAMMAR
1. S -> NP VP
2. S -> S PP
3. S -> Vst NP
4. VP -> VP PP
5. VP -> V NP
6. VP -> Vst NP
7. NP -> DET N
8. NP -> NP PP
9. NP -> NP NP
10. PP -> P NP
11. S 8
12. NP 10
13. NP 4
14. NP 3
15. VP 4
16. VP 3
17. S 8
18. NP 10
19. NP 4
20. NP 3
21. VP 4
22. VP 3
23. S 8
24. NP 10
25. NP 4
26. NP 3
27. VP 4
28. VP 3
29. S 8
30. NP 10
31. NP 4
32. NP 3
33. VP 4
34. VP 3

AGENDA
10 2PP 5
12 2VP 5

Prioritize based Inference
Priority-based Inference

Agenda-based Parsing

GRAMMAR
1. S -> NP VP
6. S -> Vst NP
2. S -> S PP
1. VP -> VP PP
2. VP -> V NP
1. NP -> DET N
2. NP -> NP PP
3. NP -> NP NP
0. PP -> P NP

AGENDA
10. PP
12. VP

0. Time
1. flies
2. like
3. an
4. arrow
5.
All experiments are on Penn Treebank WSJ with sentence length $\leq 15$.

Preliminary results setup:
- Berkeley latent variable PCFG trained on section 2-20
- Training set: 100 sentences from section 21
- Evaluated on the same 100 sentences

Baseline 1: Exhaustive Search
Recall: 93.3; Relative number of pops: 3.0x

Baseline 2: Uniform Cost Search (UC)
Recall: 93.3; Relative number of pops: 1.0x

Baseline 3: Pruned Uniform Cost Search
Recall: 92.0; Relative number of pops: 0.33x
Agenda-based Parsing as a Markov Decision Process

- State space: current chart and agenda
- Action: pop a partial parse from the agenda
- Transition: Given the chosen action, deterministically updates chart and pushes other parses to the agenda
- Policy: computes action priorities from extracted features

\[ \pi_\theta(s) = \arg \max_a \theta \cdot \phi(a, s) \]

(Delayed) Reward

\[ \text{reward} = \text{accuracy} - \lambda \times \text{time} \]

- accuracy = labeled span recall
- time = # of pops from agenda
Agenda-based Parsing as a Markov Decision Process

- State space: current chart and agenda
- Action: *pop* a partial parse from the agenda
- Transition: Given the chosen action, deterministically updates chart and pushes other parses to the agenda
- Policy: computes action priorities from extracted features

\[ \pi_\theta(s) = \arg \max_a \theta \cdot \phi(a, s) \]

- (Delayed) Reward

\[ \text{reward} = \text{accuracy} - \lambda \times \text{time} \]

- accuracy = labeled span recall
- time = # of pops from agenda

Learning Policy = Learning Prioritization Function
Decoding as a Markov Decision Process (MDP)

**GRAMMAR**

1. $S \rightarrow NP \ VP$
2. $S \rightarrow Vst \ NP$
3. $S \rightarrow S \ PP$
4. $VP \rightarrow VP \ PP$
5. $VP \rightarrow V \ NP$
6. $NP \rightarrow DET \ N$
7. $NP \rightarrow NP \ PP$
8. $NP \rightarrow NP \ NP$
9. $PP \rightarrow P \ NP$

**AGENDA**

- $10???$
- $12???$
- $2PP_5$
- $2VP_5$
Boltzmann Exploration

- Transition at test time: deterministic
- Transition at training time: exploration with stochastic policies: $\pi_{\theta}(a \mid s)$.
- Boltzmann exploration:

$$\pi_{\theta}(a \mid s) = \frac{1}{Z(s)} \exp \left[ \frac{1}{\text{temp}} \theta \cdot \phi(a, s) \right]$$

- Temperature $\to 0$, exploration $\to$ exploitation
- A trajectory $\tau = \langle s_0, a_0, r_0, s_1, a_1, r_1, \ldots, s_T, a_T, r_T \rangle$.
- Expected future reward:

$$R = \mathbb{E}_{\tau \sim \pi_{\theta}} [R(\tau)] = \mathbb{E}_{\tau \sim \pi_{\theta}} \left[ \sum_{t=0}^{T} r_t \right].$$
Policy Gradient

- Find parameters that maximize the expected reward with respect to the induced distribution over trajectories

- Policy gradient [Sutton et al., 2000]
  The gradient of the objective

\[
\nabla_\theta \mathbb{E}_\tau [R(\tau)] = \mathbb{E}_\tau \left[ R(\tau) \sum_{t=0}^{T} \nabla_\theta \log \pi(a_t | s_t) \right]
\]

where

\[
\nabla_\theta \log \pi_\theta(a | s) = \frac{1}{\text{temp}} \left( \tilde{\phi}(a_t, s_t) - \sum_{a' \in A} \pi_\theta(a' | s_t) \tilde{\phi}(a', s_t) \right)
\]
Features

1. Width of partial parse
2. Viterbi inside score
3. Touches start of sentence?
4. Touches end of sentence?
5. Ratio of width to sentence length
6. $\log p(label \mid \text{prev POS})$ and $\log p(label \mid \text{next POS})$ 
   (statistics extracted from labeled trees, word POS assumed to be most frequent)
7. Case pattern of first word in partial parse and previous/next word
8. Punctuation pattern in partial parse (five most frequent)
Preliminary results:

<table>
<thead>
<tr>
<th>Method</th>
<th>Recall</th>
<th>Relative # of pops</th>
</tr>
</thead>
<tbody>
<tr>
<td>Policy Gradient w/ Boltzmann Exploration</td>
<td>56.4</td>
<td>0.46x</td>
</tr>
<tr>
<td>Uniform cost search</td>
<td>93.3</td>
<td>1.0x</td>
</tr>
<tr>
<td>Pruned uniform cost search</td>
<td>92.0</td>
<td>0.33x</td>
</tr>
</tbody>
</table>

Main Difficulty: Jiang, Teichert, Daumé, Eisner (UMD, JHU)
Policy Gradient with Boltzmann Exploration

- Preliminary results:

<table>
<thead>
<tr>
<th>Method</th>
<th>Recall</th>
<th>Relative # of pops</th>
</tr>
</thead>
<tbody>
<tr>
<td>Policy Gradient w/ Boltzmann Exploration</td>
<td>56.4</td>
<td>0.46x</td>
</tr>
<tr>
<td>Uniform cost search</td>
<td>93.3</td>
<td>1.0x</td>
</tr>
<tr>
<td>Pruned uniform cost search</td>
<td>92.0</td>
<td>0.33x</td>
</tr>
</tbody>
</table>

- Main Difficulty:

Which actions were “responsible” for a trajectory’s reward?
Reward Shaping

- Goal: give the agent reward *earlier* in a trajectory in order to improve its convergence rate
- Push back reward to actions

\[ \tilde{r}(s, a) = \begin{cases} 
\frac{\xi(a)}{n} - \lambda & \text{if } a \text{ is a full parse tree} \\
\frac{1}{n} - \lambda & \text{if } a \text{ is in the true parse} \\
-\lambda & \text{otherwise}
\end{cases} \]

\( \xi(s) \): a negative reward for actions which received early reward for constituents that were not in the final parse

- Property: \( R(\tau) = \sum_{t=0}^{T} \tilde{r}(s, a) \)
Reward Shaping
Attempt 2: Policy Gradient with Reward Shaping

Reward Shaping
 Attempt 2: Policy Gradient with Reward Shaping

Reward Shaping

- **The man ate**
- **r=0-α**
- **r=1/3-α**
- **PP**
- **NP**
- **VP**
- **VP**
- **FRAG**
- **R=1/3-α4**
- **r=0-α**
- **r=1/3-α**
- **PP**
- **r=0-α**
- **r=1/3-α**
- **PP**
- **r=0-α**
- **FRAG**
- **S**
- **R=3/3-α4**
- **R=3/3-α3**
Reward Shaping

**Gradient step:**

\[
\nabla_\theta \mathbb{E}_\tau [R(\tau)] = \nabla_\theta \mathbb{E}_\tau [\tilde{R}(\tau)] = \mathbb{E}_\tau \left[ \sum_{t=0}^{T} \left( \sum_{t'=t}^{T} \gamma^{t'-t} \tilde{r}_{t'} \right) \right] \nabla_\theta \log \pi(a_t | s_t)
\]
Reward Shaping

**Gradient step:**

$$\nabla_{\theta} \mathbb{E}_{\tau}[R(\tau)] = \nabla_{\theta} \mathbb{E}_{\tau}[\tilde{R}(\tau)] = \mathbb{E}_{\tau} \left[ \sum_{t=0}^{T} \left( \sum_{t'=t}^{T} \gamma^{t'-t} \tilde{r}_{t'} \right) \nabla_{\theta} \log \pi(a_t | s_t) \right]$$

**Preliminary results:**

<table>
<thead>
<tr>
<th>Method</th>
<th>Recall</th>
<th>Relative # of pops</th>
</tr>
</thead>
<tbody>
<tr>
<td>Policy Gradient w/ Reward Shaping</td>
<td>76.5</td>
<td>0.13x</td>
</tr>
<tr>
<td>Policy Gradient w/ Boltzmann Exploration</td>
<td>56.4</td>
<td>0.46x</td>
</tr>
<tr>
<td>Uniform cost search</td>
<td>93.3</td>
<td>1.0x</td>
</tr>
<tr>
<td>Pruned uniform cost search</td>
<td>92.0</td>
<td>0.33x</td>
</tr>
</tbody>
</table>

Main difficulty: Only a few trajectories are reasonable!

Jiang, Teichert, Daumé, Eisner (UMD, JHU)
Reward Shaping

Gradient step:

\[ \nabla_{\theta} \mathbb{E}_{\tau} [R(\tau)] = \nabla_{\theta} \mathbb{E}_{\tau} [\tilde{R}(\tau)] = \mathbb{E}_{\tau} \left[ \sum_{t=0}^{T} \left( \sum_{t'=t}^{T} \gamma^{t'-t} \tilde{r}_{t'} \right) \nabla_{\theta} \log \pi(a_t | s_t) \right] \]

Preliminary results:

<table>
<thead>
<tr>
<th>Method</th>
<th>Recall</th>
<th>Relative # of pops</th>
</tr>
</thead>
<tbody>
<tr>
<td>Policy Gradient w/ Reward Shaping</td>
<td>76.5</td>
<td>0.13x</td>
</tr>
<tr>
<td>Policy Gradient w/ Boltzmann Exploration</td>
<td>56.4</td>
<td>0.46x</td>
</tr>
<tr>
<td>Uniform cost search</td>
<td>93.3</td>
<td>1.0x</td>
</tr>
<tr>
<td>Pruned uniform cost search</td>
<td>92.0</td>
<td>0.33x</td>
</tr>
</tbody>
</table>

Main difficulty:

Only a few trajectories are reasonable!
Oracle Actions

- Focus on high-reward regions of policy space
Oracle Actions

- Focus on high-reward regions of policy space
- Oracle action: an action that leads to a maximum-reward tree, where reward is defined in terms of accuracy \textit{and} speed
Oracle Actions

- Focus on high-reward regions of policy space
- Oracle action: an action that leads to a maximum-reward tree, where reward is defined in terms of accuracy and speed
- How to get oracle actions?
  - Ground truth of a sentence
  - Exact parse with the best speed-accuracy tradeoff
Oracle Actions

- Focus on high-reward regions of policy space
- Oracle action: an action that leads to a maximum-reward tree, where reward is defined in terms of accuracy and speed
- How to get oracle actions?
  - Ground truth of a sentence
  - Exact parse with the best speed-accuracy tradeoff
- Apprenticeship learning via classification
  1. Generate classification examples \((s_t, a_t)\) labeled according to oracle actions
  2. Train a maximum entropy classifier
  3. Classifier objective: maximize number of times policy matches oracle action
Apprenticeship Learning via Classification

Preliminary results:

<table>
<thead>
<tr>
<th>Method</th>
<th>Recall</th>
<th>Relative # of pops</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apprenticeship Learning via Classification</td>
<td>84.2</td>
<td>0.85x</td>
</tr>
<tr>
<td>Policy Gradient w/ Reward Shaping</td>
<td>76.5</td>
<td>0.13x</td>
</tr>
<tr>
<td>Policy Gradient w/ Boltzmann Exploration</td>
<td>56.4</td>
<td>0.46x</td>
</tr>
<tr>
<td>Uniform cost search</td>
<td>93.3</td>
<td>1.0x</td>
</tr>
<tr>
<td>Pruned uniform cost search</td>
<td>92.0</td>
<td>0.33x</td>
</tr>
</tbody>
</table>
Apprenticeship Learning via Classification

- Preliminary results:

<table>
<thead>
<tr>
<th>Method</th>
<th>Recall</th>
<th>Relative # of pops</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apprenticeship Learning via Classification</td>
<td>84.2</td>
<td>0.85x</td>
</tr>
<tr>
<td>Policy Gradient w/ Reward Shaping</td>
<td>76.5</td>
<td>0.13x</td>
</tr>
<tr>
<td>Policy Gradient w/ Boltzmann Exploration</td>
<td>56.4</td>
<td>0.46x</td>
</tr>
<tr>
<td>Uniform cost search</td>
<td>93.3</td>
<td>1.0x</td>
</tr>
<tr>
<td>Pruned uniform cost search</td>
<td>92.0</td>
<td>0.33x</td>
</tr>
</tbody>
</table>

- Main difficulty:

**Too hard to imitate oracle with our features!**
Goal: “interleaving” oracle actions with policy actions both feasible and sensible

Let $\pi$ be an arbitrary policy and let $\delta \in [0, 1]$. The oracle infused policy $\pi^+_\delta$ is defined as follows:

$$\pi^+_\delta(a | s) = \delta \pi^*(a | s) + (1 - \delta) \pi(a | s)$$

- $\delta = 1$: the classifier-based approach
- $\delta = 0$: policy gradient
- $\delta = 0.8^{\text{epoch}}$
## Oracle-Infused Policy Gradient

### Preliminary results:

<table>
<thead>
<tr>
<th>Method</th>
<th>Recall</th>
<th>Relative # of pops</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle-Infused Policy Gradient</td>
<td>91.2</td>
<td>0.46x</td>
</tr>
<tr>
<td>Apprenticeship Learning via Classification</td>
<td>84.2</td>
<td>0.85x</td>
</tr>
<tr>
<td>Policy Gradient w/ Reward Shaping</td>
<td>76.5</td>
<td>0.13x</td>
</tr>
<tr>
<td>Policy Gradient w/ Boltzmann Exploration</td>
<td>56.4</td>
<td>0.46x</td>
</tr>
<tr>
<td>Uniform cost search Pruned uniform cost search</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>93.3</td>
<td>1.0x</td>
</tr>
<tr>
<td></td>
<td>92.0</td>
<td>0.33x</td>
</tr>
</tbody>
</table>
**Final Results** Setup:
- Berkeley latent variable PCFG trained on sections 2-21
- RL (if any) trained on section 22
- evaluated on section 23

**Baselines:**
- (HA*) a Hierarchical A* parser [3] with same pruning threshold at each hierarchy level
- (UC) uniform cost search
- (UC_p) pruned uniform cost search
- (A*_p) an A* variant, on which we decrease the pruning threshold if no tree is returned
- (CTF) an agenda-based coarse-to-fine parser [4].
**Figure:** Pareto frontiers: Our $I^+$ parser at different values of $\lambda$, against the baselines at different pruning levels. *Lower and further right* is better.
A novel oracle-infused variant of the policy gradient algorithm for reinforcement learning

Learn a fast and accurate parser with only a simple set of features

Limitation of the model:
  - Feature effectiveness v.s. cost
  - Stop criteria


5. S. Petrov and D. Klein. 2007. Improved inference for unlexicalized parsing. In NAACL/HLT.