Combining Active Learning and Partial Annotation for Domain Adaptation of a Japanese Dependency Parser

Daniel FLANNERY\textsuperscript{1} \hspace{1cm} Shinsuke MORI\textsuperscript{2}

\textsuperscript{1}Vitei Inc. (work at Kyoto University)
\textsuperscript{2}Kyoto University

IWPT 2015, July 22nd
My first international presentation!!
  “Parsing Without Grammar” [Mori 95]

This is the second!!
Statistical Parsing

- Technology for finding the structure of natural language sentences
- Performed after low-level tasks
  - word segmentation (ja, zh, ...)
  - part-of-speech tagging
- Parse trees useful for higher-level tasks
  - information extraction
  - machine translation
  - automatic summarization
  - etc.
Portability Problems

- Accuracy drop on a test in a different domain [Petrov 10]
- Need systems for specialized text (patents, medical, etc.)

In this way print plate 31 is positioned against elastic material 32.
Parser Overview

- EDA parser: Easily Domain Adaptable Parser [Flannery 12]
  http://plata.ar.media.kyoto-u.ac.jp/tool/EDA/home-e.html
  - 1st order Maximum Spanning Tree parsing [McDonald 05]
  - Allows partial annotation: only annotate some words in a sentence

- Use this flexibility for domain adaptation
  - Active learning: Select only informative examples for annotation
  - Goal: Reduce the amount of data needed to train a parser for a new type of text
Pointwise Estimation of Edge Scores

Choosing a head is an n-class classification problem

\[ \sigma(\langle i, d_i \rangle) = p(d_i|\bar{w}, i), \quad (d_i \in [0, n] \land d_i \neq i) \]

Calculate edge scores independently

Features

1. Distance between dependent/head
2. Surface forms/POS of dependent/head
3. Surface/POS for 3 surrounding words
4. No surrounding dependencies! (1st order)
Our method can use a partially annotated corpus

- Only annotate some words with heads
- Pointwise estimation

Cf. fully annotated corpus
- Must annotate all words with heads
1. Train classifier $C$ from labeled training set $D_L$
2. Apply $C$ to the unlabeled data set $D_U$ and select $I$, the $n$ most informative training examples
3. Ask oracle to label examples in $I$
4. Move training instances in $I$ from $D_U$ to $D_L$
5. Train a new classifier $C'$ on $D_L$
6. Repeat 2 to 5 until stopping condition is fulfilled
Criteria used to select training examples to annotate from the pool of unlabeled data

Should allow for units smaller than full sentences

Problems

- Single-word annotations for a sentence are too difficult
- Realistically, annotators must think about dependencies for some other words in the sentence (not all of them)

Need to measure actual annotation time to confirm the query strategy’s performance!
Tree Entropy [Hwa 04]

- Criterion for selecting sentences to annotate with full parse trees

\[ H(V) = - \sum_{v \in V} p(v) \log(p(v)) \]

- Models distribution of trees for a sentence
- \( V \) is the set of possible trees, \( p(v) \) is the probability of choosing a particular tree \( v \)
- In our case, change the unit from sentences to words and model the distribution of heads for a single word (head entropy)
  - use the edge score \( p(d_i | w, i) \) in place of \( p(v) \)
- Rank all words in the pool, and annotate those with the highest values (1-Stage Selection)
1-Stage Selection

- Change the selection unit from sentences to words
  - Need to model the distribution of heads for a single word
  - Simple application of tree entropy to the word case

- Instead of probability for an entire tree \( p(v) \), use the edge score \( p(d_i | \tilde{w}, i) \) of a word-head pair given by a parsing model

- Rank all words by head entropy, and annotate those with the highest values

- The annotator must consider the overall sentence structure
2-Stage Selection

1. Rank sentences by summed head entropy

2. Rank words in each by head entropy

3. Annotate a fixed fraction
   - partial: annotate top $r = \frac{1}{3}$ of words
   - full: annotate all words
Example

- Pool of three sentences

<table>
<thead>
<tr>
<th>sent.</th>
<th>words</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1:</td>
<td>A/0.2</td>
</tr>
<tr>
<td></td>
<td>B/0.1</td>
</tr>
<tr>
<td></td>
<td>C/0.5</td>
</tr>
<tr>
<td></td>
<td>D/0.1</td>
</tr>
<tr>
<td>s2:</td>
<td>E/0.4</td>
</tr>
<tr>
<td></td>
<td>F/0.3</td>
</tr>
<tr>
<td></td>
<td>G/0.1</td>
</tr>
<tr>
<td></td>
<td>H/0.2</td>
</tr>
<tr>
<td>s3:</td>
<td>I/0.4</td>
</tr>
<tr>
<td></td>
<td>J/0.2</td>
</tr>
<tr>
<td></td>
<td>K/0.3</td>
</tr>
<tr>
<td></td>
<td>L/0.2</td>
</tr>
</tbody>
</table>

- 1-stage

C, E, I, F, K, ...

- 2-stage, \( r = \frac{1}{2} \)

<table>
<thead>
<tr>
<th>sent.</th>
<th>sum</th>
<th>words</th>
</tr>
</thead>
<tbody>
<tr>
<td>s3:</td>
<td>1.1</td>
<td>I/0.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>J/0.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>K/0.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>L/0.2</td>
</tr>
<tr>
<td>s2:</td>
<td>1.0</td>
<td>E/0.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F/0.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>G/0.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>H/0.2</td>
</tr>
<tr>
<td>s1:</td>
<td>0.9</td>
<td>A/0.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B/0.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C/0.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>D/0.1</td>
</tr>
</tbody>
</table>
### Evaluation Settings

<table>
<thead>
<tr>
<th>ID</th>
<th>source</th>
<th>sent.</th>
<th>words /sent.</th>
<th>dep.</th>
</tr>
</thead>
<tbody>
<tr>
<td>EHJ-train</td>
<td>Dictionary examples</td>
<td>11,700</td>
<td>12.6</td>
<td>136,264</td>
</tr>
<tr>
<td>NKN-train</td>
<td>Newspaper articles</td>
<td>9,023</td>
<td>29.2</td>
<td>254,402</td>
</tr>
<tr>
<td>JNL-train</td>
<td>Journal abstracts</td>
<td>322</td>
<td>38.1</td>
<td>11,941</td>
</tr>
<tr>
<td>NPT-train</td>
<td>NTCIR patents</td>
<td>450</td>
<td>40.8</td>
<td>17,928</td>
</tr>
<tr>
<td>NKN-test</td>
<td>Newspaper articles</td>
<td>1,002</td>
<td>29.0</td>
<td>28,035</td>
</tr>
<tr>
<td>JNL-test</td>
<td>Journal abstracts</td>
<td>32</td>
<td>34.9</td>
<td>1,084</td>
</tr>
<tr>
<td>NPT-test</td>
<td>NTCIR patents</td>
<td>50</td>
<td>45.5</td>
<td>2,225</td>
</tr>
</tbody>
</table>

- The initial model: EHJ
- The target domains: NKN, JNL, NPT
  - Manual annotation except for POS by KyTea
  - Some are publicly available [Mori 14].

[http://plata.ar.media.kyoto-u.ac.jp/data/word-dep/home-e.html](http://plata.ar.media.kyoto-u.ac.jp/data/word-dep/home-e.html)
Exp.1: Number of Annotations

- Reduction of the **number** of in-domain dependencies

- **Simulation** by selecting the gold standard dependency labels from the annotation pool

- Necessary but not sufficient condition for an effective strategy

- Simple baselines
  - **random** simply selects words randomly from the pool.
  - **length** strategy simply chooses words with the longest possible dependency length.

- One iteration:
  1. a batch of one hundred dependency annotations
  2. model retraining
  3. accuracy measurement
length and 2-stage-full work good for the first ten iterations but soon begin to falter.

2-stage-partial > 1-stage > others
Exp.2: Annotation Pool Size

- NKN annotation pool size $\approx 21.3 \times$ JNL, $14.2 \times$ NPT
- The total number of dependencies selected is 3k (only 1.2% of NKN-train).
- 2-stage accuracy may suffer when a much larger fraction of the pool is selected.
  - Because the 2-stage strategy chooses some dependencies with lower entropy over competing ones with higher entropy from other sentences in the pool.
- Test a small pool case like JNL or NPT
  - First 12,165 dependencies as the pool
After 17 rounds of annotation
- 1-stage > 2-stage partial > 2-stage full

The relative performance is influenced by the pool size.
- 1-stage is robust.
- 2-stage partial can outperform it for a very large pool.
Exp.3: Time Required for Annotation

- **Annotation time** for a more realistic evaluation
  - Simulation experiments are still common in active learning
  - Increasing interest in measuring the true costs [Settles 08]

- **Settings for annotation time measurement**
  - 2-stage strategies
  - Initial model: EHJ-train *plus* NKN-train
  - Target domain: blog in BCCWJ (Balanced Corpus of Contemporary Written Japanese [Maekawa 08])
  - Pool size: 747 sentences
  - One iteration: 2k dependency annotations
Annotation Time Estimation

- A single annotator, 2-stage partial and full
  - one hour for partial \(\Rightarrow\) one hour for full \(\Rightarrow\) one hour for partial ...

<table>
<thead>
<tr>
<th>method</th>
<th>0.25 [h]</th>
<th>0.5 [h]</th>
<th>0.75 [h]</th>
<th>1.0 [h]</th>
</tr>
</thead>
<tbody>
<tr>
<td>partial</td>
<td>226</td>
<td>458</td>
<td>710</td>
<td>1056</td>
</tr>
<tr>
<td>full</td>
<td>141</td>
<td>402</td>
<td>756</td>
<td>1018</td>
</tr>
</tbody>
</table>

- After one hour the number of annotations was almost identical
  - For full the annotator was forced to check the annotation standard for subtle linguistic phenomena.
    - partial allows the annotator to delete the estimated heads.
  - 1.4k dependencies per hour
Applied estimated time by the speeds measured in blog

2-stage partial > 2-stage full

The difference becomes pronounced after 0.5[h].
## Results for Additional Domains

<table>
<thead>
<tr>
<th>ID</th>
<th>source</th>
<th>sent.</th>
<th>words/sent.</th>
<th>dep/sent.</th>
</tr>
</thead>
<tbody>
<tr>
<td>EHJ-train</td>
<td>Dictionary examples</td>
<td>11,700</td>
<td>12.6</td>
<td>136,264</td>
</tr>
<tr>
<td>NKN-train</td>
<td>Newspaper articles</td>
<td>9,023</td>
<td>29.2</td>
<td>254,402</td>
</tr>
<tr>
<td>JNL-train</td>
<td>Journal abstracts</td>
<td>322</td>
<td>38.1</td>
<td>11,941</td>
</tr>
<tr>
<td>NPT-train</td>
<td>NTCIR patents</td>
<td>450</td>
<td>40.8</td>
<td>17,928</td>
</tr>
<tr>
<td>NKN-test</td>
<td>Newspaper articles</td>
<td>1,002</td>
<td>29.0</td>
<td>28,035</td>
</tr>
<tr>
<td>JNL-test</td>
<td>Journal abstracts</td>
<td>32</td>
<td>34.9</td>
<td>1,084</td>
</tr>
<tr>
<td>NPT-test</td>
<td>NTCIR patents</td>
<td>50</td>
<td>45.5</td>
<td>2,225</td>
</tr>
</tbody>
</table>

- **Small pool sizes**

To JNL or NPT in (Annotations)

- 1-stage > 2-stage partial
  - The pool size is small.
  - 3k dependencies = 25.1% for JNL and 16.7% for NPT
- 2-stage partial > 2-stage full
To JNL or NPT (Time)

- Estimated annotation time
- 2-stage partial > 2-stage full
- The gap is the largest for NPT and the smallest for JNL.
Reduction in In-domain Data

<table>
<thead>
<tr>
<th>domain</th>
<th>random</th>
<th>full</th>
<th>partial</th>
</tr>
</thead>
<tbody>
<tr>
<td>NKN</td>
<td>3,000</td>
<td>–</td>
<td>1,300</td>
</tr>
<tr>
<td>JNL</td>
<td>3,000</td>
<td>1,800</td>
<td>900</td>
</tr>
<tr>
<td>NPT</td>
<td>2,700</td>
<td>–</td>
<td>1,500</td>
</tr>
</tbody>
</table>

- random: #annotations needed for the highest accuracy by the random baseline
- full, partial: #annotations needed for the full and partial versions of 2-stage to outperform it
- 2-stage full had mixed results.
- 2-stage partial offers large savings consistently.
Conclusion

- A practical criterion for active learning of a dependency parser
  - Entropy-based
  - Semi-sentence-based

- 2-stage partial: the best when a large size of pool is available

- The corpora and the parser available at http://plata.ar.media.kyoto-u.ac.jp/home-e.html

- Future work
  - Combine with a 2nd or 3rd order parser
References


