A Statistical Parser for Hindi in ≤ 2 weeks

Corpus-Based Natural Language Processing Workshop, AU-KBC Center, Madras Institute of Technology, December 17-31, 2001

Anoop Sarkar
Vivek Mehta
Mona Parakh
T. Papi Reddy
Pranjal Kanade

A Statistical Parser for Hindi
Initial Goals

- Build a statistical parser for Hindi (provides single-best parse for a given input)
- Train on the Hindi Treebank (built at LTRC, Hyderabad)
- Active learning experiments: Informative sampling of data to be annotated
• Annotated corpus: marked for verb-argument relations.
  • Corpus size: 338 sentences.
  • Corpus description: extracts from Premchand's novels.
  • Abad

Annotated corpus for Hindi, "Anncorpa" prepared at LTRC, IIIT, Hyderabad

Initial Linguistic Resources
Annotators who were part of the team manually corrected the following:

- Inconsistency of tags resolved.
- Resolved the discrepancies in the tags.
- Problems of local word grouping resolved.
- Explicitly marked the clause boundaries to disambiguate long complex sentences without punctuation in the corpus.

- Inconsistency of tags resolved.
- Resolved the discrepancies in the tags.
- Problems of local word grouping resolved.
- Explicitly marked the clause boundaries to disambiguate long complex sentences without punctuation in the corpus.

4}
Goals: Reconsidered

- Corpus Cleanup and Correction
- Default rules and Explicit Dependency Trees
- Various models of parsing based on the Treebank
- Trigram tagger/chunker
- Papi's parser and sentence units
- Fully lexicalized statistical parser (with smoothing)
- Probabilistic CFG parser (stemming, no smoothing)
- Fully lexicalized statistical parser (with smoothing)
Explicit dependencies are not marked.

In ten minutes the green garden was destroyed.

\[
\text{was-destroy-past} \quad \text{nashtra-ho-gaya:} \quad \text{ten minutes in green garden}
\]

Raw corpus:

Default rules and explicit dependency trees.
• Defaultrulesarelistedintheguidelines
• Evaluatedthedefaultrulesandbuiltaprogramtoconvertoriginalcorpus
• Pusintoevidenteedependencytrees
• Defaultrulesarelistedintheguidelines
In ten minutes the green garden was destroyed.

default rules and explicit dependency trees
नाश्ता-हो-गया<  

> बागे<  

> हरा-भरा<  

> मिनीत-मेले<  

> दसा<  

> K1<  

> k7.1<  

> v<
DefaultrulesandExplicitDependencyTrees

- Defaultrulescouldnothandle24outof334sentences
- ad-hoc defaults for multiple sentence units within a single sentence
  - added $\checkmark$ as parent of all clauses
- Default rules could not handle 24 out of 334 sentences
- Default rules and Explicit Dependency Trees
TrigramTagger/Chunker

• Input:

vyasana-tha//v//co
kuchha//adv//co
bagavani-ka//6//co
bhavamishri-pratamadhya-prakai//adj//cb
mushthi//adj//cb
mandasa//adj//cb
tahalisna//adj//cb

Converted to representation for Tagger:

{vyasana-tha:
  kuchha:adv
  bagavani-ka/6
  pratamadhya-prakai:mushthi bhavamishri-pratamadhya-prakai//adj//cb]
tahalisna mandasa//adj//cb}

Input:

Trigram Tagger/Chunker
TrigramTagger/Chunker
• Bootstrapped using existing supertagger code
• 70-30 training-test split
• Testing on training data performance:
  – tag accuracy: 95.17%
  – chunk accuracy: 96.69%
• Unseen Test data
  – tag accuracy: 55%
  – chunk accuracy: 71.8%
• Used some existing code written earlier for prob CKY parsing

• Used PCFG parser to compute the best derivation for a given sentence

• Estimated probabilities for each rule using counts from the Treebank

• Extracted probabilistic rules from the Treebank

Probabilistic CFG Parser
Probabilistic CFG Parser

12

>miniTa_meM<

>naShTa_ho_gayA<

<k1

'>bAga'<

'hara-bhara'

'>miniTa_meM<

'>bAga'<

'>daSA<

<k1
K1 \rightarrow \text{hara-bhara baga}
K7.1 \rightarrow \text{dasa muni tapa mem}
\wedge \rightarrow K7.1 K1 \text{nashita ho-gaya}
Probabilistic CFG Parser: Results on Training Data

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>1min 27secs</td>
</tr>
<tr>
<td>Number of sentence</td>
<td>310</td>
</tr>
<tr>
<td>Number of Error sentence</td>
<td>13</td>
</tr>
<tr>
<td>Number of Skip sentence</td>
<td>0</td>
</tr>
<tr>
<td>Number of Valid sentence</td>
<td>297</td>
</tr>
<tr>
<td>Bracketing Recall</td>
<td>76.94</td>
</tr>
<tr>
<td>Bracketing Precision</td>
<td>86.29</td>
</tr>
<tr>
<td>Complete match</td>
<td>48.82</td>
</tr>
<tr>
<td>Average crossing</td>
<td>0.12</td>
</tr>
<tr>
<td>No crossing</td>
<td>91.25</td>
</tr>
<tr>
<td>2 or less crossing</td>
<td>99.33</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>------------</td>
</tr>
<tr>
<td>2 or less crossing</td>
<td>94.95</td>
</tr>
<tr>
<td>No crossing</td>
<td>66.33</td>
</tr>
<tr>
<td>Average crossing</td>
<td>60.05</td>
</tr>
<tr>
<td>Complete match</td>
<td>25.59</td>
</tr>
<tr>
<td>Bracketing Precision</td>
<td>60.05</td>
</tr>
<tr>
<td>Bracketing Recall</td>
<td>59.74</td>
</tr>
<tr>
<td>Number of Valid sentence</td>
<td>297</td>
</tr>
<tr>
<td>Number of Skip sentence</td>
<td>0</td>
</tr>
<tr>
<td>Number of Error sentence</td>
<td>13</td>
</tr>
<tr>
<td>Number of sentence</td>
<td>310</td>
</tr>
</tbody>
</table>

Probabilistic CFG Parser: Results with Stemming on Training Data
Probabilistic CFG Parser: Unseen Data; Unseen Data: Test Data = 20%

<table>
<thead>
<tr>
<th>Measurement</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 or less crossing</td>
<td>91.23</td>
</tr>
<tr>
<td>No crossing</td>
<td>73.68</td>
</tr>
<tr>
<td>Average crossing</td>
<td>69.53</td>
</tr>
<tr>
<td>Complete match</td>
<td>57.26</td>
</tr>
<tr>
<td>Bracketing Precision</td>
<td>53.45</td>
</tr>
<tr>
<td>Bracketing Recall</td>
<td>37.96</td>
</tr>
<tr>
<td>Number of Valid Sentence</td>
<td>57</td>
</tr>
<tr>
<td>Number of Skip Sentence</td>
<td>0</td>
</tr>
<tr>
<td>Number of Error Sentence</td>
<td>5</td>
</tr>
<tr>
<td>Number of Sentence</td>
<td>62</td>
</tr>
</tbody>
</table>
Lexicalized StatParser: Building up the parse tree
Lexicalized StatParser: Building up the parse tree

\[
\begin{align*}
(5) & \quad ( \rightarrow a \ \text{'
Na shyta', 'a'} | \text{'Das}
\text{'a, 'Mint}
\text{a, 'n'}} )^{\text{un}} \mathcal{P} \\
(4) & \quad \times ( \rightarrow a \ \text{'
Na shyta', 'a'} | \text{'Har a - phara', 'a'} | \text{'Bag a, n'}} )^{\text{un}} \mathcal{P} \\
(3) & \quad \times ( \rightarrow a \ \text{'
Na shyta', 'a'} | \text{'Na shyta', 'v', 'Na}
\text{ shyta', 'n'}} )^{\text{un}} \mathcal{P} \\
(2) & \quad \times ( \rightarrow a \ \text{'
Na shyta', 'a'} | \text{'Na}
\text{ shyta', 'v', 'Na}
\text{ shyta', 'n'}} )^{\text{un}} \mathcal{P} \\
(1) & \quad \times ( \rightarrow a \ \text{'
Na shyta', 'a'} | \text{'Na}
\text{ shyta', 'v', 'Na}
\text{ shyta', 'n'}} )^{\text{un}} \mathcal{P}
\end{align*}
\]
\( p^s_{\alpha\top} \) 
\( \times p^s_{(\alpha\top)\text{naShTa}} \) 
\( \times p^s_{(\alpha\top)} \) 
\( = p^s_{(\alpha\top)\text{naShTa}} \) 

<table>
<thead>
<tr>
<th>( p^s_{(\alpha\top)} )</th>
<th>( p^s_{(\alpha\top)\text{naShTa}} )</th>
<th>( p^s_{(\alpha\top)\text{naShTa}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Lexicalized StatParser: Start Probabilities
\[
(\rightarrow \text{, n\text{'}, n\text{\textdash}sh\text{\textmeta}, } \forall \text{, i\text{'}, n\text{\textdash}sh\text{\textmeta}, v | } \forall \text{, i\text{'}, n\text{\textdash}sh\text{\textmeta}, v}) \times (\rightarrow \text{, n\text{'}, n\text{\textdash}sh\text{\textmeta}, } \forall \text{, i\text{'}, n\text{\textdash}sh\text{\textmeta}, v | } \forall \text{, i\text{'}, n\text{\textdash}sh\text{\textmeta}, v})
\times (\rightarrow \text{, n\text{'}, n\text{\textdash}sh\text{\textmeta}, } \forall \text{, i\text{'}, n\text{\textdash}sh\text{\textmeta}, v | } \forall \text{, i\text{'}, n\text{\textdash}sh\text{\textmeta}, v})
= (\rightarrow \text{, n\text{\textdash}sh\text{\textmeta}, } \forall \text{, i\text{'}, n\text{\textdash}sh\text{\textmeta}, v})
\]

<table>
<thead>
<tr>
<th></th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>(l_t, v_a)</td>
<td>v_m \epsilon_w d</td>
<td>(l_t, v_a)</td>
<td>v_m \epsilon_w d</td>
<td>(l_t, v_a)</td>
<td>v_m \epsilon_w d</td>
</tr>
<tr>
<td>(l_t, v_a)</td>
<td>v_m \epsilon_w d</td>
<td>(l_t, v_a)</td>
<td>v_m \epsilon_w d</td>
<td>(l_t, v_a)</td>
<td>v_m \epsilon_w d</td>
</tr>
<tr>
<td>(d_t, l_t, v_a)</td>
<td>v_m \epsilon_w d</td>
<td>(d_t, l_t, v_a)</td>
<td>v_m \epsilon_w d</td>
<td>(d_t, l_t, v_a)</td>
<td>v_m \epsilon_w d</td>
</tr>
<tr>
<td>(d_t, l_t, v_a)</td>
<td>v_m \epsilon_w d</td>
<td>(d_t, l_t, v_a)</td>
<td>v_m \epsilon_w d</td>
<td>(d_t, l_t, v_a)</td>
<td>v_m \epsilon_w d</td>
</tr>
</tbody>
</table>

Lexicalled StatParser: Modification Probabilities
\[(\gamma, u | \text{tag} \gamma) \cdot P(\text{tag} \gamma | u) \cdot P(\text{tag} \gamma) \cdot P(k_1, bAga, n) = P(k_1) \cdot P(n | k_1) \cdot P(bAga | n, k_1)\]

\[
\begin{array}{|c|c|c|}
\hline
\text{tag} \gamma & \text{tag} \gamma & \text{tag} \gamma \\
\hline
(\gamma, u | \text{tag} \gamma) \cdot P(\text{tag} \gamma | u) & (\gamma, u | \text{tag} \gamma) \cdot P(\text{tag} \gamma | u) & (\gamma, u | \text{tag} \gamma) \cdot P(\text{tag} \gamma | u) \\
\hline
3 & 2 & 1 \\
\hline
\end{array}
\]

Lexicalized StatParser: Prior Probabilities
Contributionsoftheproject

- Cleaned and clause-bracketed Hindi Treebank
- Implementation of default rules listed in the Anncorrguidelines
- Conversion of Anncorra into dependency trees
- Trigram tagger/chunker (with evaluation)
- Lexicalized statistical parsing model (still in progress)
- Probabilistic CFG parser (with evaluation)

New NLP tools developed for Hindi:

Contributionsof the project
Future Work:

- Corpus development and bug fixes
- Fix remaining errors in annotated clause boundaries

Part-of-speech (POS) information can then be folded into the PCFG and lexicalized

- Combine POS information into the corpus
- Eliminate stemming from PCFG parser
- Combine part-of-speech information into the corpus

Current assumption: LWG gets 100% of the groups correct

Evaluate the local word grouper performance

 Corpus: Fix remaining errors in annotated clause boundaries

Future Work: Lexicalized Statistical Parser

• Write a paper describing the project

• Continue implementation and evaluation of lexicalized statistical parser

• Clean up the clause-butcheting annotation in the corpus

• Active learning experiments: Informativity sampling of data to be anno-

• Evalute the clause-bracketing parser based on the parser
Future Work: Active Learning

- Current learning model: fixed size of training and test data
- Annotation combined with learning
- Model we can explore (similar to ideas in online learning and active learning)
- Learning has no impact on the original annotated data

Annotation $\rightarrow$ Machine Learner $\rightarrow$ Annotation

Annotation $\leftarrow$ Machine Learner $\leftarrow$ Annotation
Future Work: Improving Existing Rule-based Parser for Hindi

- Transformation rules that modify Karaka charts based on tense-aspect-modality.
- Verb-argument dependencies: Demand (Karaka) charts.
- Dependency parser for Indian languages.
Future Work: Improving Existing Rule-based Parser for Hindi

Current Limitations of the Parser:

- Noun chunks that are larger than basal noun-phrases have to be handled.

- Local word grouper performs only on verb chunks.

- Insufficient lexical resources (≈ 19 Demand charts)

- Local word grouper performs only on verb chunks.

- Phrase chunker only on noun chunks that are larger than basal noun-phrases, which handle noun chunks sentences.

- Creates number of spurious analyses when handling multiple-clause sentences.
Future Work: Improving Existing Rule-based Parser for Hindi

Current directions for improvement:

- Heuristics for specifying clausal boundaries.
- Dealing with ellipsis, negation, etc.
- Learning the Karaka charts and the transformation rules from the annotated corpus.
- Using default Karaka charts for unknown verbs.
- Associating adjectives with the corresponding nouns.