A Statistical Parser for Hindi

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

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

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)

- Disambiguate existing rule-based parser (Papi's Parser) using the Tree-bank

- Active learning experiments: informative sampling of data to be annotated based on the parser

Initial Linguistic Resources

- Annotated corpus for Hindi, ”AnnCorra” prepared at LTRC, IIIT, Hyderabad
- Corpus description: extracts from Premchand’s novels.
- Corpus size: 338 sentences.
- Manually annotated corpus; marked for verb-argument relations.

Goals: Reconsidered

- Corpus Cleanup and Correction
- Default rules and Explicit Dependency Trees
- Various models of parsing based on the Treebank
  - Trigram tagger/chunker
  - Probabilistic CFG parser (stemming, no smoothing)
  - Fully lexicalized statistical parser (with smoothing)
  - Papi's parser and sentence units

Corpus Cleanup and Correction

- Problems in the Corpus:
  - Inconsistency in tags
  - Discrepancy in the use of tagsets.
  - Improper local word grouping.
- Cause of these problems: Inter-annotator consistency on labels.

Corpus Cleanup and Correction

- Solution: Annotators who were part of the team manually corrected the following problems
  - Inconsistency of tags resolved.
  - Resolved the discrepancies in the tagsets
  - Problems of local word grouping resolved.

- Explicitly marked the clause boundaries to disambiguate long complex sentences without punctuation in the corpus.

Default rules and Explicit Dependency Trees

- Raw corpus:

```
{ [dasa miniTa_meM]/k7.1 [harA-bharA bAga]/k1
            naShTa_ho_gayA::v }
```

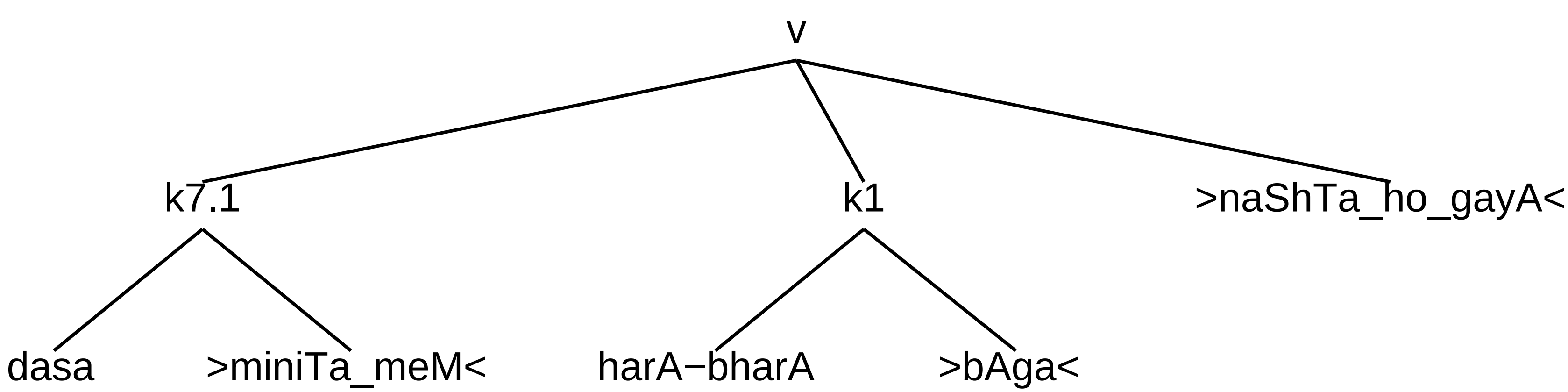
- Explicit dependencies are not marked

- Default rules are listed in the guidelines

- Evaluated the default rules and built a program to convert original corpus into explicit dependency trees

Default rules and Explicit Dependency Trees

```
{ [dasa miniTa_meM]/k7.1 [harA-bharA bAga]/k1
            naShTa_ho_gayA::v }
```



Default rules and Explicit Dependency Trees

- Default rules could not handle 24 out of 334 sentences

- ad-hoc defaults for multiple sentence units within a single sentence (added yo as parent of all clauses)

Trigram Tagger/Chunker

- Input:

```
{[tahasIla madarasA barA.Nva_ke]/6
 [prathamAdhyApaka muMshI bhavAnIsahAya_ko]/k1
 bAgavAnI_kA/6
 kuchha::adv
 vyasana_thA::v}
```

- Converted to representation for tagger:

```
tahasIla//adj//cb
madarasA//adj//cb
barA.Nva_ke//6//cb
prathamAdhyApaka//adj//cb
muMshI//adj//cb
bhavAnIsahAya_ko//k1//cb
bAgavAnI_kA//6//co
kuchha//adv//co
vyasana_thA//v//co
```

Trigram Tagger/Chunker

- Bootstrapped using existing supertagger code
  `http://www.cis.upenn.edu/~xtag/`
- 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%

Probabilistic CFG Parser

- Extracted context-free rules from the Treebank
- Estimated probabilities for each rule using counts from the Treebank
- Used PCFG parser to compute the best derivation for a given sentence
- Used some existing code written earlier for prob CKY parsing
  `http://www.cis.upenn.edu/~anoop/distrib/ckycfg/`

Probabilistic CFG Parser: Results on Training Data

Time	=	1min 27secs
Number of sentence	=	310
Number of Error sentence	=	13
Number of Skip sentence	=	0
Number of Valid sentence	=	297
Bracketing Recall	=	76.94
Bracketing Precision	=	86.29
Complete match	=	48.82
Average crossing	=	0.12
No crossing	=	91.25
2 or less crossing	=	99.33

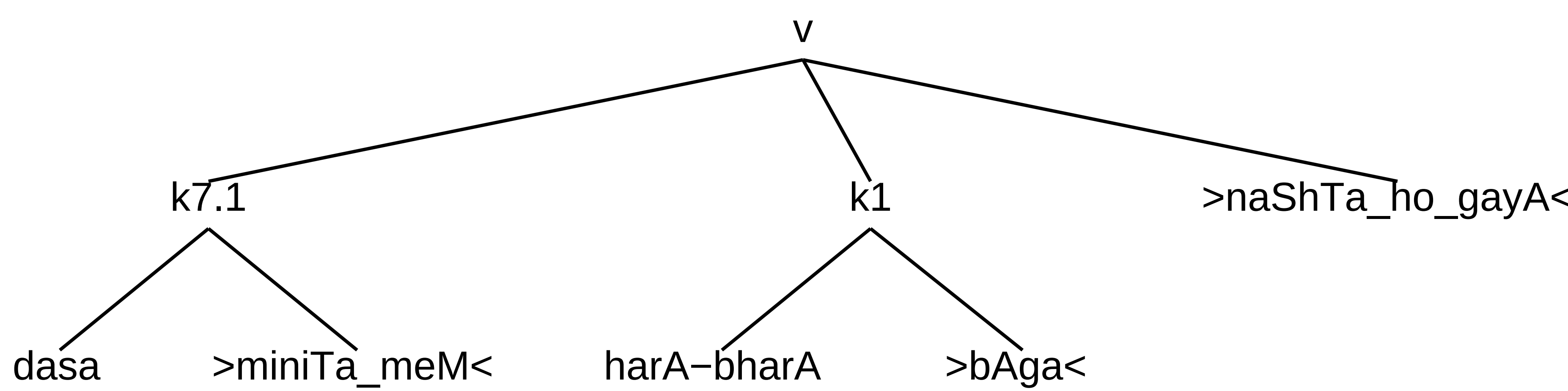
Probabilistic CFG Parser: Results with Stemming on Training Data

Number of sentence	=	310
Number of Error sentence	=	13
Number of Skip sentence	=	0
Number of Valid sentence	=	297
Bracketing Recall	=	59.74
Bracketing Precision	=	60.05
Complete match	=	25.59
Average crossing	=	0.58
No crossing	=	66.33
2 or less crossing	=	94.95

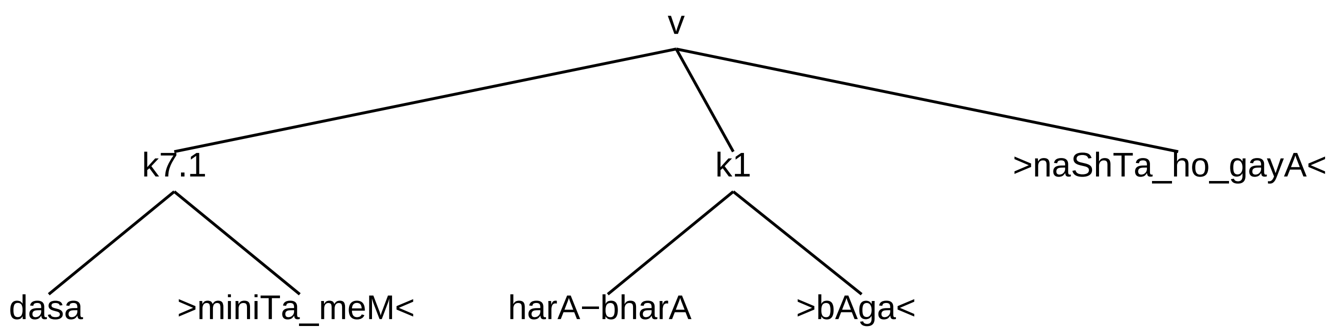
Probabilistic CFG Parser: Unseen Data; Test Data = 20%

Number of sentence	=	62
Number of Error sentence	=	5
Number of Skip sentence	=	0
Number of Valid sentence	=	57
Bracketing Recall	=	37.96
Bracketing Precision	=	53.45
Complete match	=	5.26
Average crossing	=	0.53
No crossing	=	73.68
2 or less crossing	=	91.23

Lexicalized StatParser: Building up the parse tree



Lexicalized StatParser: Building up the parse tree



$$P_s(v, \texttt{naShTa}, \texttt{v} \mid \textsf{TOP}) \times \tag{1}$$

$$P_m(k1, \texttt{bAga}, \texttt{n} \mid v, \texttt{naShTa}, \texttt{v}, \leftarrow) \times \tag{2}$$

$$P_m(k7.1, \texttt{miniTa}, \texttt{n} \mid v, \texttt{naShTa}, \texttt{v}, \leftarrow) \times \tag{3}$$

$$P_m(\cdot, \texttt{harA} - \texttt{bharA}, \texttt{a} \mid k7.1, \texttt{bAga}, \texttt{n}, \leftarrow) \times \tag{4}$$

$$P_m(\cdot, \texttt{dasa}, \texttt{a} \mid k1, \texttt{miniTa}, \texttt{n}, \leftarrow) \tag{5}$$

Lexicalized StatParser: Start Probabilities

$P_s(\alpha \mid \text{TOP})$	1	2	3
0	$P_{s1}(t_\alpha \mid \text{TOP})$	$P_{s2}(w_\alpha \mid t_\alpha, \text{TOP})$	$P_{s3}(\tau_\alpha \mid t_\alpha, w_\alpha, \text{TOP})$
1	$P_{s1}(t_\alpha)$	$P_{s2}(w_\alpha \mid t_\alpha)$	$P_{s3}(\tau_\alpha \mid t_\alpha, w_\alpha)$
2			$P_{s3}(\tau_\alpha \mid t_\alpha)$

$$
\begin{aligned}
P_s(v, \texttt{naShTa}, \texttt{v} \mid \text{TOP}) = \\
& P_{s1}(\texttt{v} \mid \text{TOP}) \times \\
& P_{s2}(\texttt{naShTa} \mid \texttt{v}, \text{TOP}) \times \\
& P_{s3}(v \mid \texttt{naShTa}, \texttt{v}, \text{TOP})
\end{aligned}
$$

Lexicalized StatParser: Modification Probabilities

$P_m(\alpha \mid \eta)$	1	2	3
0	$P_{m1}(\tau_\alpha \mid \tau_\eta, t_\eta, w_\eta, p)$	$P_{m2}(t_\alpha \mid \tau_\alpha, t_\eta, w_\eta, p)$	$P_{m3}(w_\alpha \mid \tau_\alpha, t_\alpha, t_\eta, w_\eta, p)$
1	$P_{m1}(\tau_\alpha \mid \tau_\eta, t_\eta, p)$	$P_{m2}(t_\alpha \mid \tau_\alpha, t_\eta, p)$	$P_{m3}(w_\alpha \mid \tau_\alpha, t_\alpha, t_\eta, p)$
2	$P_{m1}(\tau_\alpha \mid \tau_\eta, t_\eta)$	$P_{m2}(t_\alpha \mid \tau_\alpha, t_\eta)$	$P_{m3}(w_\alpha \mid \tau_\alpha, t_\alpha, t_\eta)$
3	$P_{m1}(\tau_\alpha \mid \tau_\eta)$	$P_{m2}(t_\alpha \mid t_\eta)$	$P_{m3}(w_\alpha \mid t_\alpha, t_\eta)$

$$
\begin{aligned}
P_m(k1, \texttt{bAga}, \texttt{n} \mid v, \texttt{naShTa}, \texttt{v}, \leftarrow) = \\
P_{m1}(k1 \mid v, \texttt{naShTa}, \texttt{v}, \leftarrow) \times \\
P_{m2}(\texttt{n} \mid k1, v, \texttt{naShTa}, \texttt{v}, \leftarrow) \times \\
P_{m3}(\texttt{bAga} \mid \texttt{n}, k1, v, \texttt{naShTa}, \texttt{v}, \leftarrow)
\end{aligned}
$$

Lexicalized StatParser: Prior Probabilities

$P_{pr}(\alpha)$	1	2	3
0	$P_{pr1}(t_\alpha)$	$P_{pr2}(w_\alpha \mid t_\alpha)$	$P_{pr3}(\tau_\alpha \mid t_\alpha, w_\alpha)$
1			$P_{pr3}(\tau_\alpha \mid t_\alpha)$

$$
\begin{aligned}
P_{pr}(k1, \texttt{bAga}, \texttt{n}) = \\
P_{pr1}(k1) \times \\
P_{pr2}(\texttt{n} \mid k1) \times \\
P_{pr3}(\texttt{bAga} \mid \texttt{n}, k1)
\end{aligned}
$$

Contributions of the project

- Cleaned and clause-bracketed Hindi Treebank
- Implementation of default rules listed in the AnnCorra guidelines
  Conversion of AnnCorra into dependency trees
- New NLP tools developed for Hindi:
  - Trigram tagger/chunker (with evaluation)
  - Probabilistic CFG parser (with evaluation)
  - Lexicalized statistical parsing model (still in progress)

Future Work: Corpus development and Bugfixes

- Corpus: fix remaining errors in annotated clause boundaries ({, })
- Evaluate the local word grouper performance
  Current assumption: LWG gets 100% of the groups correct
- Combine part-of-speech information into the corpus
- Part-of-speech info can then be folded into the PCFG and Lexicalized Parser
- Eliminate stemming from PCFG parser

Future Work: Lexicalized Statistical Parser

- Clean up the clause-bracketing annotation in the corpus
- Continue implementation and evaluation of lexicalized statistical parser
- Active learning experiments: informative sampling of data to be annotated based on the parser
- Write a paper describing the project

Future Work: Active Learning

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

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

- Annotation combined with learning

Future Work: Improving Existing Rule-based Parser for Hindi

- Dependency parser for Indian languages.
- Verb-argument dependencies: Demand (Karaka) charts.
- Transformation rules that modify Karaka charts based on tense-aspect-modality.

Future Work: Improving Existing Rule-based Parser for Hindi

- Current Limitations of the parser.
  - Creates number of spurious analyses when handling multiple-clause sentences.
  - Insufficient lexical resources ($\approx$ 119 Demand charts)
  - Local word grouper performs only on verb chunks.
    Noun chunks that are larger than basal noun-phrases have to be handled.

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.