Introduction to Data-Driven Dependency Parsing

Introductory Course, ESSLLI 2007

Ryan McDonald\textsuperscript{1}  Joakim Nivre\textsuperscript{2}

\textsuperscript{1}Google Inc., New York, USA
E-mail: ryanmcd@google.com

\textsuperscript{2}Uppsala University and Växjö University, Sweden
E-mail: nivre@msi.vxu.se
Overview of the Course

- Dependency parsing (Joakim)
- Machine learning methods (Ryan)
- Transition-based models (Joakim)
- Graph-based models (Ryan)
- **Loose ends** (Joakim, Ryan):
  - Other approaches
  - Empirical results
  - Available software
Other Approaches – Overview

- Graph-based methods – new developments
- Transition-based methods – new developments
- Ensemble methods
- Constraint-based methods
- Phrase structure parsing
- Unsupervised parsing
Graph-based Methods

- Last lecture we discussed arc-factored models
- Models are inherently local
  - Local feature scope
  - Local structural constraints
- This is a strong assumption!!
- Question: how do we incorporate non-local information?
Other Approaches

Integer Linear Programming

- Often intractable inference problems can be written as Integer Linear Programming (ILP) problems
- ILP’s are optimization problems with linear objectives and constraints
- Non-projective parsing with global constraints can be written as an ILP [Riedel and Clarke 2006]
- ILP’s are still NP-hard, but have well known branch-and-bound solutions
- First, let’s define a set of **binary variables**
  - $a^k_{ij} \in \{0, 1\}$ is 1 if arc $(i, j, k)$ is in the dependency graph
  - $a$ is the vector of all variables $a^k_{ij}$
Integer Linear Programming

- We can define the arc-factored parsing problem as the following objective function

  \[
  \arg \max_a \sum_{i,j,k} \log w^k_{ij} \cdot a^k_{ij}
  \]

  such that:
  \[
  \forall j > 0, \sum_{i,k} a^k_{ij} = 1 \quad \text{(single head)}
  \]
  \[
  \sum_{i,k} a^k_{i0} = 0 \quad \text{($w_0$ is root)}
  \]
  \[
  \forall \text{ cycles } C, \sum_{(i,j,k) \in C} a^k_{ij} \leq |C| - 1 \quad \text{(no cycles)}
  \]

- This is an ILP!!
- Linear objective
- Linear constraints over integer variables
Other Approaches

**Integer Linear Programming**

- [Riedel and Clarke 2006] showed that this formulation allows for non-local constraints.

- e.g., a verb can only have a single subject

\[ \forall w_i \text{ that are verbs} \]

\[ \sum_j a_{ij}^{sbj} \leq 1 \]

- This is non-local since we are forcing constraints on all the modifiers of \( w_i \).

- [Riedel and Clarke 2006] also includes constraints on co-ordination as well as projectivity if desired.

- Is this still data-driven?
Sampling Methods

- Used for dependency parsing with global features by [Nakagawa 2007]

- Define a conditional log-linear probability model

\[ P(G|x) = \frac{1}{Z_x} e^{w \cdot f(G)} \]

- \( Z_x = \sum_{G'} e^{w \cdot f(G')} \)

- \( f(G) \) is a global feature map – can contain global features of dependency graph

- i.e., does not necessarily factor by the arcs
Other Approaches

Sampling Methods

- arg max_G P(G|x) cannot be solved efficiently
- Assume we have N samples from the distribution P(G|x)
  - Can be found efficiently with Gibbs sampling
  - Call them G_1, ..., G_N
- We want **marginal distribution** of the arc (i, j, k), \( \mu_{ij}^k \)

\[
\mu_{ij}^k \approx \sum_{t=1}^{N} P(G_t | x) \mathbb{1}[(i, j, k) \in G_t] \approx \frac{1}{N} \sum_{t=1}^{N} \mathbb{1}[(i, j, k) \in G_t]
\]

- Since N should be a manageable size, this can be calculated
- Set arc weights \( w_{ij}^k = \mu_{ij}^k \) and **find the MST**
- w is found using Monte Carlo sampling
Transition-based Methods

- Transition-based models may suffer from
  - Error propagation because of greedy inference,
  - Label bias because of local training.
- Recent developments seek to remedy this:
  - Beam search instead of greedy best-first search
    [Johansson and Nugues 2006, Duan et al. 2007]
  - Globally trained probabilistic model
    [Johansson and Nugues 2007, Titov and Henderson 2007]
Generative Model [Titov and Henderson 2007]

- Probabilistic model defined in terms of transitions:

\[ P(G_{cm}) = P(c_0, c_1, \ldots, c_m) = \prod_{i=1}^{m} P(c_i|c_0, \ldots, c_{i-1}) \]

- Similar to HMMs

- Transition system of [Nivre 2003] with two modifications:
  - Splits Right-Arc into Right-Arc' and Shift
  - Adds transitions for generating words (generative model)

- \( P(c_i|c_0, c_1, \ldots, c_{i-1}) \) modeled by neural network approximating Incremental Sigmoid Belief Network (ISBN)
  - Belief network hidden layer acts as feature selection algorithm

- Parsing with heuristic beam search
Ensemble Methods

- Input: Sentence $x = w_0, w_1, \ldots, w_n$
- Input: Output of $N$ different parsers for $x$, $G_1, G_2, \ldots, G_N$
- Output: A single dependency graph $G$ for sentence $x$

Question
How do we combine the the parsers and their outputs to create a new and better parse for sentence $x$?

- Assumption: The $N$ parsers make different mistakes
- Assumption: All of the $N$ parsers are relatively strong
Ensemble Methods [Sagae and Lavie 2006]

- Simple but elegant solution
- Use arc-factored graph-based models
- Set arc weights equal to the number of parsers that predicted that arc
  \[ w_{ij}^k = e^{\sum_i \alpha_i \times 1[(i,j,k) \in G_i]} \]
- \( \alpha_i \) usually equals 1, but can be modified if prior knowledge exists
- **Solution**: Find MST for \( G_x \) with the above weights
- The resulting graph has on average the arcs that were preferred by most systems
Ensemble Methods [Sagae and Lavie 2006]

Example: ensemble of parsers from this year's CoNLL shared-task
Constraint-based Methods

- Statistical constraint dependency parsing in two steps [Wang and Harper 2004]:
  1. Supertagging using a trigram Hidden Markov Model to assign the top $n$-best constraints to an input sentence $x$.
  2. Stack-based, best-first search to build the most probable dependency graph given the constraints.

- Anytime transformation-based parsing with constraints [Foth and Menzel 2006]:
  1. Use data-driven transition-based parser to derive initial dependency graph.
  2. Use graph transformations to improve score relative to weighted constraints.
Phrase Structure Parsing

- Phrase structure parsers used for dependency parsing:
  1. Transform training data from dependencies to phrase structure
  2. Train a parser on the transformed structures
  3. Parse new sentences with the trained parser
  4. Transform parser output from phrase structure to dependencies

- Example:
  - Parsing Czech with the Collins and Charniak parsers
    [Collins et al. 1999, Hall and Novák 2005]

- Note:
  - Both of these parsers internally extract dependencies from phrase structures.
Unsupervised Parsing

- Often we do not have a large corpus with annotated dependency graphs
- Can we still learn to parse dependencies from unlabeled data?
- There has been much research along these lines lately
  - Lexical attraction [Yuret 1998]
  - Grammatical bi-grams [Paskin 2001]
  - Top-down generative models [Klein and Manning 2004]
  - Contrastive estimation [Smith and Eisner 2005]
  - Non-projective examples [McDonald and Satta 2007]
Empirical Results – Overview

- Evaluation metrics
- Benchmarks:
  - Penn Treebank (Wall Street Journal)
  - Prague Dependency Treebank
- CoNLL 2006 shared task [Buchholz and Marsi 2006]:
  - 19 parsers for 13 languages
  - Error analysis for the two top systems
    [McDonald and Nivre 2007]
- CoNLL 2007 shared task [Nivre et al. 2007]:
  - 23 parsers for 10 languages
  - Domain adaptation for English
Evaluation Metrics

- **Per token:**
  - Labeled attachment score (LAS):
    - Percentage of tokens with correct head and label
  - Unlabeled attachment score (UAS):
    - Percentage of tokens with correct head
  - Label accuracy (LA):
    - Percentage of tokens with correct label

- **Per sentence:**
  - Labeled complete match (LCM):
    - Percentage of sentences with correct labeled graph
  - Unlabeled complete match (UCM):
    - Percentage of sentences with correct unlabeled graph
Empirical Results

State of the Art – English

- Penn Treebank (WSJ) converted to dependency graphs
  - Transition-based parsers
    - Yamada and Matsumoto 2003, Isozaki et al. 2004
  - Graph-based parsers
    - McDonald et al. 2005a, McDonald and Pereira 2006
  - Ensemble parser
    - Sagae and Lavie 2006, McDonald 2006
  - Phrase structure parsers
    - Collins 1999, Charniak 2000

<table>
<thead>
<tr>
<th>Parser</th>
<th>UAS</th>
<th>UCM</th>
</tr>
</thead>
<tbody>
<tr>
<td>McDonald</td>
<td>93.2</td>
<td>47.1</td>
</tr>
<tr>
<td>Sagae and Lavie</td>
<td>92.7</td>
<td>–</td>
</tr>
<tr>
<td>Charniak</td>
<td>92.2</td>
<td>45.2</td>
</tr>
<tr>
<td>Collins</td>
<td>91.7</td>
<td>43.3</td>
</tr>
<tr>
<td>McDonald and Pereira</td>
<td>91.5</td>
<td>42.1</td>
</tr>
<tr>
<td>Isozaki et al.</td>
<td>91.4</td>
<td>40.7</td>
</tr>
<tr>
<td>McDonald et al.</td>
<td>91.0</td>
<td>37.5</td>
</tr>
<tr>
<td>Yamada and Matsumoto</td>
<td>90.4</td>
<td>38.4</td>
</tr>
</tbody>
</table>
State of the Art – Czech

- Prague Dependency Treebank (PDT)
  - Pseudo-projective transition-based parser [Nilsson et al. 2006]
  - Non-projective spanning tree parser [McDonald et al. 2005b]
  - Approximate second-order spanning tree parser [McDonald and Pereira 2006]
  - Phrase structure projective (Charniak, Collins)
  - Phrase structure (Charniak) + corrective modeling [Hall and Novák 2005]

<table>
<thead>
<tr>
<th>Parser</th>
<th>UAS</th>
<th>UCM</th>
</tr>
</thead>
<tbody>
<tr>
<td>McDonald and Pereira</td>
<td>85.2</td>
<td>35.9</td>
</tr>
<tr>
<td>Hall and Novák</td>
<td>85.1</td>
<td>—</td>
</tr>
<tr>
<td>Nilsson et al.</td>
<td>84.6</td>
<td>37.7</td>
</tr>
<tr>
<td>McDonald et al.</td>
<td>84.4</td>
<td>32.3</td>
</tr>
<tr>
<td>Charniak</td>
<td>84.4</td>
<td>—</td>
</tr>
<tr>
<td>Collins</td>
<td>81.8</td>
<td>—</td>
</tr>
</tbody>
</table>
CoNLL Shared Task 2006

- Multilingual dependency parsing:
  - Train a single parser on data from thirteen languages
  - Gold standard annotation (postags, lemmas, etc.)
  - Main evaluation metric: LAS

- Results:
  - 19 systems, 17 described in [Buchholz and Marsi 2006]
  - Considerable variation across languages (top scores):
    - Japanese: 91.7%
    - Turkish: 65.7%
  - Best systems:
    - MSTParser (graph-based) [McDonald et al. 2006]
    - MaltParser (transition-based) [Nivre et al. 2006]
## MSTParser and MaltParser

<table>
<thead>
<tr>
<th>Language</th>
<th>MST</th>
<th>Malt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arabic</td>
<td>66.91</td>
<td>66.71</td>
</tr>
<tr>
<td>Bulgarian</td>
<td>87.57</td>
<td>87.41</td>
</tr>
<tr>
<td>Chinese</td>
<td>85.90</td>
<td>86.92</td>
</tr>
<tr>
<td>Czech</td>
<td>80.18</td>
<td>78.42</td>
</tr>
<tr>
<td>Danish</td>
<td>84.79</td>
<td>84.77</td>
</tr>
<tr>
<td>Dutch</td>
<td>79.19</td>
<td>78.59</td>
</tr>
<tr>
<td>German</td>
<td>87.34</td>
<td>85.82</td>
</tr>
<tr>
<td>Japanese</td>
<td>90.71</td>
<td>91.65</td>
</tr>
<tr>
<td>Portuguese</td>
<td>86.82</td>
<td>87.60</td>
</tr>
<tr>
<td>Slovene</td>
<td>73.44</td>
<td>70.30</td>
</tr>
<tr>
<td>Spanish</td>
<td>82.25</td>
<td>81.29</td>
</tr>
<tr>
<td>Swedish</td>
<td>82.55</td>
<td>84.58</td>
</tr>
<tr>
<td>Turkish</td>
<td>63.19</td>
<td>65.68</td>
</tr>
<tr>
<td><strong>Overall</strong></td>
<td><strong>80.83</strong></td>
<td><strong>80.75</strong></td>
</tr>
</tbody>
</table>
Comparing the Models

▸ Inference:
  ▹ Exhaustive (MSTParser)
  ▹ Greedy (MaltParser)

▸ Training:
  ▹ Global structure learning (MSTParser)
  ▹ Local decision learning (MaltParser)

▸ Features:
  ▹ Local features (MSTParser)
  ▹ Rich decision history (MaltParser)

▸ Fundamental trade-off:
  ▹ Global learning and inference vs. rich feature space
Error Analysis

Aim:
- Relate parsing errors to linguistic and structural properties of the input and predicted/gold standard dependency graphs

Three types of factors:
- Length factors: sentence length, dependency length
- Graph factors: tree depth, branching factor, non-projectivity
- Linguistic factors: part of speech, dependency type

Statistics:
- Labeled accuracy, precision and recall
- Computed over the test sets for all 13 languages
MaltParser is more accurate than MSTParser for short sentences (1–10 words) but its performance degrades more with increasing sentence length.
MaltParser is more precise than MSTParser for short dependencies (1–3 words) but its performance degrades drastically with increasing dependency length (> 10 words).

MSTParser has more or less constant precision for dependencies longer than 3 words.

Recall is very similar across systems.
**Tree Depth (Distance to Root)**

MSTParser is much more precise than MaltParser for dependents of the root and has roughly constant precision for depth $> 1$, while MaltParser’s precision improves with increasing depth (up to 7 arcs).

Recall is very similar across systems.
Degrees of Non-Projectivity

- Degree of a dependency arc \((i, j, k)\) = The number of words in the span \(\min(i, j), \ldots, \max(i, j)\) that are not descendants of \(i\) and have their head outside the span.
- MaltParser has slightly higher precision, and MSTParser slightly higher recall, for non-projective arcs (degree > 0).
- No system predicts arcs with a higher degree than 2.
Part of Speech

- MSTParser is more accurate for verbs, adjectives, adverbs, adpositions, and conjunctions.
- MaltParser is more accurate for nouns and pronouns.
MSTParser has higher precision (and recall) for roots.

MSTParser has higher recall (and precision) for subjects.
Discussion

Many of the results are indicative of the fundamental trade-off: global learning/inference versus rich features.

Global inference improves decisions for long sentences and those near the top of graphs.

Rich features improve decisions for short sentences and those near the leaves of the graphs.

Important question:

How do we use this to improve parser performance?

Oracle Experiments:

- Graph-based selection: 81% → 85%
- Arc-based selection [Sagae and Lavie 2006]: 81% → 87%
CoNLL Shared Task 2007

Two tracks:
- Multilingual dependency parsing (10 languages)
- Domain adaptation (English)

Results (multilingual track):
- 28 systems, 23 described in [Nivre et al. 2007]
- A little less variation across languages (top scores):
  - English: 89.6%
  - Greek: 76.3%
- Best systems:
  - Ensemble systems [Hall et al. 2007, Sagae and Tsujii 2007]
  - Graph-based systems with global features [Nakagawa 2007, Carreras 2007]
  - Transition-based systems with global training [Titov and Henderson 2007]
Available Software – Overview

- Dependency Parsing Wiki:
  - http:depparse.uvt.nl/depparse-wiki/
- Parsers:
  - Trainable data-driven parsers
  - Parsers for specific languages (grammar-based)
- Other tools:
  - Pseudo-projective parsing
  - Evaluation software
  - Constituency-to-dependency conversion
- Data sets:
  - Dependency treebanks
  - Other treebanks with dependency conversions
Trainable Parsers

- Jason Eisner’s **probabilistic dependency parser**
  - Based on bilexical grammar
  - Contact Jason Eisner: jason@cs.jhu.edu
  - Written in LISP

- Ryan McDonald’s **MSTParser**
  - Graph-based spanning tree parsers with online learning
  - URL: http://sourceforge.net/projects/mstparser
  - Written in Java
Trainable Parsers (2)

- Joakim Nivre’s **MaltParser**
  - Transition-based parsers with MBL and SVM
  - URL: http://w3.msi.vxu.se/~nivre/research/MaltParser.html
  - Executable versions are available for Solaris, Linux, Windows, and MacOS (open source version in Java planned for fall 2007)

- Ivan Titov’s **ISBN Dependency Parser**
  - Incremental Sigmoid Belief Network Dependency Parser
  - Transition-based inference
  - URL: http://cui.unige.ch/~titov/idp/
  - Written in C
Parsers for Specific Languages

▶ Dekang Lin’s Minipar
  ▶ Principle-based parser
  ▶ Grammar for English
  ▶ URL: http://www.cs.ualberta.ca/~lindek/minipar.htm
  ▶ Executable versions for Linux, Solaris, and Windows

▶ Wolfgang Menzel’s CDG Parser:
  ▶ Weighted constraint dependency parser
  ▶ Grammar for German, (English under construction)
  ▶ Online demo: http://nats-www.informatik.uni-hamburg.de/Papa/ParserDemo
  ▶ Download: http://nats-www.informatik.uni-hamburg.de/download
Parsers for Specific Languages (2)

- **Taku Kudo’s CaboCha**
  - Based on algorithms of [Kudo and Matsumoto 2002], uses SVMs
  - Web page in Japanese

- **Gerold Schneider’s Pro3Gres**
  - Probability-based dependency parser
  - Grammar for English
  - Written in PROLOG

- **Daniel Sleator’s & Davy Temperley’s Link Grammar Parser**
  - Undirected links between words
  - Grammar for English
  - URL: [http://www.link.cs.cmu.edu/link/](http://www.link.cs.cmu.edu/link/)
Other Tools

- **Pseudo-projective parsing:**
  - Software based on [Nivre and Nilsson 2005]
  - [http://w3.msi.vxu.se/~nivre/research/proj/0.2/doc/Proj.html](http://w3.msi.vxu.se/~nivre/research/proj/0.2/doc/Proj.html)

- **Evaluation software:**
  - CoNLL shared tasks:
    - [http://nextens.uvt.nl/~conll/software.html](http://nextens.uvt.nl/~conll/software.html)
    - [http://deppare.uvt.nl/depparse-wiki/SoftwarePage](http://deppare.uvt.nl/depparse-wiki/SoftwarePage)

- **Treebank conversion software:**
  - CoNLL 2006 shared task treebanks:
    - [http://depparse.uvt.nl/depparse-wiki/SoftwarePage](http://depparse.uvt.nl/depparse-wiki/SoftwarePage)
  - Penn Treebank:
    - [http://w3.msi.vxu.se/~nivre/research/Penn2Malt.html](http://w3.msi.vxu.se/~nivre/research/Penn2Malt.html)
    - [http://nlp.cs.lth.se/pennconverter/](http://nlp.cs.lth.se/pennconverter/)
Dependency Treebanks

- Arabic: Prague Arabic Dependency Treebank
- Basque: Eus3LB
- Czech: Prague Dependency Treebank
- Danish: Danish Dependency Treebank
- Greek: Greek Dependency Treebank
- Portuguese: Bosque: Floresta sintá(c)tica
- Slovene: Slovene Dependency Treebank
- Turkish: METU-Sabanci Turkish Treebank
Other Treebanks

- Bulgarian: BulTreebank
- Catalan: CESS-ECE
- Chinese: Penn Chinese Treebank, Sinica Treebank
- Dutch: Alpino Treebank for Dutch
- English: Penn Treebank
- German: TIGER/NEGRA, TüBa-D/Z
- Hungarian: Szeged Treebank
- Italian: Italian Syntactic-Semantic Treebank
- Japanese: TüBa-J/S
- Spanish: Cast3LB
- Swedish: Talbanken05
Summary

- State of the art in data-driven dependency parsing:
  - Transition-based models
  - Graph-based models
  - New developments (often) targeting the weaknesses of standard models

- Empirical results:
  - CoNLL shared tasks: Dependency parsing results for some twenty languages
  - Many (different) systems achieve similar accuracy, but performance varies across languages

- Available resources: Try them out!
Dependency Treebanks (1)

- Prague Arabic Dependency Treebank
  - ca. 100,000 words
  - Available from LDC, license fee
    (CoNLL-X shared task data, catalogue number LDC2006E01)
  - URL: http://ufal.mff.cuni.cz/padt/

- Eus3LB
  - ca. 50,000 words
  - Restricted availability
  - URL: http://ixa.si.ehu.es/lxa/lkerlerroak
Dependency Treebanks (2)

- **Prague Dependency Treebank**
  - 1.5 million words
  - 3 layers of annotation: morphological, syntactical, tectogrammatical
  - Available from LDC, license fee
    (CoNLL-X shared task data, catalogue number LDC2006E02)
  - URL: http://ufal.mff.cuni.cz/pdt2.0/

- **Danish Dependency Treebank**
  - ca. 5 500 trees
  - Annotation based on Discontinuous Grammar [Kromann 2003]
  - Freely downloadable
  - URL: http://www.id.cbs.dk/~mtk/treebank/
Dependency Treebanks (3)

- **Greek Dependency Treebank**
  - ca. 70,000 words
  - Restricted availability.
  - Contact ILSP, Athens, Greece.

- **Bosque, Floresta sintá(c)tica**
  - ca. 10,000 trees
  - Freely downloadable
  - URL: http://acdc.linguateca.pt/treebank/info_floresta_English.html
Dependency Treebanks (4)

- Slovene Dependency Treebank
  - ca. 30 000 words
  - Freely downloadable
  - URL: http://nl.ijs.si/sdt/

- METU-Sabanci Turkish Treebank
  - ca. 7 000 trees
  - Freely available, license agreement
  - URL: http://www.ii.metu.edu.tr/~corpus/treebank.html
Other Treebanks (1)

- BulTreebank
  - ca. 14,000 sentences
  - URL: http://www.bultreebank.org/
  - Dependency version available from Kiril Simov (kivs@bultreebank.org)

- CESS-ECE
  - ca. 500,000 words
  - Freely available for research
  - URL: http://www.lsi.upc.edu/~mbertran/cess-ece2/
  - Dependency version available from Toni Marti
Other Treebanks (2)

▶ Penn Chinese Treebank
  ▶ ca. 4 000 sentences
  ▶ Available from LDC, license fee
  ▶ URL: http://www.cis.upenn.edu/~chinese/ctb.html
  ▶ For conversion with arc labels: Penn2Malt:
    http://w3.msi.vxu.se/~nivre/research/Penn2Malt.html

▶ Sinica Treebank
  ▶ ca. 61 000 sentences
  ▶ Available Academia Sinica, license fee
  ▶ URL: http://godel.iis.sinica.edu.tw/CKIP/engversion/treebank.htm
  ▶ Dependency version available from Academia Sinica
Other Treebanks (3)

- **Alpino Treebank for Dutch**
  - ca. 150,000 words
  - Freely downloadable
  - URL: [http://www.let.rug.nl/vannoord/trees/](http://www.let.rug.nl/vannoord/trees/)
  - Dependency version downloadable at [http://nextens.uvt.nl/~conll/free_data.html](http://nextens.uvt.nl/~conll/free_data.html)

- **Penn Treebank**
  - ca. 1 million words
  - Available from LDC, license fee
  - URL: [http://www.cis.upenn.edu/~treebank/home.html](http://www.cis.upenn.edu/~treebank/home.html)
  - Conversion to labeled dependencies: Penn2Malt, pennconverter (see above)
Other Treebanks (4)

- **TIGER/NEGRA**
  - ca. 50 000/20 000 sentences
  - Freely available, license agreement
  - TIGER URL: [http://www.ims.uni-stuttgart.de/projekte/TIGER/TIGERCorpus/](http://www.ims.uni-stuttgart.de/projekte/TIGER/TIGERCorpus/)
  - NEGRA URL: [http://www.coli.uni-saarland.de/projects/sfb378/negra-corpus/](http://www.coli.uni-saarland.de/projects/sfb378/negra-corpus/)
  - Dependency version of TIGER is included in release

- **TüBa-D/Z**
  - ca. 22 000 sentences
  - Freely available, license agreement
  - URL: [http://www.sfs.uni-tuebingen.de/en_tuebadz.shtml](http://www.sfs.uni-tuebingen.de/en_tuebadz.shtml)
  - Dependency version available from SfS Tübingen
Other Treebanks (5)

- **Szeged Treebank**
  - ca. 82 000 sentences (1.2 million words)
  - Freely available, license agreement
  - URL: http://www.inf.u-szeged.hu/hlt
  - Subset in dependency format (6 000 sentences)

- **Italian Syntactic-Semantic Treebank**
  - ca. 300 000 words
  - Available through ELDA
  - URL: http://www.ilc.cnr.it/viewpage.php/sez=ricerca/id=874/vers=ita
  - Dependency version available
Other Treebanks (6)

- **Cast3LB**
  - ca. 18 000 sentences
  - URL: [http://www.dlsi.ua.es/projectes/3lb/index_en.html](http://www.dlsi.ua.es/projectes/3lb/index_en.html)
  - Dependency version available from Toni Martí (amarti@ub.edu)

- **Talbanken05 (Swedish)**
  - ca. 300 000 words
  - Freely downloadable
  - URL: [http://w3.msi.vxu.se/~nivre/research/Talbanken05.html](http://w3.msi.vxu.se/~nivre/research/Talbanken05.html)
  - Dependency version also available
References and Further Reading


Kilian A. Foth and Wolfgang Menzel. 2006. 


References and Further Reading


Discovery of linguistic relations using lexical attraction. Ph.D. thesis, MIT.