Statistical Parsing (and related stuff)

Yannick Versley
CiMeC - Università di Trento

May 21st, 2009

Introduction: What and Why?

Wanted: something that will select the best possible parse

- by assigning a score to it
- without us putting disproportionate effort into it

This is a good idea if you:

- believe in graded grammaticality[1] *or*
- want to approximate semantic/pragmatic preferences *or*
- have a grammar that seriously overgenerates

[1] (Sorace and Keller, 2005; Featherston, 2005; Bresnan et al., 2007)

Overview

- Machine Learning in a (tiny) Nutshell
- Parse selection with a ranking function
- Decision-based incremental parsing
- Dynamic Programming models
- The Role of Treebanks
- Summary

Machine Learning

What we want to use:

- have a simple model for the decisions (decision function)
- have a (mathematical) function that, based on
  - our training data
  - some weights

  tells us how well we are doing (*loss function*)
- use numerical techniques to find good weights

$$\operatorname*{argmin}_{w} L(w, \mathrm{data})$$

Decision function (1)

Given multiple alternatives y_i for a datum x, extract a vector of features $\Phi(x, y_i)$ and compute $\langle \Phi(x, y_i), w \rangle$.
Example: guess the category of $x =$*[il] giocatore* from $y_1 = \mathsf{N}, y_2 = \mathsf{V}$

$$\Phi(\textit{[il] giocatore}, \mathsf{N}) = \begin{pmatrix} \text{N:-ore} & \mapsto 1 \\ \text{N:gio-} & \mapsto 1 \\ \text{N:prev=il} & \mapsto 1 \end{pmatrix}$$

Decision function (1)

Given multiple alternatives y_i for a datum x, extract a vector of features $\Phi(x, y_i)$ and compute $\langle \Phi(x, y_i), w \rangle$.
Example: guess the category of $x =$*[il] giocatore* from $y_1 = \mathsf{N}, y_2 = \mathsf{V}$

$$\Phi(\textit{[il] giocatore}, \mathsf{V}) = \begin{pmatrix} \text{V:-ore} & \mapsto 1 \\ \text{V:gio-} & \mapsto 1 \\ \text{V:prev=il} & \mapsto 1 \end{pmatrix}$$

Decision function (2)

Example: guess the category of $x =$*[il] giocatore* from $y_1 = \mathrm{N}, y_2 = \mathrm{V}$.
With

$$w = \begin{pmatrix} \text{N:-ore} & \mapsto +1 \\ \text{N:prev=il} & \mapsto +1 \\ \text{V:prev=il} & \mapsto -1 \end{pmatrix}$$

we would get $\langle \Phi(\textit{[il] giocatore}, \mathrm{N}), w \rangle = 1 \cdot (+1) + 1 \cdot (+1) = 2$
and $\langle \Phi(\textit{[il] giocatore}, \mathrm{V}), w \rangle = 1 \cdot (-1) = -1$

Loss function (1)

Our training data consists of pairs (x, y) of some datum and its *correct* classification plus (implicitly) some set Y of possible classifications for x.

- *Log Loss*: use $\mu(y) = \exp(\langle\Phi(x, y), w\rangle)$ as weights for a probability distribution

$$p(y|x) = \frac{\mu(y)}{\sum_{y' \in Y} \mu(y')}$$

and minimize

$$L(w, \theta) = \sum_{x,y \in \theta} \log p(y|x)$$

Loss function (1)

Our training data consists of pairs (x, y) of some datum and its *correct* classification plus (implicitly) some set Y of possible classifications for x.

- *Log Loss*: use $\mu(y) = \exp\left(\langle \Phi(x, y), w \rangle\right)$ as weights for a probability distribution

$$p(y|x) = \frac{\mu(y)}{\sum_{y' \in Y} \mu(y')}$$

and minimize

$$L(w, \theta) = \sum_{x,y \in \theta} \log p(y|x) + ||w||^2$$

Loss function (1)

Our training data consists of pairs (x, y) of some datum and its *correct* classification plus (implicitly) some set Y of possible classifications for x.

- *Hinge Loss*: try to have the correct y with a safety distance (*margin*) to the others:

$$L(w, \theta) = \sum_{x,y \in \theta} \max\Big(0, \langle \Phi(x, y), w \rangle - \max_{y' \in Y \setminus \{y\}} \langle \Phi(x, y'), w \rangle - 1\Big)$$

Loss function (1)

Our training data consists of pairs (x, y) of some datum and its *correct* classification plus (implicitly) some set Y of possible classifications for x.

- *Hinge Loss*: try to have the correct y with a safety distance (*margin*) to the others:

$$
\begin{aligned}
L(w,\theta) = \sum_{x,y\in\theta} \max\Big(0,\\
&\langle\Phi(x,y),w\rangle - \max_{y'\in Y\setminus\{y\}} \langle\Phi(x,y'),w\rangle - 1\Big)\\
&+ ||w||_1
\end{aligned}
$$

Why these loss functions?

The old days (Artificial Neuronal Networks):

- everyone comes up with their own loss function
- weird loss function = local minima
- $\Rightarrow$ optimization is sensitive to starting conditions
- $\Rightarrow$ numerically problematic

Why these loss functions?

The old days (Artificial Neuronal Networks):

- everyone comes up with their own loss function
- weird loss function = local minima
- ⇒ optimization is sensitive to starting conditions
- ⇒ numerically problematic

Now (Convex loss functions):

- choose one of a few sensible loss functions
- convex function = one global minimum
- use standard numerical optimization techniques
- ⇒ spend more time on interesting things

Preferences in Parsing

(1) He that fears not the future may enjoy the present.

- sounds weird (normally: *does not fear*)
- but we can understand it and want to parse it

Preferences in Parsing

(1) He that fears not the future may enjoy the present.

- sounds weird (normally: *does not fear*)
- but we can understand it and want to parse it

(2) Octuplets mother fears not getting her infants.

- It's *fears [not getting her infants].*
- Don't want to mis-parse it

⇒ need to *prefer* some parses to others.

Most restrictive context

Favor the structure that places greater constraints on allowable constituents.

(3) John looked for Mary.

Argument (*look for X*) > Adjunct (*do sth for X*)

(4) John wants his driver to go to Los Angeles.

Argument (*want+complement*) > Adjunct (*want+purpose*)

(Hobbs and Bear, 1990)

Attach Low and Parallel

Attach as low as possible, and in parallel with other constituents

(5) John phoned the man in Chicago.

(no obj-PP ⇒ nearest attachment)

(6) oil sample and filter

(prefer symmetrical interpretation *. . . and oil filter*)

(7) a program to promote safety in trucks and minivans

(Hobbs and Bear, 1990)

Statistical parse selection

It's possible to encode preferences by hand

- but only up to a point
- and it's tedious
- and you still need data

⇒ use annotated corpora and machine learning

(Frank et al., 1998; Schröder et al., 2000)

Features for Parse selection

- the rule (e.g., S$\longrightarrow$ NP VP)
- grammatical relation (e.g., SUBJ(Mary,sleep))
- structure parallelity
  (e.g. 2 for *[the bucket of water] and [the glass of wine]*)
- subcategorization frame
  (e.g., *sleep(subj), eat(subj,obj)*)
- number of right children
  (*the man with the hat with the feather*)

(Johnson et al., 1999; Riezler et al., 2002; Forst, 2007)

and now?

The LFG grammar used by Riezler et al.

- took about 9 years to develop
- gives millions of parses for long sentences
- has a full parse for 74.7% of the sentences (91.1% including fragments)

How can we

- make parse selection more efficient by not looking at *every* parse?
  → *incremental parsing, dynamic programming*
- turn treebanks into (large, messy) grammars (without spending 9 years)?
  → *treebank grammars*

Decision-based parsing

One idea how to use ML in parsing:

- do simple bottom-up parsing (mostly)
- if there's multiple possible decisions,
  let the classifier decide
- either fully deterministic (never look back)
  or with beam search (keep the *n* best-looking hypotheses)

Decision-based parsing

Shift-reduce parsing

- a stack of partially analyzed fragments
- shift: put new word on stack
- reduceR: add stack[-2] as dependent of stack[-1]
- linkL: link stack[-1] to stack[-2]
- reduceL: pop stack[-1]

Similar approaches for constituent parsing

(Yamada and Matsumoto, 2003; Nivre, 2003)

(Briscoe and Caroll, 1993; Magerman, 1995; Ratnaparkhi, 1999; Hall and Nivre, 2008)

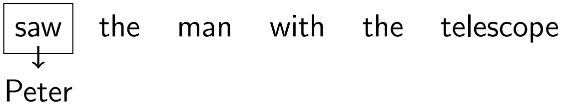
Decision-based parsing

Peter saw the man with the telescope

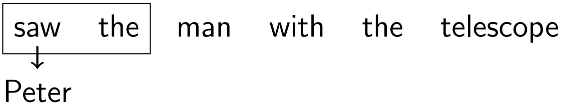
Decision-based parsing

Peter saw the man with the telescope

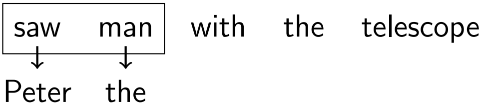
Decision-based parsing



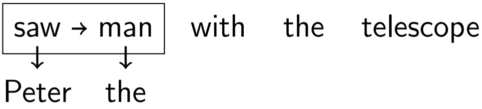
Decision-based parsing



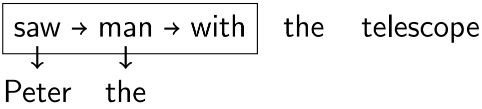
Decision-based parsing



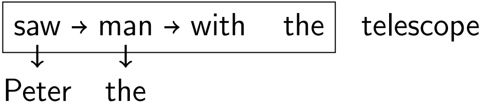
Decision-based parsing



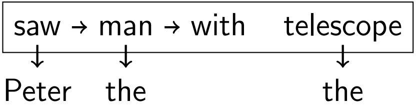
Decision-based parsing



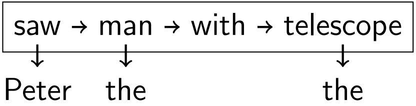
Decision-based parsing



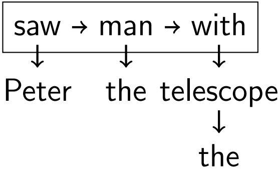
Decision-based parsing



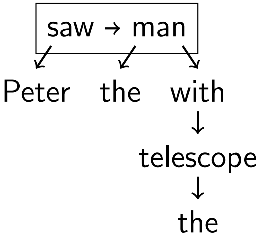
Decision-based parsing



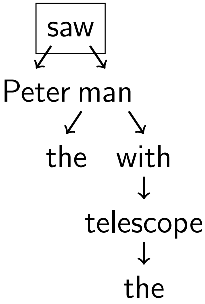
Decision-based parsing



Decision-based parsing



Decision-based parsing



Features for Decision-based parsing

Examples:

- reduceR/LinkL: POS/word of head and dependent
- LinkL: POS/word of grandparent
  (i.e., stack[-3], if it is linked to stack[-1])
- number and kind of dependents

Dynamic Programming Approaches

What to do if there are millions of different parses?

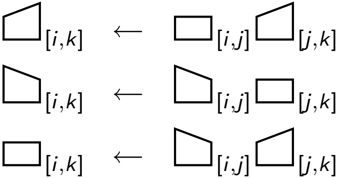
- Chart packing:
  instead of keeping track of full parses,
  form equivalence classes of sub-parses (e.g. $NP_{[0,2]}$)
  and keep track of the best daughter nodes
- Use the scoring function to only return best parse
- Scoring is restricted to equivalence classes
- With 'deep'/unification-based grammar:
  need to choose what to abstract to form equivalence classes

Dynamic Programming approaches

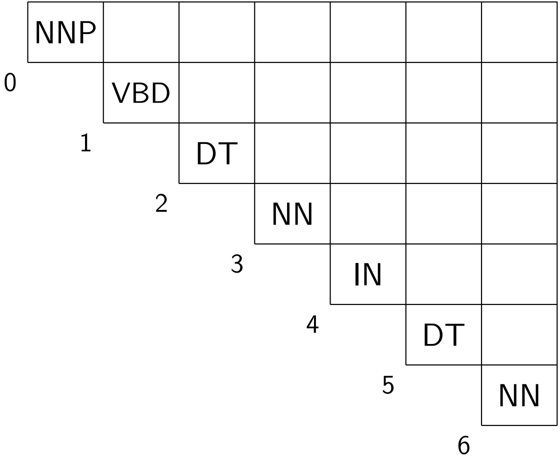
(weighted) deduction on CFG items – Cocke/Kasami/Younger

$$S_{[i,k]} \leftarrow NP_{[i,j]} VP_{[j,k]}$$

lexicalized dependency parsing – Eisner/Satta, McDonald

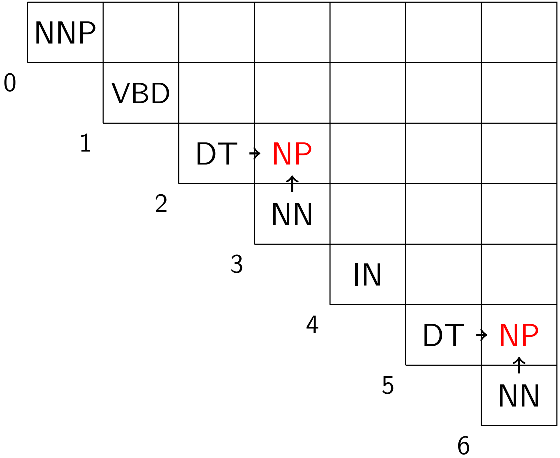


The CKY algorithm



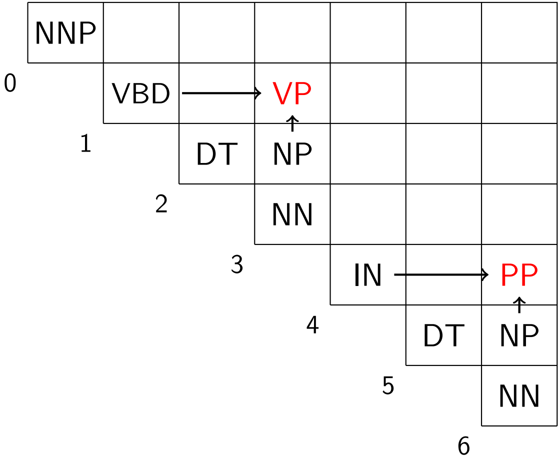
Peter$_{NNP}$ saw$_{VBD}$ the$_{DT}$ man$_{NN}$ with$_{IN}$ the$_{DT}$ telescope$_{NN}$

The CKY algorithm



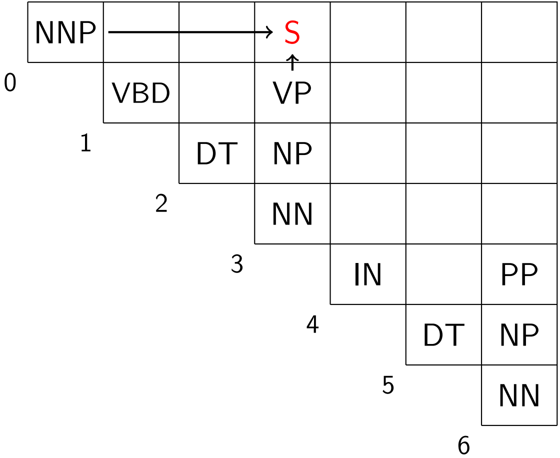
Peter$_{NNP}$ saw$_{VBD}$ the$_{DT}$ man$_{NN}$ with$_{IN}$ the$_{DT}$ telescope$_{NN}$

The CKY algorithm



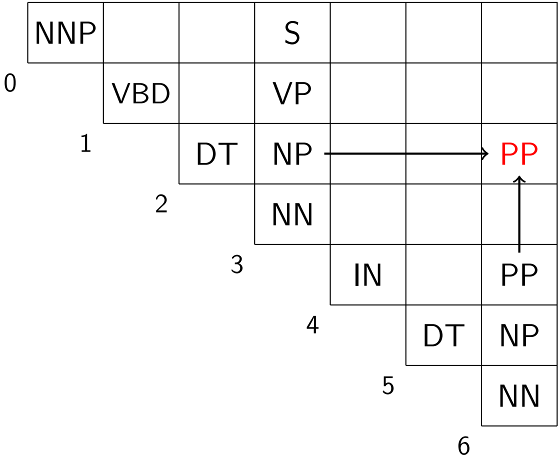
Peter$_{NNP}$ saw$_{VBD}$ the$_{DT}$ man$_{NN}$ with$_{IN}$ the$_{DT}$ telescope$_{NN}$

The CKY algorithm



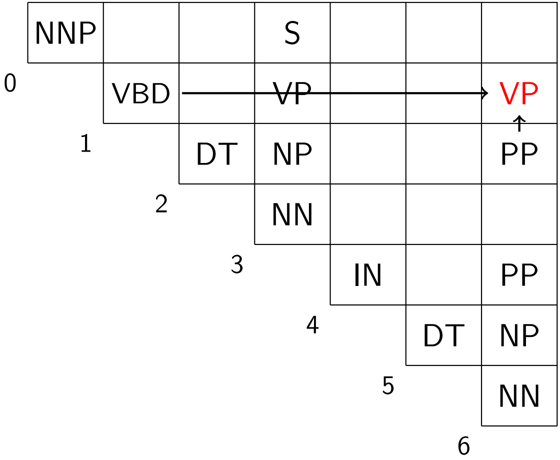
Peter$_{NNP}$ saw$_{VBD}$ the$_{DT}$ man$_{NN}$ with$_{IN}$ the$_{DT}$ telescope$_{NN}$

The CKY algorithm



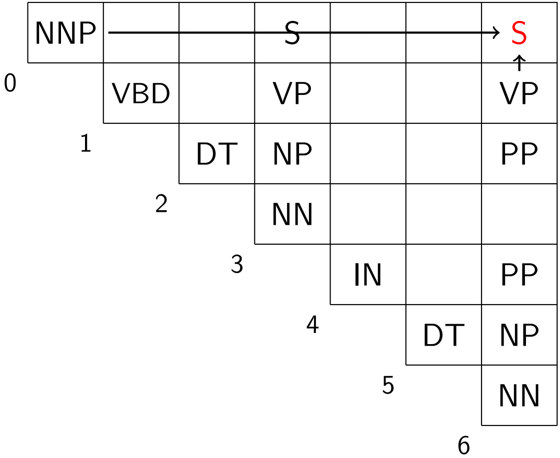
Peter$_{NNP}$ saw$_{VBD}$ the$_{DT}$ man$_{NN}$ with$_{IN}$ the$_{DT}$ telescope$_{NN}$

The CKY algorithm



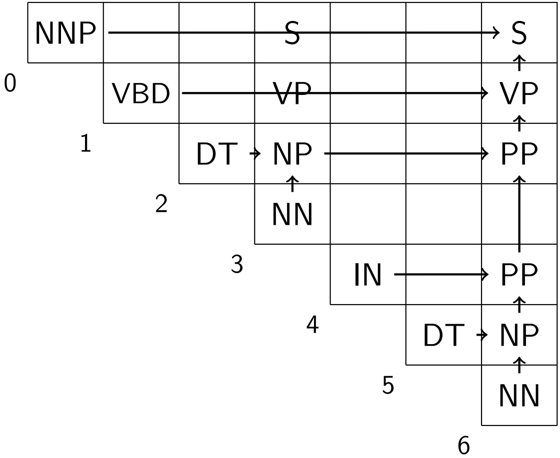
Peter$_{NNP}$ saw$_{VBD}$ the$_{DT}$ man$_{NN}$ with$_{IN}$ the$_{DT}$ telescope$_{NN}$

The CKY algorithm



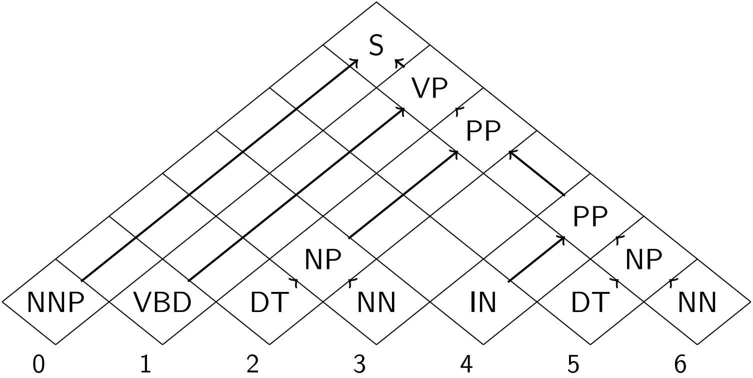
Peter$_{NNP}$ saw$_{VBD}$ the$_{DT}$ man$_{NN}$ with$_{IN}$ the$_{DT}$ telescope$_{NN}$

The CKY algorithm



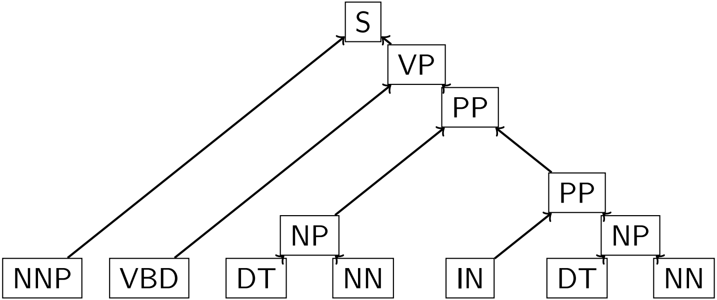
Peter$_{NNP}$ saw$_{VBD}$ the$_{DT}$ man$_{NN}$ with$_{IN}$ the$_{DT}$ telescope$_{NN}$

The CKY algorithm



Peter$_{NNP}$ saw$_{VBD}$ the$_{DT}$ man$_{NN}$ with$_{IN}$ the$_{DT}$ telescope$_{NN}$

The CKY algorithm



Peter$_{NNP}$ saw$_{VBD}$ the$_{DT}$ man$_{NN}$ with$_{IN}$ the$_{DT}$ telescope$_{NN}$

Reranking

What if you want to use non-local features (e.g., parallelism) but keep the efficiency of dynamic programming?
n-best parsing and reranking:

- associate each parse item $Cat_{[i,j]}$ with a list of the n derivations that score highest according to local features (instead of just one)
- get n best parses from root node
- apply model with global features

(Collins, 2000; Charniak and Johnson, 2005; Collins and Koo, 2005)

Improving efficiency in Dynamic Programming

- Supertagging (Bangalore and Joshi, 1999)
  In a lexicalized ("deep") grammar,
  filter out unlikely lexicon entries
- Beam thresholding (Collins, 1996)
  If span $[i, j]$ looks like category X (with probability p),
  ignore anything with a probability of less than $\frac{p}{10\,000}$
- Coarse-to-fine parsing (Goodman, 1997)
  Use a simpler grammar to filter out very unlikely items

Treebanks for parsing

- Treebanks provide us with pairs of sentence and syntactic analysis
- *implicit* grammar (treebank grammar)
- usually need to refine to get a useful grammar:
  - add linguistic information
  - improve sparse data
  - treatment for unknown words

A simple treebank PCFG

- count each rule occurrence (e.g., $S \rightarrow NP\ VP$) in the treebank
- compute rule probabilities

$$p(S \rightarrow NP\ VP|S) = \frac{\text{count}(S \rightarrow NP\ VP)}{\text{count}(S \rightarrow *)}$$

- compute lexical probabilities

$$p(NN \rightarrow \text{dog}|NN) = \frac{\text{count}(NN \rightarrow \text{dog})}{\text{count}(NN \rightarrow *)}$$

(Charniak, 1996)

Unknown words

What is the probability of seeing a sentence with the word "octuplet" in it, according to that grammar?

Unknown words

What is the probability of seeing a sentence with the word "octuplet" in it, according to that grammar?

- exactly zero because no *Cat* $\rightarrow$ octuplet has been observed.

Unknown words

What is the probability of seeing a sentence with the word "octuplet" in it, according to that grammar?

- exactly zero because no *Cat* → octuplet has been observed.

most straightforward way:
create lexicon entries for rare/unknown words

- rare uppercase/rare lowercase
- by word shape (START-II → AAAAA-AA)
- using suffixes (octuplet → UNK-let)
- using a POS tagger (octuplet → UNK-NN)

Adding subcategorization

- Treebank analyses don't contain all the information that is needed – and also should not: Treebanking is tedious enough and we can add the information automatically
- Examples:
  - distinguish auxiliaries (be,have) from main verb *(protect/VB → protect/VV, has/VBZ → has/VHZ)*
  - subcategorize VP, S *(finite, infinitive, to-, gerund, past part., passive)*
  - mark verb argument sequence (e.g., *eat NP* → VV/n)

(Klein and Manning, 2003; Schmid, 2006)

Breaking up rules: Markovization

Many rules just occur once in the corpus

- 17% of test sentences contain a rule that's not in the training data
- Examples:
  VP $\rightarrow$ V NP PP PP PP ...
  [NP [NP John], [NP Peter], [NP Bill], ..., and [NP the others]]

(Collins, 1999; Klein and Manning, 2003)

Breaking up rules: Markovization

Many rules just occur once in the corpus

- 17% of test sentences contain a rule that's not in the training data
- Examples:
  VP → V NP PP PP PP . . .
  [NP [NP John], [NP Peter], [NP Bill], . . . , and [NP the others]]

Solution: break up rules into multiple parts

- VP → VP[V]<PP
- VP[V]<PP → VP[V]<PP PP
- VP[V]<PP → VP[V]<NP PP
- VP[V]<NP → V NP

(Collins, 1999; Klein and Manning, 2003)

Transforming a treebank

. . . to something suitable for your favorite formalism

- role of each constituent in the dominated expansion:
  [VP [ADVP:a just] [VBZ:h opened]
  [NP:c its doors] [PP:a in July]]

(Xia et al., 2000; Hockenmaier and Steedman, 2002)
(Miyao et al., 2004; Hockenmaier, 2006)

Transforming a treebank

...to something suitable for your favorite formalism

- role of each constituent in the dominated expansion:
  [VP [ADVP:a just] [VBZ:h opened]
  [NP:c its doors] [PP:a in July]]
- binarize (i.e., break up rules)
  [VP [ADVP:a just] [VP:h opened its doors in July]]

(Xia et al., 2000; Hockenmaier and Steedman, 2002)
(Miyao et al., 2004; Hockenmaier, 2006)

Transforming a treebank

...to something suitable for your favorite formalism

- role of each constituent in the dominated expansion:
  [VP [ADVP:a just] [VBZ:h opened]
  [NP:c its doors] [PP:a in July]]
- binarize (i.e., break up rules)
  [VP [ADVP:a just] [VP:h opened its doors in July]]
- assign categories (e.g., CCG)
  [$S[dcl]\backslash NP$
  [$(S\backslash NP)/(S\backslash NP)$ *just*]
  [$S[dcl]\backslash NP$ *opened its doors in July*]]

(Xia et al., 2000; Hockenmaier and Steedman, 2002)
(Miyao et al., 2004; Hockenmaier, 2006)

Parsers for treebanking

- Treebanking is expensive (and tedious)
- Use automatic tools to help treebanking
- The simple way: use a POS tagger
  (and maybe a chunker) to provide basic structure
- Can we do better?

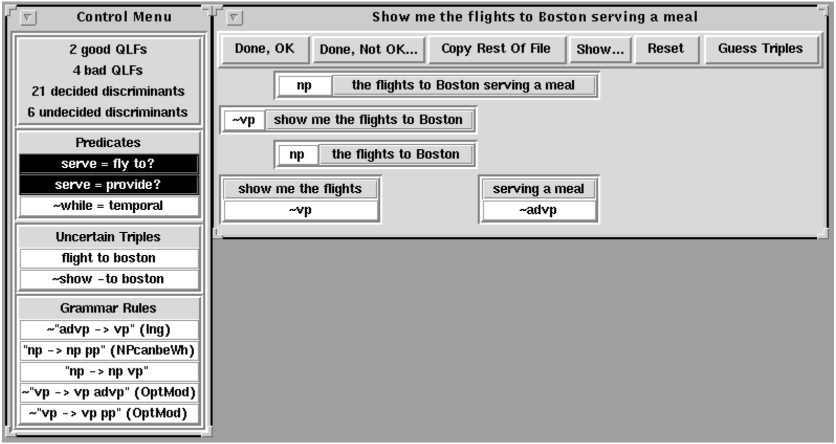
Discriminant-based parse selection

- Let user choose among possible parses with simple yes/no questions
  - word categories *can/NN vs. can/MD*
  - argument tuples *eat(fish) vs. eat(day)*

(Carter, 1997; Rosén et al., 2009)

Discriminant-based parse selection

Carter's TreeBanker



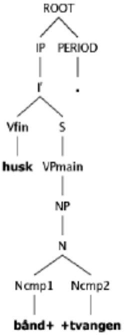
Discriminant-based parse selection

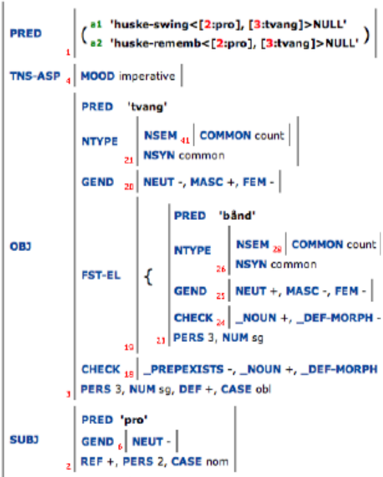
Rosén *et al.*'s ParseBanker

elected solutions: 2 of 2 | ☐ intended

-structure discriminants

'OP 'huske-swing<[],[]>NULL'	1	compl
'OP 'huske-rememb<[],[]>NULL'	1	compl
ıske-swing<[],[]>NULL' VTYPE main	1	compl
ıske-swing<[],[]>NULL' VFORM fin	1	compl
ıske-swing<[],[]>NULL' TNS-ASP MOOD imperative	1	compl
ıske-swing<[],[]>NULL' SUBJ 'pro'	1	compl
ıske-swing<[],[]>NULL' STMT-TYPE imp	1	compl
ıske-swing<[],[]>NULL' OBJ 'tvang'	1	compl
ıske-swing<[],[]>NULL' MAIN-CL +	1	compl
ıske-rememb<[],[]>NULL' VTYPE main	1	compl
ıske-rememb<[],[]>NULL' VFORM fin	1	compl
ıske-rememb<[],[]>NULL' TNS-ASP MOOD imperative	1	compl
ıske-rememb<[],[]>NULL' SUBJ 'pro'	1	compl
ıske-rememb<[],[]>NULL' STMT-TYPE imp	1	compl
ıske-rememb<[],[]>NULL' OBJ 'tvang'	1	compl
ıske-rememb<[],[]>NULL' MAIN-CL +	1	compl





Grammar-based do-what-I-mean: Xcdg

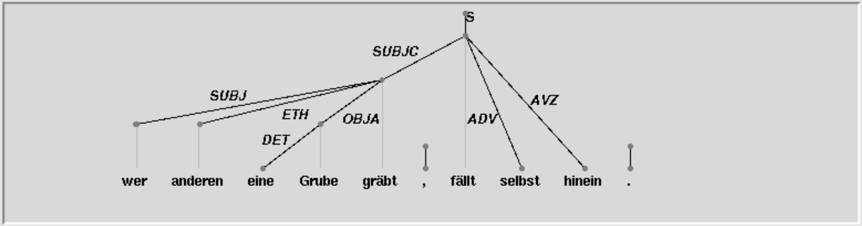
- WCDG formalism: weighted constraints give a score for a tree
- any tree is *feasible*
  (i.e., no need to tweak the grammar to continue with annotation)
- use grammar constraints to automatically choose
  - a headword
  - an edge label
  - a lexical entry
- colorized view to find constraint-violating edges

(Foth et al., 2004)

netsearch:wordgraph0

File Edit View Settings

Parse: parse18 Score: 0/10/7.177e-01



SYN REF

ethischer Dativ	8.000e-01	SYN:anderen-gräbt
Subjektsatz	9.600e-01	SYN:gräbt-fällt
mod-Distanz	9.709e-01	SYN:anderen-gräbt
subcat	9.900e-01	SYN:selbst-fällt
mod	9.900e-01	SYN:anderen-gräbt
mod	9.900e-01	SYN:selbst-fällt
Komplementdistanz	9.960e-01	SYN:wer-gräbt
Komplementdistanz	9.980e-01	SYN:gräbt-fällt
Komplementdistanz	9.980e-01	SYN:hinein-fällt
direction	9.999e-01	SYN:anderen-gräbt

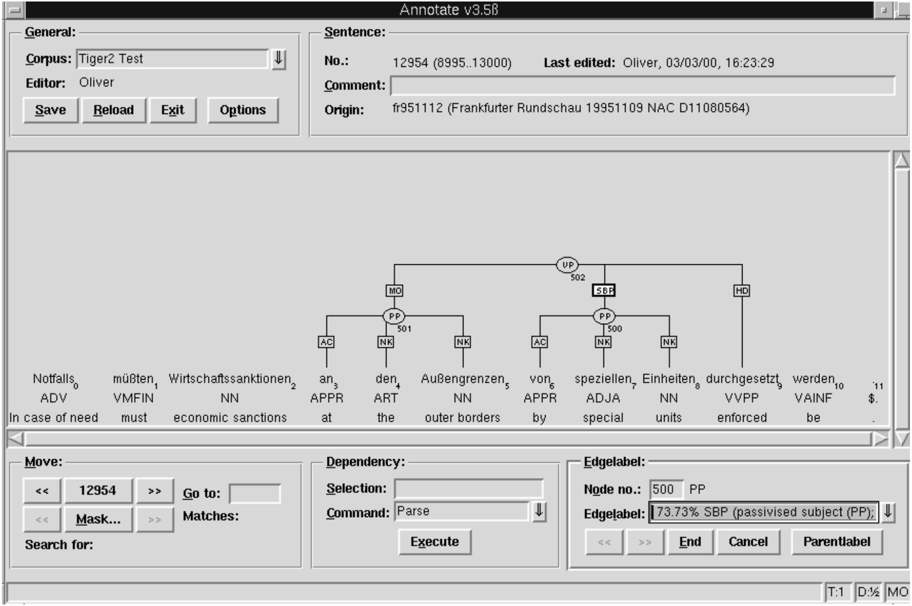
Constituent annotation with automated guesses

The *annotate* tool

- assign node label to selected group of nodes
- assign grammatical functions (with confidence display)
- suggest parent phrases (span + node label)

(Skut et al., 1997; Brants and Plaehn, 2000)

http://www.coli.uni-saarland.de/projects/sfb378/negra-corpus/annotate.html


Annotate v3.5ß
General:
Corpus: Tiger2 Test
Editor: Oliver
Save
Reload
Exit
Options
Sentence:
No.: 12954 (8995..13000)
Last edited: Oliver, 03/03/00, 16:23:29
Comment:
Origin: fr951112 (Frankfurter Rundschau 19951109 NAC D11080564)
VP 502
MO
SBP
HD
PP 501
PP 500
AC
NK
NK
AC
NK
NK
Notfalls 0 ADV In case of need
müßten 1 VMFIN must
Wirtschaftssanktionen 2 NN economic sanctions
an 3 APPR at
den 4 ART the
Außengrenzen 5 NN outer borders
von 6 APPR by
speziellen 7 ADJA special
Einheiten 8 NN units
durchgesetzt 9 VVPP enforced
werden 10 VAINF be
. 11 $. .
Move:
<<
12954
>>
Go to:
Mask...
Matches:
Search for:
Dependency:
Selection:
Command: Parse
Execute
Edgelabel:
Node no.: 500 PP
Edgelabel: 73.73% SBP (passivised subject (PP);
End
Cancel
Parentlabel
T:1
D:½
MO

Summary

- Graded notions of grammaticality
- Parse selection, Incremental parsing
- Dynamic Programming, Reranking
- Extraction of treebank grammars
- Grammar-aided annotation

Link collection

- AMIS (Maximum Entropy learner)
  http://www-tsujii.is.s.u-tokyo.ac.jp/amis/
- MaltParser (incremental dependency parsing)
  http://w3.msi.vxu.se/~nivre/research/MaltParser.html
- BitPar (PCFG parsing)
  http://www.ims.uni-stuttgart.de/tcl/SOFTWARE/BitPar.html
- WCDG parser + Xcdg annotation tool
  http://nats-www.informatik.uni-hamburg.de/view/CDG/DownloadPage
- List of Treebanks
  http://en.wikipedia.org/wiki/Treebank

Thanks for listening!

The End

Idle fun things

Demos

- Enju (HPSG from treebank)
  http://www-tsujii.is.s.u-tokyo.ac.jp/enju/demo.html
- C&C (CCG from treebank)
  http://svn.ask.it.usyd.edu.au/trac/candc/wiki/Demo
- Charniak parser + LFG postprocessing
  http://lfg-demo.computing.dcu.ie/lfgparser.html

Bangalore, S. and Joshi, A. K. (1999). Supertagging: An approach to almost parsing. *Computational Linguistics*, 25:237–265.

Brants, T. and Plaehn, O. (2000). Interactive corpus annotation. In *LREC 2000*.

Bresnan, J., Cueni, A., Nikitina, T., and Baayen, H. (2007). Predicting the dative alternation. In Bouma, G., Kraemer, I., and Swarts, J., editors, *Cognitive Foundations of Interpretation*, pages 69–94. Royal Netherlands Academy of Science, Amsterdam.

Briscoe, T. and Caroll, J. (1993). Generalized probabilistic LR parsing of natural language (corpora) with unification-based grammars. *Computational Linguistics*, 19:25–55.

Carter, D. (1997). The TreeBanker: a tool for supervised training of parsed corpora. In *ACL'97 workshop on*

Computational Environments for Grammar Development and Linguistic Engineering.

Charniak, E. (1996). Tree-bank grammars. Technical Report CS-96-02, Department of Computer Science, Brown University.

Charniak, E. and Johnson, M. (2005). Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proc. ACL 2005*.

Collins, M. (1996). A new statistical parser based on bigram lexical dependencies. In *ACL 1996*.

Collins, M. (1999). *Head-driven statistical models for natural-language parsing*. PhD thesis, Univ. of Pennsylvania.

Collins, M. (2000). Discriminative reranking for natural language parsing. In *ICML 2000*.

Collins, M. and Koo, T. (2005). Discriminative reranking for natural language parsing. *Computational Linguistics*, 31(1):25–69.

Featherston, S. (2005). The decathlon model of empirical syntax. In Reis, M. and Kepser, S., editors, *Linguistic Evidence: Empirical, Theoretical and Computational Perspectives*, pages 187–208. Mouton de Gruyter.

Forst, M. (2007). Filling statistics with linguistics - property design for the disambiguation of German LFG parses. In *ACL 2007 workshop on deep linguistic processing*.

Foth, K., Daum, M., and Menzel, W. (2004). Interactive grammar development with WCDG. In *ACL 2004 Poster session*.

Frank, A., King, T. H., Kuhn, J., and Maxwell, J. (1998).

Optimality Theory style constraint ranking in large-scale LFG grammars. In *Proc. LFG98 Conference*.

Goodman, J. (1997). Global thresholding and multi-pass parsing. In *Second Conference on Empirical Methods in Natural Language Processing (EMNLP 1997)*.

Hall, J. and Nivre, J. (2008). Parsing discontinuous phrase structure with grammatical functions. In *Proceedings of the 6th International Conference on Natural Language Processing (GoTAL 2008)*.

Hobbs, J. R. and Bear, J. (1990). Two principles of parse preference. In *Coling 1990*.

Hockenmaier, J. (2006). Creating a CCGbank and a wide-coverage CCG lexicon for German. In *Coling/ACL 2006*.

Hockenmaier, J. and Steedman, M. (2002). Acquiring compact lexicalized grammars from a cleaner treebank. In *LREC 2002*.

Johnson, M., Geman, S., Canon, S., Chi, Z., and Riezler, S. (1999). Estimators for stochastic "unification-based" grammars. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL-1999)*.

Klein, D. and Manning, C. D. (2003). Accurate unlexicalized parsing. In *ACL 2003*.

Magerman, D. M. (1995). Statistical decision-tree models for parsing. In *ACL'1995*.

Miyao, Y., Ninomiya, T., and Tsujii, J. (2004). Corpus-oriented grammar development for acquiring a head-driven phrase structure grammar from the penn treebank. In *IJCNLP 2004*.

Nivre, J. (2003). An efficient algorithm for projective dependency parsing. In *8th International Workshop on Parsing Technologies*.

Ratnaparkhi, A. (1999). Learning to parse natural language with maximum entropy models. *Machine Learning*, 34:151–178.

Riezler, S., Kind, T., Kaplan, R. M., Crouch, R., Maxwell III, J. T., and Johnson, M. (2002). Parsing the wall street journal using a lexical-functional grammar and discriminative estimation techniques. In *ACL 2002*.

Rosén, V., Meurer, P., and de Smedt, K. (2009). Lfg parsebanker: A toolkit for building and searching a treebank as a parsed corpus. In *TLT 2009*.

Schmid, H. (2006). Trace prediction and recovery with unlexicalized PCFGs and slash features. In *Proceedings of COLING-ACL 2006*.

Schröder, I., Menzel, W., Foth, K., and Schulz, M. (2000). Modeling dependency grammar with restricted constraints. *Traitement Automatique des Langues (TAL)*, 41(1):113–144.

Skut, W., Krenn, B., Brants, T., and Uszkoreit, H. (1997). Software for annotating argument structure. In *ANLP 1997*.

Sorace, A. and Keller, F. (2005). Gradience in linguistic data. *Lingua*, 115:1497–1524.

Xia, F., Palmer, M., and Aravind Joshi, A. (2000). A uniform method of grammar extraction and its applications. In *EMNLP/WVLC 2000*.

Yamada, H. and Matsumoto, Y. (2003). Statistical dependency analysis with support vector machines. In *IWPT 2003*.