Learning a Highly Lexicalized Grammar

Çağrı Çöltekin

Center for Language and Cognition
University of Groningen
c.coltekin@rug.nl

March 4, 2009

Overview

Why lexicalized grammars?

- Most of the current theories of grammar are lexicalized:
  - HPSG
  - LFG
  - CCG
  - Recent Minimalist Theory approaches (where only operation is 'merge')
- Psycholinguistics: The grammar and the lexicon is not as separable as it is traditionally assumed (Bates and Goodman, 1997).
- More suitable for computational simulation: No need to learn a separate rule system.

Categorial Grammars

- Example CG lexical entries:

  Peter := NP : *Peter*

  Mary := NP : *Mary*

  likes := $(S\backslash NP)/NP$: $\lambda x \lambda y . like' \, x \, y$

- Forward Application:

  X/Y: f Y: a $\Rightarrow$ X: fa $(>)$

  Backward Application:

  Y: a $X\backslash Y$: f $\Rightarrow$ X: fa $(<)$

- An example derivation:

$$\dfrac{\dfrac{\dfrac{\textit{Mary}}{\text{NP: } \textit{Mary}'} \quad \dfrac{\textit{likes}}{(S\backslash NP)/NP\text{: } \lambda x \lambda y . like' \, x \, y} \quad \dfrac{\textit{Peter}}{\text{NP: } \textit{Peter}'}}{S\backslash NP\text{: } \lambda y . like' \, Peter' \, y}\,{>}}{S\text{:} like' \, Peter' \, Mary'}\,{<}$$

CFG vs. CG: the lexicon and the rules

CFG:	**CG:**
$NP \rightarrow she$ $V \rightarrow read$ $DET \rightarrow a$ $ADJ \rightarrow nice$ $N \rightarrow book$	$\text{she} := \text{NP}$ $\text{read} := (\text{S}\backslash\text{NP})/\text{NP}$ $\text{a} := \text{NP}/\text{N}$ $\text{nice} := \text{N}/\text{N}$ $\text{book} := \text{N}$
$S \rightarrow NP\ VP$ $VP \rightarrow V\ NP$ $NP \rightarrow DET\ N$ $N \rightarrow ADJ\ N$	$\text{X}/\text{Y}\quad \text{Y} \Rightarrow \text{X} \quad (>)$ $\text{Y}\quad \text{X}\backslash\text{Y} \Rightarrow \text{X} \quad (<)$

Why learning with categorial grammars?

- Highly lexicalized
- Based on sound mathematical formalisms
- Transparency between syntax and semantics
- Encouraging formal results from learning theory
- Extensions (e.g. CCG) are possible for wider coverage of natural languages

What/How to learn?

- All we need to do is assign categories to lexical units.
- Knowledge of semantics helps learning syntactic category.
- This turns the problem into a (semi)supervised classification task.
- A large repository of learning techniques from machine learning is applicable.
- How about learning rules?

A simple/generic algorithm

Input: input sentence with a (possibly noisy) representation of the meaning of the input utterance.
Output: a lexicon containing lexical items of the form $\langle \phi, \sigma, \mu \rangle$.

A simple/generic algorithm

Input: input sentence with a (possibly noisy) representation of the meaning of the input utterance.
Output: a lexicon containing lexical items of the form $\langle \phi, \sigma, \mu \rangle$.

1. For each input, create form a number of segments of the phonological form $\phi = \phi_0 \ldots \phi_N$, and logical form $\mu = \mu_0 \ldots \mu_M$.

A simple/generic algorithm

Input: input sentence with a (possibly noisy) representation of the meaning of the input utterance.
Output: a lexicon containing lexical items of the form $\langle \phi, \sigma, \mu \rangle$.

1. For each input, create form a number of segments of the phonological form $\phi = \phi_0 \ldots \phi_N$, and logical form $\mu = \mu_0 \ldots \mu_M$.
2. For each $\langle \phi_i, \mu_j \rangle$ pair, create a set of lexical hypotheses of the form $\langle \phi_i, \sigma_k, \mu_j \rangle$, where σ_k is possible syntactic categories given ϕ_i and μ_j.

A simple/generic algorithm

Input: input sentence with a (possibly noisy) representation of the meaning of the input utterance.
Output: a lexicon containing lexical items of the form $\langle \phi, \sigma, \mu \rangle$.

1. For each input, create form a number of segments of the phonological form $\phi = \phi_0 \dots \phi_N$, and logical form $\mu = \mu_0 \dots \mu_M$.
2. For each $\langle \phi_i, \mu_j \rangle$ pair, create a set of lexical hypotheses of the form $\langle \phi_i, \sigma_k, \mu_j \rangle$, where σ_k is possible syntactic categories given ϕ_i and μ_j.
3. Parse the input using the lexical hypotheses.

A simple/generic algorithm

Input: input sentence with a (possibly noisy) representation of the meaning of the input utterance.
Output: a lexicon containing lexical items of the form $\langle \phi, \sigma, \mu \rangle$.

1. For each input, create form a number of segments of the phonological form $\phi = \phi_0 \dots \phi_N$, and logical form $\mu = \mu_0 \dots \mu_M$.
2. For each $\langle \phi_i, \mu_j \rangle$ pair, create a set of lexical hypotheses of the form $\langle \phi_i, \sigma_k, \mu_j \rangle$, where σ_k is possible syntactic categories given ϕ_i and μ_j.
3. Parse the input using the lexical hypotheses.
4. Pick the highest probability parse that satisfies the meaning in the input, i.e. 'makes sense'.

A simple/generic algorithm

Input: input sentence with a (possibly noisy) representation of the meaning of the input utterance.
Output: a lexicon containing lexical items of the form $\langle \phi, \sigma, \mu \rangle$.

1. For each input, create form a number of segments of the phonological form $\phi = \phi_0 \dots \phi_N$, and logical form $\mu = \mu_0 \dots \mu_M$.
2. For each $\langle \phi_i, \mu_j \rangle$ pair, create a set of lexical hypotheses of the form $\langle \phi_i, \sigma_k, \mu_j \rangle$, where σ_k is possible syntactic categories given ϕ_i and μ_j.
3. Parse the input using the lexical hypotheses.
4. Pick the highest probability parse that satisfies the meaning in the input, i.e. 'makes sense'.
5. Update the lexicalized grammar, increasing the weights/probabilities of items used in the selected interpretation.

An example from learning Turkish morphology

Input: `a-dam-lar` : *plural*(*man*)	Lexicon contains: `adam` := *N* : *man*

An example from learning Turkish morphology

Input: a-dam-lar : $plural(man)$ Lexicon contains: adam := N : man

1. Generate all lexical hypotheses:

a := N : man
a := N_{plu}/N : $\lambda x.plural(x)$
a.dam := N : man
a.dam := N_{plu}/N : $\lambda x.plural(x)$
dam.lar := N : man
dam.lar := $N_{plu}\backslash N_{dat}$: $\lambda x.plural(x)$
lar := N : man
lar := $N_{plu}\backslash N_{dat}$: $\lambda x.plural(x)$

An example from learning Turkish morphology

Input: a-dam-lar : $plural(man)$ Lexicon contains: adam := $N : man$

1. Generate all lexical hypotheses:

a := $N : man$
a := $N_{plu}/N : \lambda x.plural(x)$
a.dam := $N : man$
a.dam := $N_{plu}/N : \lambda x.plural(x)$
dam.lar := $N : man$
dam.lar := $N_{plu}\backslash N_{dat} : \lambda x.plural(x)$
lar := $N : man$
lar := $N_{plu}\backslash N_{dat} : \lambda x.plural(x)$

2. Parse the input:

(1) $$\dfrac{\dfrac{adam : man}{N : man} \quad \dfrac{lar : \lambda x.plural(x)}{N_{plu}\backslash N : \lambda x.plural(x)}<}{N_{plu} : plural(man)}$$

(2) $$\dfrac{\dfrac{adam : \lambda x.plural(x)}{N_{plu}/N : \lambda x.plural(x)} \quad \dfrac{lar : man}{N : man}<}{N_{plu} : plural(man)}$$

(3) $$\dfrac{\dfrac{a : man}{N : man} \quad \dfrac{damlar : \lambda x.plural(x)}{N_{plu}\backslash N : \lambda x.plural(x)}<}{N_{plu} : plural(man)}$$

(4) $$\dfrac{\dfrac{a : \lambda x.plural(x)}{N_{plu}/N : \lambda x.plural(x)} \quad \dfrac{damlar : man}{N : man}<}{N_{plu} : plural(man)}$$

An example from learning Turkish morphology

Input: a-dam-lar : $plural(man)$ Lexicon contains: adam := N : man

1. Generate all lexical hypotheses:

a := N : man
a := N_{plu}/N : $\lambda x.plural(x)$
a.dam := N : man
a.dam := N_{plu}/N : $\lambda x.plural(x)$
dam.lar := N : man
dam.lar := $N_{plu}\backslash N_{dat}$: $\lambda x.plural(x)$
lar := N : man
lar := $N_{plu}\backslash N_{dat}$: $\lambda x.plural(x)$

2. Parse the input:

(1) $$\dfrac{\dfrac{adam : man}{N : man} \quad \dfrac{lar : \lambda x.plural(x)}{N_{plu}\backslash N : \lambda x.plural(x)}}{N_{plu} : plural(man)}<$$

(2) $$\dfrac{\dfrac{adam : \lambda x.plural(x)}{N_{plu}/N : \lambda x.plural(x)} \quad \dfrac{lar : man}{N : man}}{N_{plu} : plural(man)}<$$

(3) $$\dfrac{\dfrac{a : man}{N : man} \quad \dfrac{damlar : \lambda x.plural(x)}{N_{plu}\backslash N : \lambda x.plural(x)}}{N_{plu} : plural(man)}<$$

(4) $$\dfrac{\dfrac{a : \lambda x.plural(x)}{N_{plu}/N : \lambda x.plural(x)} \quad \dfrac{damlar : man}{N : man}}{N_{plu} : plural(man)}<$$

3. Parse (1) scores highest, since it is supported by the lexicon.
   - 3.1 Item '`lar` := $N_{plu}\backslash N_{dat}$: $\lambda x.plural(x)$' inserted into lexicon
   - 3.2 Weight of item '`adam` := N : man' is increased.

Unsupervised Learning

- The same learning framework works for raw-text input too.
- More difficult:
  - It is difficult to decide which interpretation is the best.
  - More hypotheses are plausible, so increased computational complexity.
- However,
  - Reasonable success indicates a lower bound on what can be learned with more data.
  - Finding semantically annotated corpora is difficult.
  - Unsupervised systems can be (more) useful in practical NLP applications.

Some results: Unsupervised morphology learning

- Child Directed Speech from Turkish CHILDES data
- A completely unsupervised morphology learner
- Using MAP estimate as learning method
- Compared to a state-of-the-art unsupervised morphology learning system.

	Precision	Recall	F-Score
CG-Learner	53.72%	49.50%	51.52%
Creutz and Lagus (2007)	51.84%	50.86%	51.34%

Some results: Learning an Artificial Grammar

A rule-based CG learner trained on an artificially generated grammar was able to learn a wide range of linguistic phenomena (Yao et.al. 2009). Including two popular examples:

- English interrogatives: learning to generate and recognize yes/no questions with subject relatives from data lacking this type of questions.
  Is the man who is hungry eating?
- English auxiliary order: learning to generate and recognize three-auxiliary sequences by only being exposed to sentences with only one and two auxiliaries.
  I should have been going?

Summary and Conclusions

- Lexicalized grammars are suitable for a *cognitively plausible*, *theoretically sound* and *computationally attractive* grammar learning.
- Based on their interaction with the environment, children's learning is an example of semi-supervised learning.
- Unsupervised learning methods may provide a lower bound for success of learning systems.
- Lexicalized grammar learners, CG learners in particular, show promising results for simulating grammar acquisition.

References

Elizabeth Bates and Judith C. Goodman. *On the inseparability of grammar and the lexicon: Evidence from acquisition, aphasia and real-time processing. Language and Cognitive Processes*, 15(5/6): 507–584, 1997.

Mathias Creutz and Krista Lagus. *Unsupervised models for morpheme segmentation and morphology learning. ACM Trans. Speech Lang. Process.*, 4(1):3, 2007.

Xuchen Yao, Jianqiang Ma, Sergio Duarte, and Çağrı Çöltekin. *Unsupervised syntax learning with categorial grammars using inference rules.* 2009. in submission.

English auxiliary inversion: the problem

(1) a. *Peter is awake.*

b. *Is Peter awake?*

c. *Peter who is sleepy is awake.*

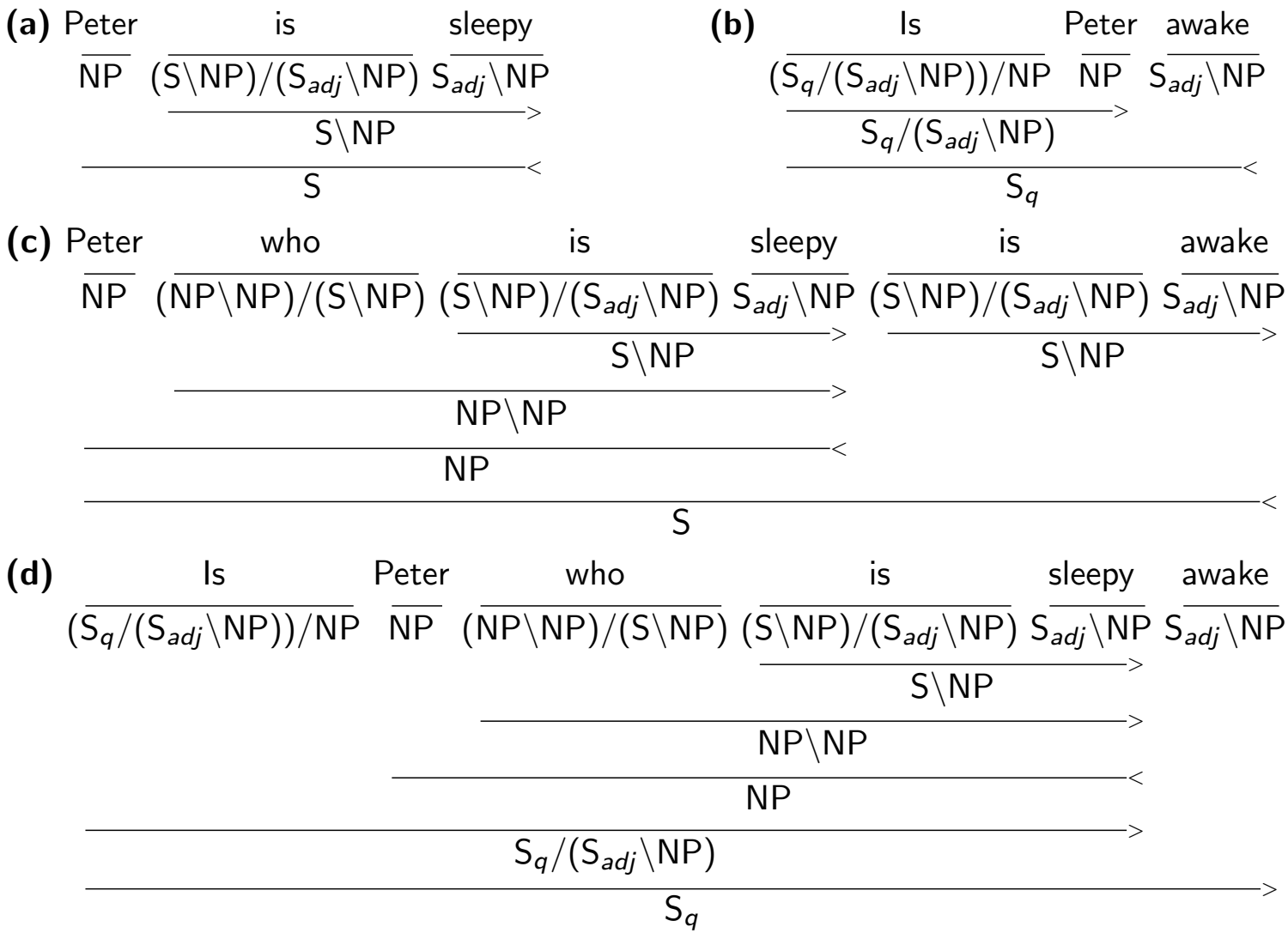
d. *Is Peter who is sleepy awake?*

e. **Is Peter who sleepy is awake?*

The claim:

- The sentences of the form (1a–1c) is common in child directed speech (CDS), but not (1d).
- Provided that, the most plausible way to form questions is arguably moving the first auxiliary to the front. This would result in questions like (1e), but children correctly learn to form questions like (1d).

English auxiliary inversion: the solution



English auxiliary order: the problem

(2) a. *I should go.*
b. *I have gone.*
c. *I am going.*
d. *I have been going.*
e. *I should have gone.*
f. *I should be going.*
g. *I should have been going.*
h. **I have should been going.*

Claim:

- Sentences of the form (2a-2f) are common in CDS, but, (2g) are rare.
- Correct order cannot be learned only from data.

English auxiliary order: the solution

The related fragment of the grammar learned by a simple CG rule learner (Yao et.al. 2009):

$$
\begin{aligned}
\mathit{should} &:= (S_s \backslash NP)/(S \backslash NP) \\
\mathit{should} &:= (S_s \backslash NP)/(S_h \backslash NP) \\
\mathit{should} &:= (S_s \backslash NP)/(S_b \backslash NP) \\
\mathit{have} &:= (S_h \backslash NP)/(S \backslash NP) \\
\mathit{have} &:= (S_h \backslash NP)/(S_b \backslash NP) \\
\mathit{be} &:= (S_b \backslash NP)/(S \backslash NP)
\end{aligned}
$$

What does morphology have in common with syntax?

Some examples from Turkish:

- Long sequences of morphemes in a single word:
  İstanbul-lu-laş-tır-ama-dık-lar-ımız-dan-mı-sınız?
  'Are you one of those we could not convert to an İstanbulite'
- Ambiguous morphemes:

Word:	Analyses:
evleri	$ev\langle N\rangle\langle p3p\rangle$
	$ev\langle N\rangle\langle plu\rangle\langle p1s\rangle$
	$ev\langle N\rangle\langle plu\rangle\langle dat\rangle$

- Recursive structure:

ev	-de	-**ki**	-nin	-**ki**	-ler	-de	-**ki**
house	-LOC	-REL	-POS3s	-REL	-PLU	-LOC	-REL