Natural Language Syntax and Data-to-Text Generation

Claire Gardent
(Joint work with Laura Perez-Beltrachini and Bikash Gyawali)

CNRS/LORIA, Nancy, France

SLSP 2014
Data-to-Text Generation

Maps data (numerical weather simulation, software specification, logical models, taxonomy, ontology, KB, Linked Data) to text
Maps data (numerical weather simulation, software specification, logical models, taxonomy, ontology, KB, Linked Data) to text

Generating from KB data
Data-to-Text Generation

Maps data (numerical weather simulation, software specification, logical models, taxonomy, ontology, KB, Linked Data) to text

Generating from KB data

- Generating descriptions of KB concepts
- Verbalising user queries (NL Interface to KBs)
The No-NL-Syntax Approach

- Angeli et al (ACL 2010): Sequence of Discriminative Models
- Konstas and Lapata (ACL 2012): Probabilistic CFG
The No-NL-Syntax Approach

- Angeli et al (ACL 2010): Sequence of Discriminative Models
- Konstas and Lapata (ACL 2012): Probabilistic CFG

Use a parallel data/text training corpus (air travel, weather forecast, sportcasting)

Learn a direct mapping between phrases and data
The Grammar-Based Approach

Uses a grammar of NL syntax to mediate between phrases and data
The Grammar-Based Approach

Uses a grammar of NL syntax to mediate between phrases and data

Provides an abstraction level which helps when
The Grammar-Based Approach

Uses a grammar of NL syntax to mediate between phrases and data

Provides an abstraction level which helps when

- little training data is available
  (linguistically guided grammar induction)
The Grammar-Based Approach

Uses a grammar of NL syntax to mediate between phrases and data

Provides an abstraction level which helps when

- little training data is available  
  (linguistically guided grammar induction)
- domain portability is required  
  (small hand written grammar + automatic lexicon acquisition)
The Grammar-Based Approach

Uses a grammar of NL syntax to mediate between phrases and data

Provides an abstraction level which helps when

- little training data is available
  (linguistically guided grammar induction)
- domain portability is required
  (small hand written grammar + automatic lexicon acquisition)
- statistical hypertagging is needed
  (learn a small set of abstract syntactic properties rather than a larger set of hypertags)
Grammar-Based Generation
Grammar-Based Generation

The train \( \text{train}(A) \) departs \( \text{departure}(B,C) \) at 10am \( \text{tenAM}(D) \).

Claire Gardent (CNRS/LORIA, Nancy, France)
Grammar-Based Generation

Input data: \( train(t), \text{departure}(e,t), \text{tenAM}(e) \)
Grammar-Based Generation

Input data: $train(t)$, $departure(e,t)$, $tenAM(e)$
Grammar-Based Generation

S

NP
The train

train(t)

VP

V[agr:3sg]

departs

departure(e,t)

NP

10am
tenAM(e)

P

at

VP

VP*e

PP

P

10am
tenAM(e)
Grammar-Based Generation

S
   NP
   | The train
   | train(t)
   VP
   | V[agr:3sg]
   | departs
   | departure(e,t)
   PP
   | P
   | at
   | 10am
   | tenAM(e)
The train departs at 10am
Since each tree is lexicalised, the resulting grammar can be very large. In practice, we therefore
Separating Grammar from Lexicon

Since each tree is lexicalised, the resulting grammar can be very large. In practice, we therefore

- abstract over lexical items in the grammar
Separating Grammar from Lexicon

Since each tree is lexicalised, the resulting grammar can be very large. In practice, we therefore

- abstract over lexical items in the grammar
- use a lexicon to determine which grammar tree is lexicalised/anchored by which lexical items
Separating Grammar from Lexicon

Syntax: CanonicalSubject
Anchor: departs

Semantics: departure
Tree: nx0V
Syntax: CanonicalSubject
Anchor: arrives

...
1. Using Syntax to learn from little Data

KBGen 2012: an international shared task
1. Using Syntax to learn from little Data

KBGen 2012: an international shared task

Given a set of relations selected from the AURA knowledge base, generate a sentence that is grammatical and fluent in English.
1. Using Syntax to learn from little Data

KBGen 2012: an international shared task

Given a set of relations selected from the AURA knowledge base, generate a sentence that is grammatical and fluent in English.

![Diagram](image)

The rate of absorption of a central vacuole is directly proportional to the size of the vacuole.
The KBGen Shared Task

Small Training Corpus: 207 training instances (data/text pairs)

3 Participants:
The KBGen Shared Task

Small Training Corpus: 207 training instances (data/text pairs)

3 Participants:

UDEL: Symbolic Rule Based System (U. Delaware)
The KBGen Shared Task

Small Training Corpus: 207 training instances (data/text pairs)

3 Participants:

UDEL: Symbolic Rule Based System (U. Delaware)

IMS: Statistical System using a probabilistic grammar induced from the training data
The KBGen Shared Task

Small Training Corpus: 207 training instances (data/text pairs)

3 Participants:

UDEL: Symbolic Rule Based System (U. Delaware)

IMS: Statistical System using a probabilistic grammar induced from the training data

LOR-KBGEN: uses a linguistically principled grammar induced from the training data
The LOR-KBGen Approach

Grammar-Based Generation

The grammar is automatically induced from the training corpus.

Grammar induction is guided by two linguistic principles namely, the *Extended Domain of Locality* and the *Semantic Principle* i.e., grammar trees must:

- capture a semantically coherent unit
- group syntactic functors with their dependents

[0] B. Gyawali and C. Gardent
Surface Realisation from Knowledge-Bases.
ACL 2014. Baltimore, USA.
Grammar Induction

For each (data,sentence) pair in the input:

- Parse sentence
- Align semantic variables with words
- Project variables up the parse tree
- Extract subtrees from the parse tree s.t. each subtree describes a coherent syntactic/semantic unit (Semantic and Extended Domain of Locality Principles)

Generalise grammar by

- “unlexicalising” the extracted tree (lexicon/grammar abstraction)
- guessing missing lexical entries
- splitting larger trees into smaller, more reusable, ones
Example

Data

Release-Of-Calcium (RoC)
Gated-Channel (GC)
Particle-In-Motion (PM)
Endoplasmic-Reticulum (ER)
agent (RoC, GC)
object (RoC, PM)
base (RoC, ER)
has-function (GC, RoC)

Sentence

The function of a gated channel is to release particles from the endoplasmic reticulum.
the fn of a gated channel to release particles from the endoplasmic reticulum

NP

VP

S

VP

S

NP

PP

NP

dt reticulum dt endoplasmic dt the

in from

nns particles

vb release

to

vbz is

np channel

jj gated

da

of

the

fn

of

a

gated

channel

to release

particles from

the endoplasmic reticulum

natural language syntax and data-to-text generation
Parse Tree and Variable Projection

S

NP

NP

VP

PP

VBZ

is

S

NP

NP

PP

DT

NN

IN

NP

d DT

NN

g gated channel

GP

TO

VB

NP

to release

NNS

IN

NP

PP

DT

DT

DT

the endoplasmic reticulum

Gated-Channel(GC)
Release-Of-Calcium(RoC)
Particle-In-Motion(PM)
Endoplasmic-Reticulum(ER)
Variable Alignement and Variable Projection

Gated-Channel(GC)
Release-Of-Calcium(RoC)
Particle-In-Motion(PM)
Endoplasmic-Reticulum(ER)
FB-LTAG Trees Extraction

- First, the subtrees associated with Entities are extracted.
- Second, the subtrees associated with Events are extracted.
First, the subtrees associated with Entities are extracted.
Second, the subtrees associated with Events are extracted.
Putting Syntax and Semantics Together

Gated-Channel\((GC)\)

Gated-Channel\((GC)\)

Endoplasmic-Reticulum\((ER)\)

Particle-In-Motion\((PM)\)

Claire Gardent (CNRS/LORIA, Nancy, France)  
Natural Language Syntax and Data-to-Text Generation

SLSP 2014 16 / 50
Lexicon/Grammar Abstraction

**Tree Schema**

**Lexicon**

<table>
<thead>
<tr>
<th>Semantics</th>
<th>Anchor</th>
<th>CoAnchors</th>
</tr>
</thead>
<tbody>
<tr>
<td>release(E), object(E,A), base(E,B), agent(E,C), has-function(C,E)</td>
<td>release</td>
<td>V1→The, V2→function, V3→of, V4→is, V5→to, V6→from</td>
</tr>
</tbody>
</table>
Lexicon/Grammar Abstraction

Tree Schema

```
NP \rightarrow S_{E_3}
  \rightarrow NP \rightarrow VP_{E_1}
  \rightarrow PP \rightarrow VBZ \rightarrow S_{E_2}
  \rightarrow VP_{E_1} \rightarrow VP_{E_2}
  \rightarrow NP \rightarrow NP \rightarrow PP
  \rightarrow DT \rightarrow NN \rightarrow IN \rightarrow NP \rightarrow C
  \rightarrow \diamond V1 \rightarrow \diamond V2 \rightarrow \diamond V3
  \rightarrow \diamond V4 \rightarrow \diamond V5 \rightarrow \diamond V6
  \rightarrow VBZ \rightarrow \diamond V4
  \rightarrow \diamond V5 \rightarrow \diamond V6
  \rightarrow IN \rightarrow \diamond V6
```

Lexicon

<table>
<thead>
<tr>
<th>Semantics</th>
<th>Anchor</th>
<th>CoAnchors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ex.1 release(E), object(E,A), base(E,B), agent(E,C), has-function(C,E)</td>
<td>release</td>
<td>V1→The, V2→function, V3→of, V4→is, V5→to, V6→from</td>
</tr>
<tr>
<td>Ex.2 carry(E), object(E,A), instrument(E,B), agent(E,C), has-function(C,E)</td>
<td>carry</td>
<td>V1→The, V2→function, V3→of, V4→is, V5→to, V6→using</td>
</tr>
</tbody>
</table>
• Ex.1: The function of $C$ is to release $A$ from $B$.
• Ex.2: The function of $C$ is to carry $A$ using $B$. 

Claire Gardent (CNRS/LORIA, Nancy, France) 
Natural Language Syntax and Data-to-Text Generation 
SLSP 2014 17 / 50
Lexicon Expansion

release(E)
object(E,A), base(E,B)
agent(E,C), has-function(C,E)
Lexicon Expansion

release(E)

object(E, A), base(E, B)

agent(E, C), has-function(C, E)
release(E)

object(E,A), base(E,B)

agent(E,C), has-function(C,E)

The function of C is to release A from B.
The function of $C$ is to \textit{release} $A$ \textit{from} $B$. 

\begin{itemize}
  \item \textit{release}(E)
  \item object(E,A), base(E,B)
  \item agent(E,C), has-function(C,E)
  \item \textit{generate}(E)
  \item object(E,A), base(E,B)
  \item agent(E,C), has-function(C,E)
\end{itemize}
The function of $C$ is to release $A$ from $B$. 
The function of C is to release A from B.

The function of C is to generate A in B.
The function of C is to *release* A from B.

The function of C is to *generate* A in B.
Grammar Expansion

S

E3

NP

VP

E3

E2

NP

PP

VBZ

is

S

E2

E1

E

VBP

E

TO

to

VB

release

NP

IN

NP

fn

of

the

release(E)

object(E,A)

agent(E,C)

has-function(C,E)

base(E,B)

from
Grammar Expansion

We further extract from each Event tree, subtrees corresponding to Subject-Verb-Object structure and optional modifiers.

```
S_{E_3}
  /  \\  
NP  VP_{E_3}^{E_2}
  \  
  /  \\  
NP  PP   VP_{E_1}^{E_1} \\
 |   |   |    
NP  IN  NP_{C}^{C}  |   |    |   |   |    |   |    |
|   |   |   |    |   |   |   |    |   |    |
the fn of is is
```

```
release(E)
object(E,A)
agent(E,C)
has-function(C,E)
```

```
base(E,B)
```
The function of X is to release Y
The function of X is to release Y to Z
Evaluation and Results

- 72 inputs from KBGEN
- Automatic Evaluation: BLEU
- Human-Based Evaluation
  - 12 participants were asked to rate sentences along three dimensions:
    - **fluency**: Is the text easy to read?
    - **grammaticality**: Is the text grammatical?
    - **adequacy**: Does the meaning conveyed by the generated sentence correspond to the meaning conveyed by the reference sentence?
  - Online evaluation (LG-Eval toolkit)
  - Subjects used a sliding scale
  - Latin Square Experimental Design was used to ensure that each evaluator sees the same number of output from each system and for each test set item.
Results

BLEU score

<table>
<thead>
<tr>
<th>System</th>
<th>BLEU Score</th>
<th>Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDEL</td>
<td>0.32</td>
<td>Hand-written Rewrite Rules</td>
</tr>
<tr>
<td>LOR-KBGen</td>
<td>0.29</td>
<td>Automatically Induced Grammar</td>
</tr>
<tr>
<td>IMS</td>
<td>0.12</td>
<td>Automatically Induced Probabilistic Grammar</td>
</tr>
</tbody>
</table>

Human Evaluation

<table>
<thead>
<tr>
<th>System</th>
<th>Fluency</th>
<th>Grammaticality</th>
<th>Meaning Similarity</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDEL</td>
<td>4.36</td>
<td>4.48</td>
<td>3.69</td>
</tr>
<tr>
<td>LOR-KBGen</td>
<td>3.45</td>
<td>3.55</td>
<td>3.65</td>
</tr>
<tr>
<td>IMS</td>
<td>1.91</td>
<td>2.05</td>
<td>1.31</td>
</tr>
</tbody>
</table>
Conclusion

Linguistically guided grammar induction ...
Conclusion

Linguistically guided grammar induction ...

permits a fully automated approach (unlike the UDEL system)
Conclusion

Linguistically guided grammar induction ...

permits a fully automated approach (unlike the UDEL system)

yields output sentences whose quality is competitive with those produced by a hand written system (unlike the IMS system)
2. Using Syntax to support portability

NLG-based Natural language interfaces to KB

The interactive refinement of the user query proceeds as follows:

- Possible (consistent with KB) extensions of the current user query are computed by an automated reasoner $\Rightarrow$ Set of DL formulae ($F$)
- Each formal extension ($f \in F$) is then verbalised using NLG
- N.B. The user may revise (substitute, delete, add) the current query

[0] L. Perez-Beltrachini and C. Gardent
Incremental Query Generation
Interactive Query Formulation

(1) a. $\top$ (initial query)
   I am looking for something

b. $\text{Man}$ (substitute concept)
   I am looking for a man

c. $\text{Man} \sqcap \text{Young}$ (add compatible concept)
   I am looking for a young man

d. $\text{Man} \sqcap \text{Young} \sqcap \exists \text{isMarried} . (\text{Person})$ (add relation)
   I am looking for a young man who is married to a person

e. $\text{MarriedMan} \sqcap \text{Young}$ (substitute selection)
   I am looking for a young married man

f. $\text{MarriedMan}$ (delete concept)
   I am looking for a married man
Constraints on Generation

Input Data: a tree shaped conjunctive formula

Output: NL verbalisation of the input data

Constraints on NLG:
- should avoid recomputing each extension from scratch (tabulation)
- should support incrementality (revisions, deletions and additions)
- should preserve NL linear order
- should be portable to any KB
- no training corpus available
The Quelo Generator

- Small, **domain independent** grammar (59 trees)
- **Automatically extracted lexicon**
- Conditional Random Field hypertagger
  - prunes the initial search space
  - performs data segmentation into sentence size chunks
- Grammar-Based Surface realisation algorithm maps data to text
  - Incremental: allows for revisions, uses tabulation to avoid recomputation
  - Beam search uses linear order preserving heuristics to guide the search
- Referring expression modules handles choice of NP (pronoun, definite or indefinite NP)
- Ranking module chooses best output
Automatic Lexicon Acquisition

Tokenize and tag relations
Automatic Lexicon Acquisition

Tokenize and tag relations

\textsc{equippedWith} \rightarrow \textit{equipped/VBD with/IN}
Automatic Lexicon Acquisition

Tokenize and tag relations

\texttt{EQUIPPED\ WITH} \rightarrow \texttt{equipped/VBD with/IN}

Map resulting sequence to a TAG family (set of grammar units)
Automatic Lexicon Acquisition

Tokenize and tag relations

\[
equippedWith \rightarrow \text{equipped/VBD with/IN}
\]

Map resulting sequence to a TAG family (set of grammar units)

\[
equipped/VBD \ with/IN \rightarrow \text{n}x0\text{V}p\text{n}x1
\]
The nx0Vpnx1 Family

The grammar captures the possible verbalisations of a relation mapped to the nx0Vpnx1 family.

<table>
<thead>
<tr>
<th>Example</th>
<th>Tree Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>hline NP₀ should be equipped with NP₁</td>
<td>nx0VVVpnx1</td>
</tr>
<tr>
<td>It₀ should be equipped with NP₁</td>
<td>PRO0VVVpnx1</td>
</tr>
<tr>
<td>and NP₀ should be equipped with NP₁</td>
<td>sCONJnx0VVVpnx1</td>
</tr>
<tr>
<td>and it₀ should be equipped with NP₁</td>
<td>sCONJPRO0VVVpnx1</td>
</tr>
<tr>
<td>NP₀ which should be equipped with NP₁</td>
<td>W0nx0VVVpnx1</td>
</tr>
<tr>
<td>NP₀ (...) and which should be equipped with NP₁</td>
<td>ANDWHn0VVVpnx1</td>
</tr>
<tr>
<td>NP₀ (...) which should be equipped with NP₁</td>
<td>COMMAWHn0VVVpnx1</td>
</tr>
<tr>
<td>NP₀ equipped with NP₁</td>
<td>betanx0VPpnx1</td>
</tr>
<tr>
<td>NP₀ (...) and equipped with NP₁</td>
<td>betanx0ANDVPpnx1</td>
</tr>
<tr>
<td>NP₀ (...) equipped with NP₁</td>
<td>betanx0COMMAVPpnx1</td>
</tr>
<tr>
<td>NP₁ with which NP₀ should be equipped</td>
<td>W1pnx1nx0VV</td>
</tr>
<tr>
<td>NP₀ (equipped with X) and with NP₁</td>
<td>betavx0ANDVVVpnx1</td>
</tr>
<tr>
<td>NP₀ (equipped with X), with NP₁</td>
<td>betavx0COMMAVVVpnx1</td>
</tr>
</tbody>
</table>
Evaluation

Lexicon

- Lexicon extraction tested on 200 ontologies (M. Trevisan)
- Coverage: 85% of the ontology relations (12000 relns, 13 templates)
Evaluation

Lexicon

- Lexicon extraction tested on 200 ontologies (M. Trevisan)
- Coverage: 85% of the ontology relations (12000 relns, 13 templates)

Grammar

- NLG tested on 5 ontologies (cinema, wines, human abilities, assistive devices, ecommerce), 40 queries.
- Coverage 87%
I am looking for a car. Its make should be a Land Rover. The body style of the car should be an off-road car. The exterior color of the car should be beige.

I am looking for a car whose make is a Land Rover, whose body style is an off-road car and whose exterior color is beige.
Assessing Quelo Template-Based Queries

41 queries capturing different combinations of concepts and relations

8 raters

50% of the queries are rated as disfluent
10% of the queries are rated as unclear

Free Comments: too repetitive, lacks aggregation
Comparing Template-Based and Grammar-Generated Queries

10 raters, 14 query pairs built from two ontologies (cars, universities)

<table>
<thead>
<tr>
<th></th>
<th>Fluency</th>
<th>Clarity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grammar Templates</td>
<td>19.76</td>
<td>6.87</td>
</tr>
<tr>
<td></td>
<td>7.2</td>
<td>8.57</td>
</tr>
</tbody>
</table>
3. Using Syntax to improve Statistical Hypertagging

The Hypertagging Task: Given a sequence of input symbols (data), hypertagging seeks to find the most likely sequence of grammar units (trees).
### The Hypertagging Task

Given a sequence of input symbols (data), hypertagging seeks to find the most likely sequence of grammar units (trees).

<table>
<thead>
<tr>
<th>CarDealer locatedIn City sell Car runOn Diesel</th>
</tr>
</thead>
<tbody>
<tr>
<td>nx betanx0VPpnx1 PRO0VVVpnx1 ANDWHnx0VVVnx1 Car nx0VVVpnx1 nx</td>
</tr>
<tr>
<td>sCONJnx0VVVpnx1 sCONJnx0VVVnx1 ANDWHnx0VVVnx1 ...</td>
</tr>
<tr>
<td>sCONJPRO0VVVpnx1 sCONJPRO0VVVnx1 ANDWHnx0VVVnx1 ...</td>
</tr>
<tr>
<td>W0nx0VVVpnx1 W0nx0VVVnx1 ANDWHnx0VVVnx1 ...</td>
</tr>
<tr>
<td>ANDWHnx0VVVpnx1 ANDWHnx0VVVnx1 ANDWHnx0VVVnx1 ...</td>
</tr>
<tr>
<td>COMMAWHnx0VVVpnx1 COMMAWHnx0VVVnx1 ANDWHnx0VVVnx1 ...</td>
</tr>
<tr>
<td>betanx0VPpnx1 betanx0VPpnx1 betanx0VPpnx1 ...</td>
</tr>
<tr>
<td>betanx0ANDVPpnx1 betanx0ANDVPpnx1 betanx0ANDVPpnx1 ...</td>
</tr>
<tr>
<td>betavx0COMMAAVVpnx1 betavx0COMMAAVVpnx1 betavx0COMMAAVVpnx1 ...</td>
</tr>
</tbody>
</table>
3. Using Syntax to improve Statistical Hypertagging

**The Hypertagging Task:** Given a sequence of input symbols (data), hypertagging seeks to find the most likely sequence of grammar units (trees).

<table>
<thead>
<tr>
<th>CarDealer locatedIn</th>
<th>City sell</th>
<th>Car runOn</th>
<th>Diesel</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>betanx0VPpnx1</strong></td>
<td><strong>ANDWHnx0VVnx1</strong></td>
<td><strong>nx0VVpnx1</strong></td>
<td><strong>nx</strong></td>
</tr>
<tr>
<td>PRO0VVVpnx1</td>
<td>PRO0VVnx1</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>sCONJnx0VVVpnx1</td>
<td>sCONJnx0VVnx1</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>sCONJPRO0VVVpnx1</td>
<td>sCONJPRO0VVnx1</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>W0nx0VVVpnx1</td>
<td>W0nx0VVVnx1</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>ANDWHnx0VVVpnx1</td>
<td>ANDWHnx0VVVnx1</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>COMMAWHnx0VVVpnx1</td>
<td>COMMAWHnx0VVVnx1</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>betanx0VPpnx1</td>
<td>betanx0VPpnx1</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>betanx0ANDVPpnx1</td>
<td>betanx0ANDVPpnx1</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>betavx0COMMAVVVpnx1</td>
<td>betavx0COMMAVVVpnx1</td>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

*I am looking for a car dealer located in a city and who should sell a car. The car should run on diesel.*
Hypertagging

Given a set of candidate hypertags associated with each literal, the hypertagging task consists into finding the optimal hypertag sequence $y^*$ for a given input semantics $x$:

$$y^* = \text{argmax}_y P(y \mid x)$$

Learn the mapping between observed input features and hidden syntactic classes using a Linear-chain Conditional Random Field (CRF) model.
Training corpus: 145 \langle \text{data}, \text{sequence of tree names} \rangle \text{ pairs}

59 tree names
Training Data, Trees and Linguistic Abstractions

Training corpus: 145 \(\text{data,sequence of tree names}\) pairs
59 tree names

E.g.,
CarDealer locatedIn City sell Car runOn Diesel
\(nx\) betan\(x_0\) VPpn\(x_1\) \(nx\) ANDWH\(nx_0\) VV\(nx_1\) \(nx\) \(nx_0\) VVpn\(x_1\) \(nx\)
Training Data, Trees and Linguistic Abstractions

Training corpus: 145 \( \langle \text{data}, \text{sequence of tree names} \rangle \) pairs
59 tree names

E.g.,

\[
\text{CarDealer} \quad \text{locatedIn} \quad \text{City} \quad \text{sell} \quad \text{Car} \quad \text{runOn} \quad \text{Diesel} \\
\text{nx} \quad \text{beta} \quad \text{nx} \quad \text{0VP} \quad \text{p} \quad \text{nx} \quad \text{1} \quad \text{nx} \quad \text{AND} \quad \text{WH} \quad \text{nx} \quad \text{0VV} \quad \text{nx} \quad \text{1} \quad \text{nx} \quad \text{nx} \quad \text{nx} \quad \text{0VV} \quad \text{p} \quad \text{nx} \quad \text{1} \quad \text{nx}
\]

- Tagging accuracy on complete input: 62.02% (on 10 best outputs)
- Often fails to predict a sequence that leads to successful generation
Using Linguistic Abstractions

Learn to detect optimal sequences of (22) **syntactic classes**, not **grammar units (trees)**

**Intuition:** Trees = syntax (grammar) + subcategorisation (lexicon)

Subcategorisation = prepositional object (**X is equipped with Y**)

<table>
<thead>
<tr>
<th>Example</th>
<th>Tree Name</th>
<th>Syntactic Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP₀ should be equipped with NP₁</td>
<td>nx₀VVVpnₓ₁</td>
<td>Canonical</td>
</tr>
<tr>
<td>It₀ should be equipped with NP₁</td>
<td>PRO₀VVVpnₓ₁</td>
<td>SubjPro</td>
</tr>
<tr>
<td>and NP₀ should be equipped with NP₁</td>
<td>sCONJ₀VVVpnₓ₁</td>
<td>Scoord</td>
</tr>
<tr>
<td>and it₀ should be equipped with NP₁</td>
<td>sCONJPRO₀VVVpnₓ₁</td>
<td>ScoordSubjPro</td>
</tr>
<tr>
<td>NP₀ which should be equipped with NP₁</td>
<td>W₀nn₀VVVpnₓ₁</td>
<td>SubjRel</td>
</tr>
<tr>
<td>NP₀ (...) and which should be equipped with NP₁</td>
<td>ANDWH₀n₀VVVpnₓ₁</td>
<td>SubjRelAnd</td>
</tr>
<tr>
<td>NP₀ (...), which should be equipped with NP₁</td>
<td>COMMAWH₀n₀VVVpnₓ₁</td>
<td>SubjRelComma</td>
</tr>
<tr>
<td>NP₀ equipped with NP₁</td>
<td>betan₀XPpnₓ₁</td>
<td>ParticipialOrGerund</td>
</tr>
<tr>
<td>NP₀ (...) and equipped with NP₁</td>
<td>betan₀ANDVPpnₓ₁</td>
<td>ParticipialOrGerundAnd</td>
</tr>
<tr>
<td>NP₀ (...), equipped with NP₁</td>
<td>betan₀COMMAVPpnₓ₁</td>
<td>ParticipialOrGerundComma</td>
</tr>
<tr>
<td>NP₁ with which NP₀ should be equipped</td>
<td>W₁pnₓ₀n₀VV</td>
<td>PObjRel</td>
</tr>
<tr>
<td>NP₀ (equipped with X) and with NP₁</td>
<td>betavₓ₀ANDVVVpnₓ₁</td>
<td>SubjEllipAnd</td>
</tr>
<tr>
<td>NP₀ (equipped with X), with NP₁</td>
<td>betavₓ₀COMMAVVVpnₓ₁</td>
<td>SubjEllipComma</td>
</tr>
</tbody>
</table>
Accuracy

Token accuracy: the ratio of input literals correctly labelled
Sequence accuracy: the ratio of input sequences correctly labeled.

<table>
<thead>
<tr>
<th>$n$</th>
<th>Tokens</th>
<th>Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Trees</td>
<td>Synt. Classes</td>
</tr>
<tr>
<td>1</td>
<td>57.90</td>
<td>68.60</td>
</tr>
<tr>
<td>10</td>
<td>80.05</td>
<td>89.00</td>
</tr>
<tr>
<td>20</td>
<td>82.06</td>
<td>92.68</td>
</tr>
</tbody>
</table>
Output Quality

Human-Based evaluation

- **Symb**: Without hypertagging
- **Hyb**: With hypertagging
- **Temp**: Template based system

<table>
<thead>
<tr>
<th></th>
<th>Symb/Hyb</th>
<th>Temp/Hyb</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clarity</td>
<td>0.67 / 1.22</td>
<td>0.32 / 1.95</td>
</tr>
<tr>
<td>Fluency</td>
<td>0.33 / 1.02</td>
<td>0.43 / 1.00</td>
</tr>
</tbody>
</table>

All differences between systems are statistically significant at $p < 0.001$
### Coverage and Speed

Coverage: Percentage of input for which generation produced an output  
Time (all): average time per input  
Time (gen): average time for those input for which generation succeeds

<table>
<thead>
<tr>
<th></th>
<th>Coverage</th>
<th>n = 1</th>
<th>n = 10</th>
<th>n = 20</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Trees</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Coverage</td>
<td>38.62</td>
<td>61.38</td>
<td>71.03</td>
<td></td>
</tr>
<tr>
<td>Time (all)</td>
<td>86</td>
<td>322</td>
<td>633</td>
<td></td>
</tr>
<tr>
<td>Time (gen)</td>
<td>84</td>
<td>292</td>
<td>634</td>
<td></td>
</tr>
<tr>
<td><strong>Synt.Cl.</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Coverage</td>
<td>73.69</td>
<td>88.28</td>
<td>88.28</td>
<td></td>
</tr>
<tr>
<td>Time (all)</td>
<td>162</td>
<td>603</td>
<td>603</td>
<td></td>
</tr>
<tr>
<td>Time (gen)</td>
<td>172</td>
<td>568</td>
<td>568</td>
<td></td>
</tr>
<tr>
<td><strong>Symb</strong></td>
<td>Coverage 51.00, avg time 17mn</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example Output

Input
Flight hasCurrentDepartureDate.[Date] hasCurrentArrivalDate.[Date] hasDestination.[Airport]
hasFlightTo.[Airport]] hasCarrier.[Airline] hasTicket.[AirTicket hasDateOfIssue.[Date]]

Temp I am looking for a flight. Its current departure date should be a date. The current arrival date of the flight should be a date. The destination of the flight should be an airport. The airport should have flight to an airport. The carrier of the flight should be an airline. The ticket of the flight should be an air ticket. The air ticket should have date of a date.

Symb I am looking for a flight whose current departure date should be a date and whose current arrival date should be a date and whose destination should be an airport which should have flight to an airport. Its carrier should be an airline, the ticket of the flight should be an air ticket and its date of issue should be a date.

Hyb I am looking for a flight whose current departure date should be a date, whose current arrival date should be a date and whose destination should be an airport. The airport should have flight to an airport. The carrier of the flight should be an airline. The ticket of the flight should be an air ticket whose date of issue should be a date.
The linguistic abstractions (e.g., canonical vs relative subject) encoded by the grammar permit learning a hypertagging model which
The linguistic abstractions (e.g., canonical vs relative subject) encoded by the grammar permit learning a hypertagging model which

- is more accurate than one based on grammar trees
- improves output quality by constraining output segmentation
Conclusions

Grammars of NL Syntax provide an abstraction level which can usefully be used to
Conclusions

Grammars of NL Syntax provide an abstraction level which can usefully be used to

- learn from little data (grammar induction)
- support domain independence (generic syntax, automated domain specific relation/lexicon mapping)
- improve speed, coverage and output quality (abstract syntactic classes for hypertagging)
Conclusions

Grammars of NL Syntax provide an abstraction level which can usefully be used to

- learn from little data (grammar induction)
- support domain independence (generic syntax, automated domain specific relation/lexicon mapping)
- improve speed, coverage and output quality (abstract syntactic classes for hypertagging)

This is particularly useful for NLG where parallel data/text data is hard to get
what next?

WebNLG Project (LORIA, SRI International, KRDB U. Bolzano)

- Generating from Linked Data
- Probabilistic NLG Grammar Induction (paraphrases)
- Reversible models (parsing and generation)
Thanks!