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

Data-to-Text Generation

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

- Wong and Mooney (NAACL 2007): Statistical Machine Translation
- Angeli et al (ACL 2010): Sequence of Discriminative Models
- Konstas and Lapata (ACL 2012): Probabilistic CFG

The No-NL-Syntax Approach

- Wong and Mooney (NAACL 2007): Statistical Machine Translation
- 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

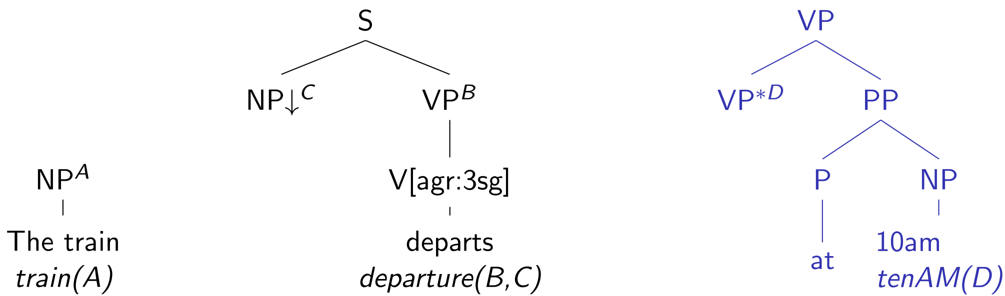
Uses a grammar of NL syntax to mediate between phrases and data

Provides an abstraction level which helps when

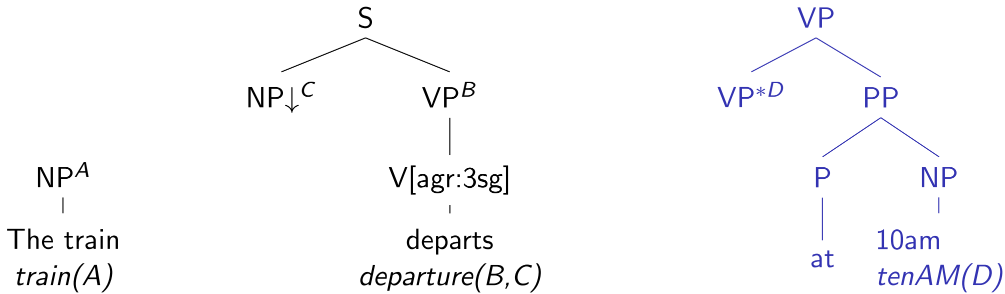
- little training data is available
  (linguistically guided grammar induction)
- domain portability is required
  (small hand written grammar + automatic lexicon acquisition)
- statistical hypertagging is needed
  (learn a small set of abstract syntactic properties rather than a larger set of hypertags)

Grammar-Based Generation

Grammar-Based Generation

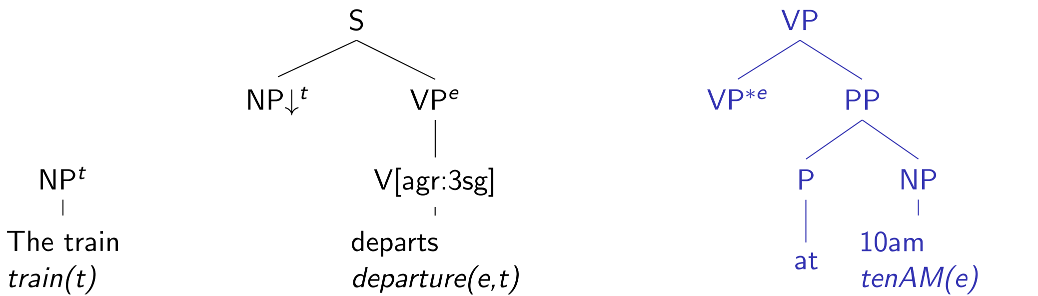


Grammar-Based Generation



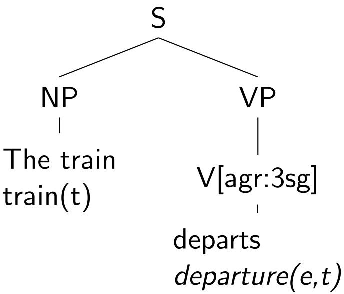
`Input data:` *train(t), departure(e,t), tenAM(e)*

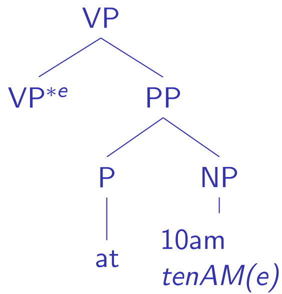
Grammar-Based Generation



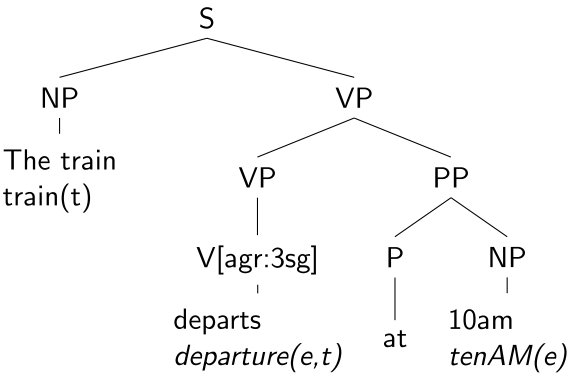
`Input data:` *train(t), departure(e,t), tenAM(e)*

Grammar-Based Generation

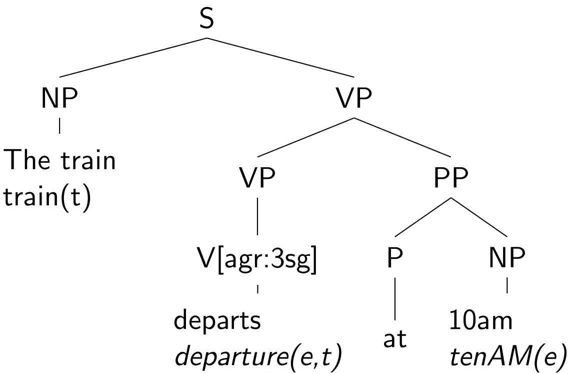




Grammar-Based Generation



Grammar-Based Generation



`The train departs at 10am`

Separating Grammar from Lexicon

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

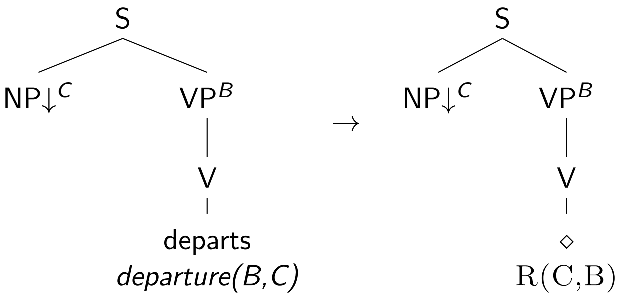
- abstract over lexical items in the grammar

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



Semantics: *departure*
Tree: nx0V
Syntax: CanonicalSubject
Anchor: *departs*

Semantics: *arrival*
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

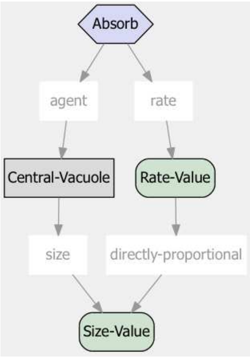
KBGen 2012: an international shared task

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.



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*)

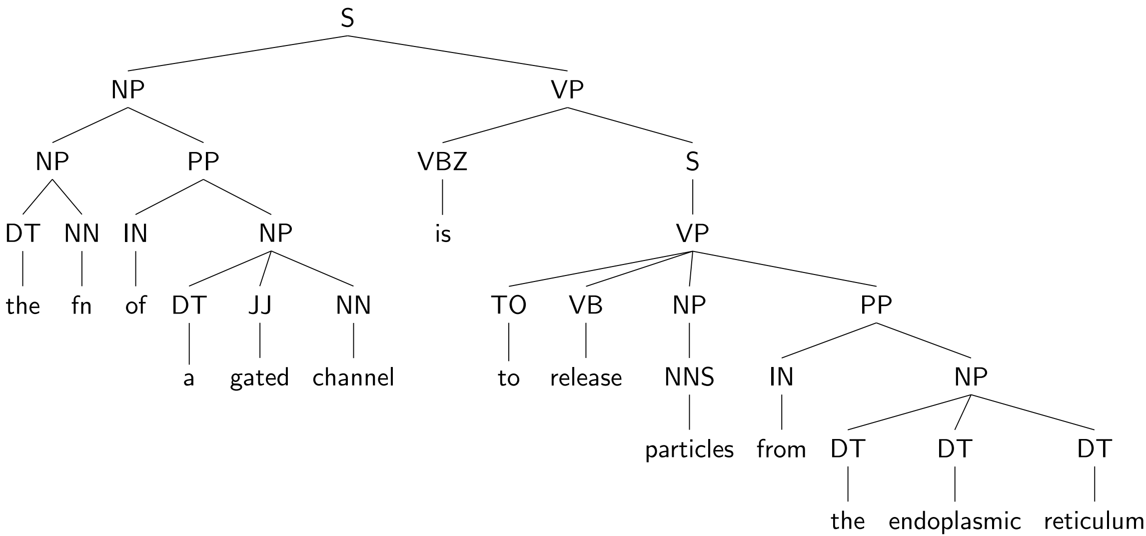
Generalise grammar by

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

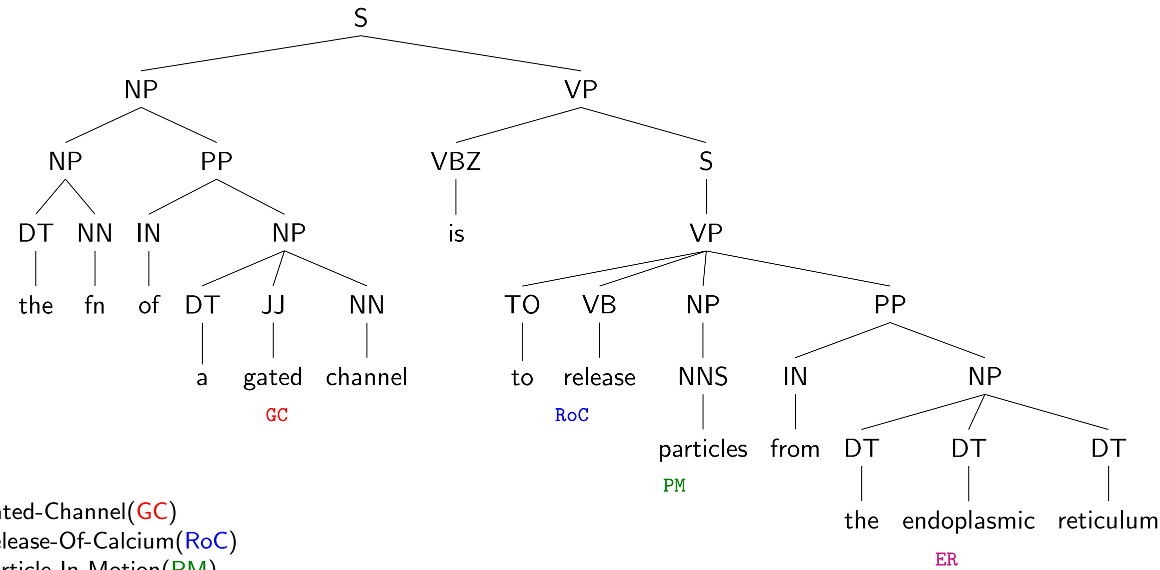
Example

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*

Parse Tree and Variable Projection

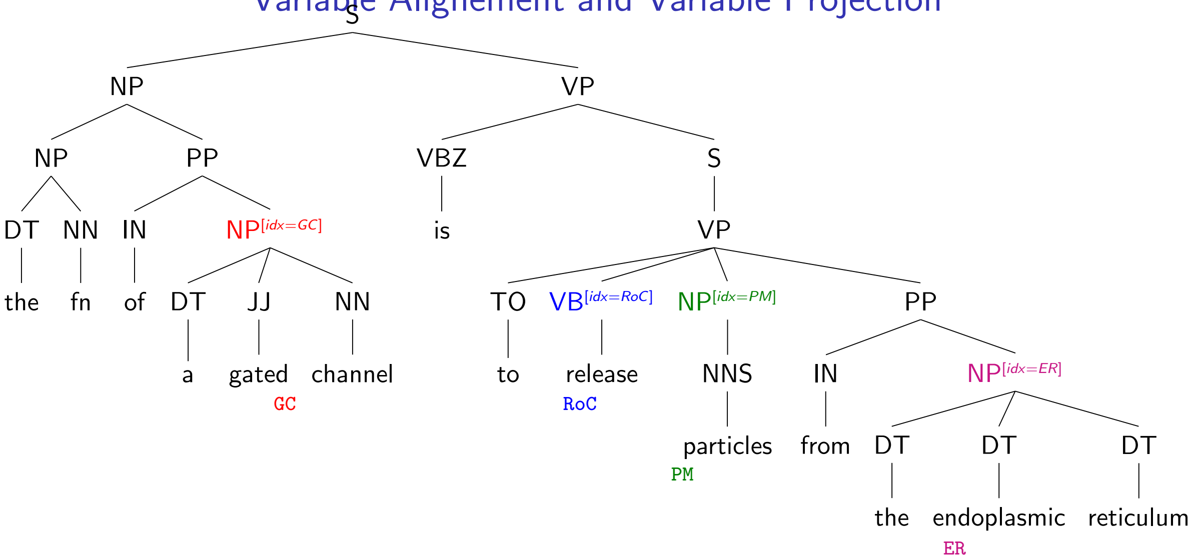


Parse Tree and Variable Projection



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

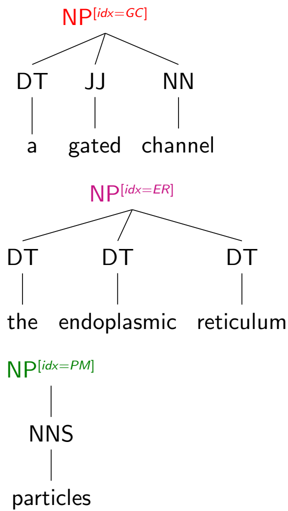
Variable Alignement and Variable Projection



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

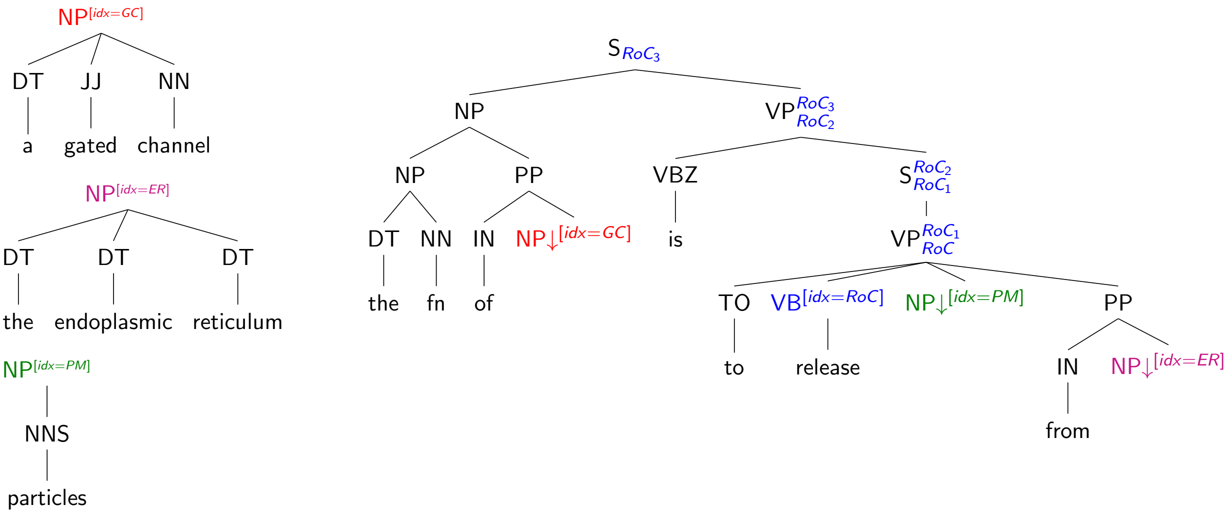
FB-LTAG Trees Extraction

- First, the subtrees associated with Entities are extracted.
- Second, the subtrees associated with Events are extracted.

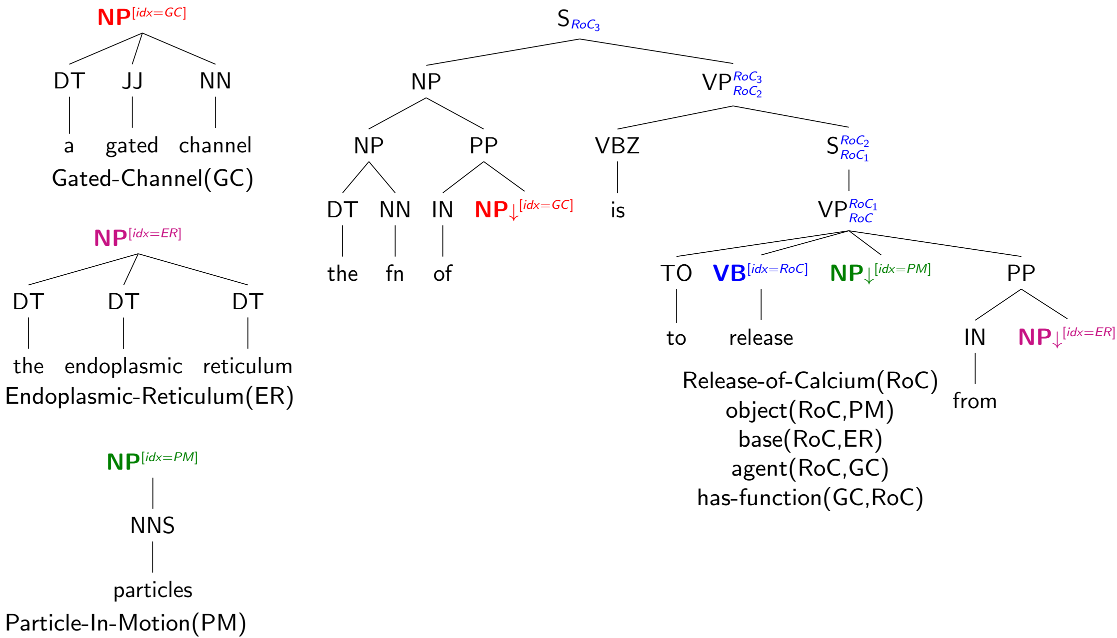


FB-LTAG Trees Extraction

- First, the subtrees associated with Entities are extracted.
- Second, the subtrees associated with Events are extracted.

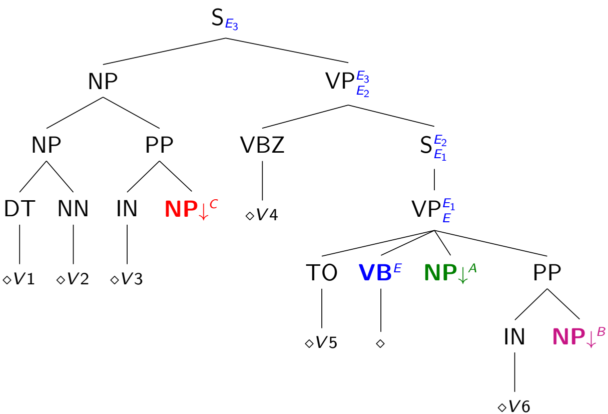


Putting Syntax and Semantics Together



Lexicon/Grammar Abstraction

Tree Schema

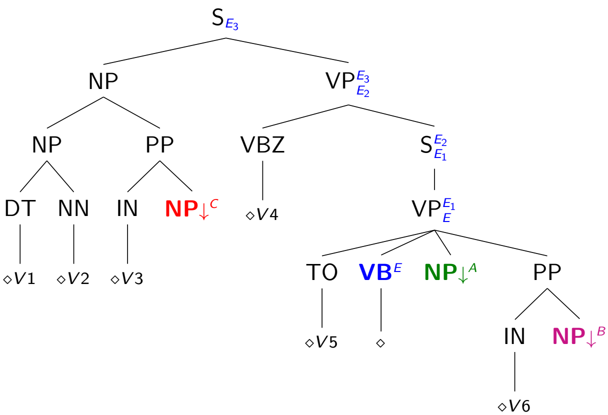


Lexicon

	Semantics	Anchor	CoAnchors
Ex.1	release(E), object(E,A), base(E,B), agent(E,C), has-function(C,E)	release	V1→The V2→function V3→of V4→is V5→to V6→from

Lexicon/Grammar Abstraction

Tree Schema

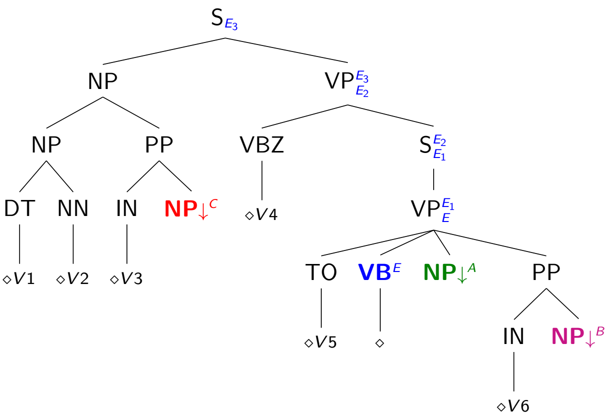


Lexicon

	Semantics	Anchor	CoAnchors
Ex.1	release(E), object(E,A), base(E,B), agent(E,C), has-function(C,E)	release	V1→The V2→function V3→of V4→is V5→to V6→from
Ex.2	carry(E) object(E,A) instrument(E,B) agent(E,C) has-function(C,E)	carry	V1→The V2→function V3→of V4→is V5→to V6→using

Lexicon/Grammar Abstraction

Tree Schema



Lexicon

	Semantics	Anchor	CoAnchors
Ex.1	release(E), object(E,A), base(E,B), agent(E,C), has-function(C,E)	release	V1→The V2→function V3→of V4→is V5→to V6→from
Ex.2	carry(E) object(E,A) instrument(E,B) agent(E,C) has-function(C,E)	carry	V1→The V2→function V3→of V4→is V5→to V6→using

- Ex.1: The function of C is to release A from B.
- Ex.2: The function of C is to carry A using B.

Lexicon Expansion

release(E)

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

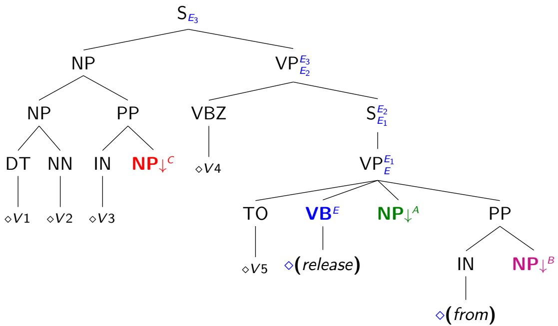
agent(E,C), has-function(C,E)

Lexicon Expansion

release(E)

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

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

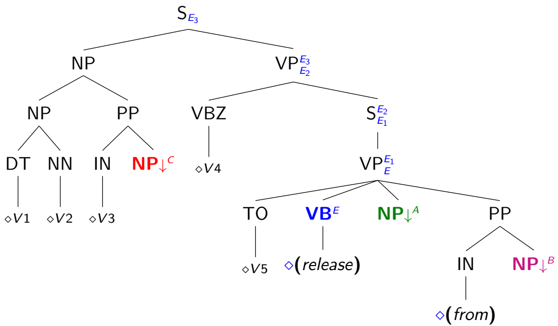


Lexicon Expansion

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.

Lexicon Expansion

release(E)

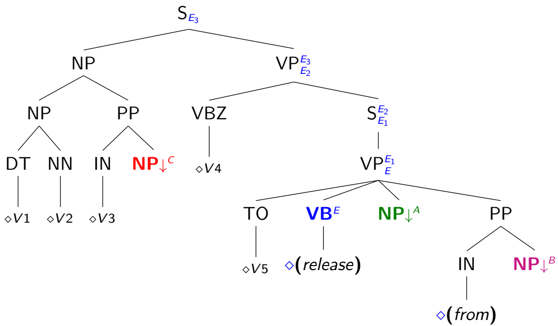
object(E,A), base(E,B)

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

generate(E)

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

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



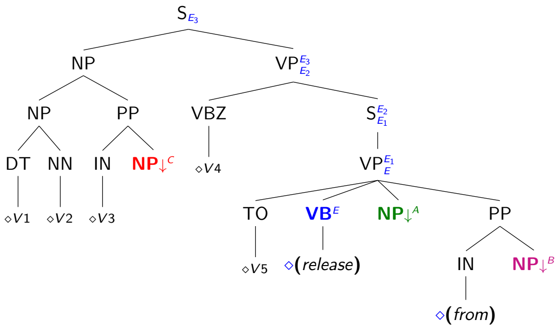
The function of C is to release A from B.

Lexicon Expansion

release(E)

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

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

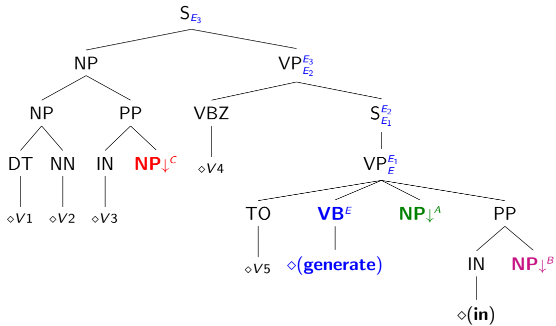


The function of C is to release A from B.

generate(E)

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

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

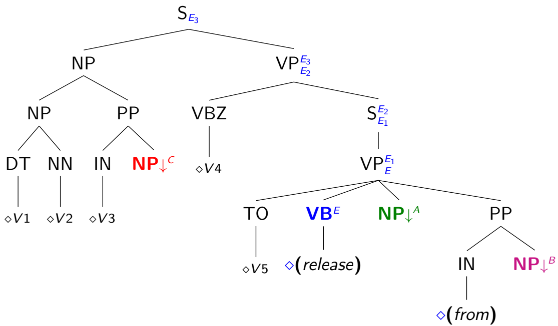


Lexicon Expansion

release(E)

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

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

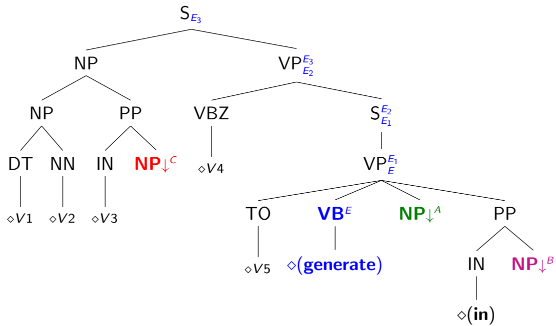


The function of C is to release A from B.

generate(E)

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

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



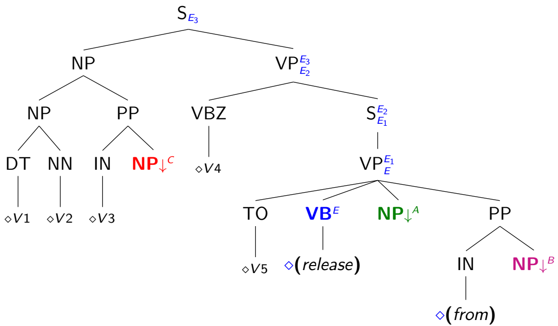
The function of C is to generate A in B.

Lexicon Expansion

release(E)

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

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

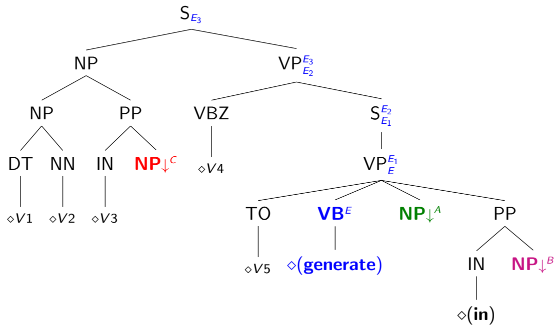


The function of C is to release A from B.

generate(E)

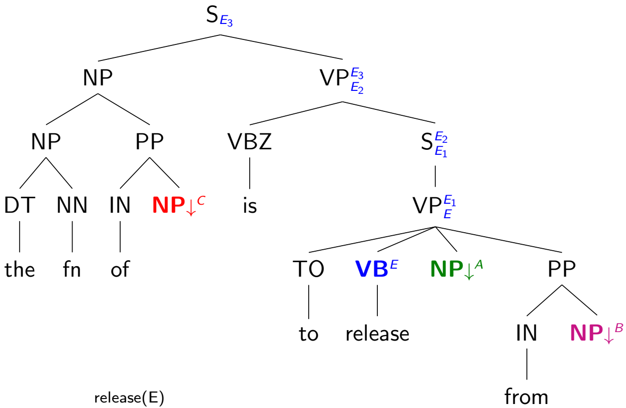
object(E,A), base(E,B)

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



The function of C is to generate A in B.

Grammar Expansion



release(E)

object(E,A)

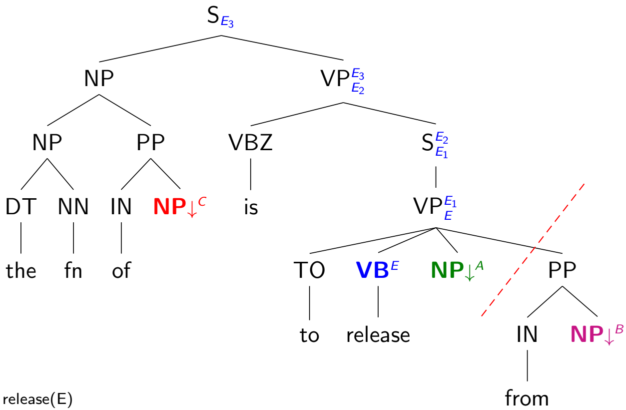
agent(E,C)

has-function(C,E)

base(E,B)

Grammar Expansion

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



release(E)

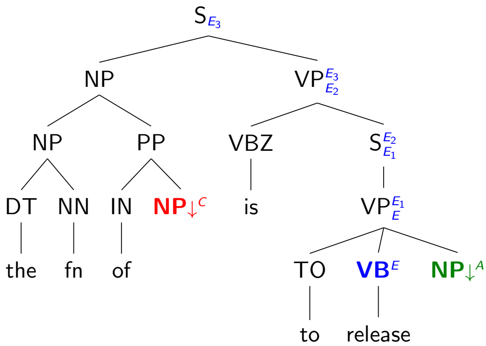
object(E,A)

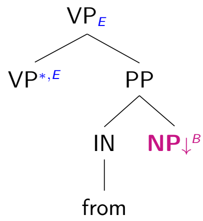
agent(E,C)

has-function(C,E)

$\boxed{\texttt{base(E,B)}}$

Grammar Expansion





base(E,B)

release(E)

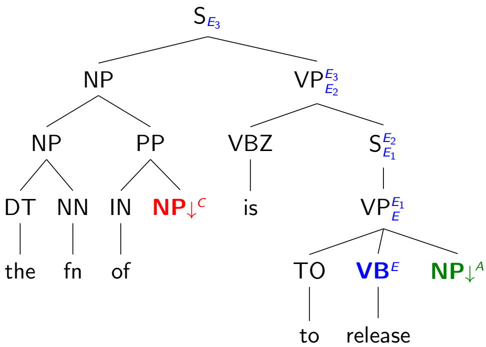
object(E,A)

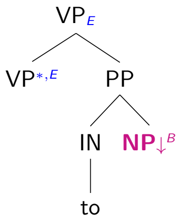
agent(E,C)

has-function(C,E)

The function of X is to release Y

Grammar Expansion





target(E,B)

release(E)

object(E,A)

agent(E,C)

has-function(C,E)

target(E,C)

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

System	BLEU Score	Approach
UDEL	0.32	Hand-written Rewrite Rules
LOR-KBGen	0.29	Automatically Induced Grammar
IMS	0.12	Automatically Induced Probabilistic Grammar

Human Evaluation

	Fluency				Grammaticality				Meaning Similarity		
System	Mean				Mean				Mean		
UDEL	4.36	B			4.48	B			3.69	A	
LOR-KBGen	3.45		C		3.55		C		3.65	A	
IMS	1.91			D	2.05			D	1.31		B

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
EACL 2014. Gothenburg, Sweden, April 2014.

Interactive Query Formulation

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

b. *Man* (substitute concept)
I am looking for **a man**

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

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

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

f. *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
  - perfoms 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

$$\textsc{equipped}\text{W}\textsc{ith} \rightarrow \text{equipped/VBD with/IN}$$

Automatic Lexicon Acquisition

Tokenize and tag relations

$$\textsc{equipped}\text{W}\textsc{ith} \rightarrow \text{equipped/VBD with/IN}$$

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

Automatic Lexicon Acquisition

Tokenize and tag relations

$$\textsc{equipped}\mathrm{W}\textsc{ith} \rightarrow \text{equipped/VBD with/IN}$$

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

$$\text{equipped/VBD with/IN} \rightarrow \text{nx0Vpnx1}$$

The nx0Vpnx1 Family

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

Example	Tree Name
hline NP_0 should be equipped with NP_1	nx0VVVpnx1
It_0 should be equipped with NP_1	PRO0VVVpnx1
and NP_0 should be equipped with NP_1	sCONJnx0VVVpnx1
and it_0 should be equipped with NP_1	sCONJPRO0VVVpnx1
NP_0 which should be equipped with NP_1	W0nx0VVVpnx1
NP_0 (...) and which should be equipped with NP_1	ANDWHnx0VVVpnx1
NP_0 (...), which should be equipped with NP_1	COMMAWHnx0VVVpnx1
NP_0 equipped with NP_1	betanx0VPpnx1
NP_0 (...) and equipped with NP_1	betanx0ANDVPpnx1
NP_0 (...), equipped with NP_1	betanx0COMMAVPpnx1
NP_1 with which NP_0 should be equipped	W1pnx1nx0VV
NP_0 (equipped with X) and with NP_1	betavx0ANDVVVpnx1
NP_0 (equipped with X), with NP_1	betavx0COMMAVVVpnx1

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%

Template vs. Grammar-Generated Queries

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)

	Fluency	Clarity
Grammar	**19.76**	6.87
Templates	7.2	8.57

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).

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).

CarDealer	locatedIn	City	sell	Car	runOn	Diesel
nx	**betanx0VPpnx1**	**nx**	**ANDWHnx0VVnx1**	**nx**	**nx0VVpnx1**	**nx**
	PRO0VVVpnx1		PRO0VVnx1		...	
	sCONJnx0VVVpnx1		sCONJnx0VVnx1		...	
	sCONJPRO0VVVpnx1		sCONJPRO0VVnx1		...	
	W0nx0VVVpnx1		W0nx0VVnx1		...	
	ANDWHnx0VVVpnx1		ANDWHnx0VVnx1		...	
	COMMAWHnx0VVVpnx1		COMMAWHnx0VVnx1		...	
	betanx0VPpnx1		betanx0VPnx1		...	
	betanx0ANDVPpnx1		betanx0ANDVPnx1		...	
	betavx0COMMAVVVpnx1		betavx0COMMAVVnx1		...	

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).

CarDealer	locatedIn	City	sell	Car	runOn	Diesel
nx	**betanx0VPpnx1**	**nx**	**ANDWHnx0VVnx1**	**nx**	**nx0VVpnx1**	**nx**
	PRO0VVVpnx1		PRO0VVnx1		...	
	sCONJnx0VVVpnx1		sCONJnx0VVnx1		...	
	sCONJPRO0VVVpnx1		sCONJPRO0VVnx1		...	
	W0nx0VVVpnx1		W0nx0VVnx1		...	
	ANDWHnx0VVVpnx1		ANDWHnx0VVnx1		...	
	COMMAWHnx0VVVpnx1		COMMAWHnx0VVnx1		...	
	betanx0VPpnx1		betanx0VPnx1		...	
	betanx0ANDVPpnx1		betanx0ANDVPnx1		...	
	betavx0COMMAVVVpnx1		betavx0COMMAVVnx1		...	

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:

$$y^* = argmax_y P(y \mid x)$$

Learn the mapping between observed input features and hidden syntactic classes using a Linear-chain Conditional Random Field (CRF) model

Training Data, Trees and Linguistic Abstractions

Training corpus: 145 ⟨ data,sequence of tree names ⟩ pairs
59 tree names

Training Data, Trees and Linguistic Abstractions

Training corpus: 145 ⟨ data,sequence of tree names ⟩ pairs
59 tree names

E.g.,

CarDealer	locatedIn	City	sell	Car	runOn	Diesel
nx	betanx0VPpnx1	nx	ANDWHnx0VVnx1	nx	nx0VVpnx1	nx

Training Data, Trees and Linguistic Abstractions

Training corpus: 145 ⟨ data,sequence of tree names ⟩ pairs
59 tree names

E.g.,

CarDealer	locatedIn	City	sell	Car	runOn	Diesel
nx	betanx0VPpnx1	nx	ANDWHnx0VVnx1	nx	nx0VVpnx1	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*)

Example	Tree Name	Syntactic Class
NP_0 should be equipped with NP_1	nx0VVVpnx1	Canonical
It_0 should be equipped with NP_1	PRO0VVVpnx1	SubjPro
and NP_0 should be equipped with NP_1	sCONJnx0VVVpnx1	Scoord
and it_0 should be equipped with NP_1	sCONJPRO0VVVpnx1	ScoordSubjPro
NP_0 which should be equipped with NP_1	W0nx0VVVpnx1	SubjRel
NP_0 (...) and which should be equipped with NP_1	ANDWHnx0VVVpnx1	SubjRelAnd
NP_0 (...), which should be equipped with NP_1	COMMAWHnx0VVVpnx1	SubjRelComma
NP_0 equipped with NP_1	betanx0VPpnx1	ParticipialOrGerund
NP_0 (...) and equipped with NP_1	betanx0ANDVPpnx1	ParticipialOrGerundAnd
NP_0 (...), equipped with NP_1	betanx0COMMAVPpnx1	ParticipialOrGerundComma
NP_1 with which NP_0 should be equipped	W1pnx1nx0VV	PObjRel
NP_0 (equipped with X) and with NP_1	betavx0ANDVVVpnx1	SubjEllipAnd
NP_0 (equipped with X), with NP_1	betavx0COMMAVVVpnx1	SubjEllipComma

Accuracy

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

	Tokens		Sequence	
n	Trees	Synt. Classes	Trees	Synt. Classes
1	57.90	68.60	33.33	48.33
10	80.05	89.00	57.86	76.48
20	82.06	92.68	61.33	82.10

Output Quality

Human-Based evaluation

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

	Symb/Hyb	Temp/Hyb
Clarity	0.67 / 1.22	0.32 / 1.95
Fluency	0.33 / 1.02	0.43 / 1.00

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

		$n = 1$	$n = 10$	$n = 20$
Trees	Coverage	38.62	61.38	71.03
	Time (all)	86	322	633
	Time (gen)	84	292	634
Synt.Cl.	Coverage	73.69	88.28	88.28
	Time (all)	162	603	603
	Time (gen)	172	568	568
Symb	Coverage 51.00, avg time 17mn			

Example Output

Input Query	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.

Summary

The linguistic abstractions (e.g., canonical vs relative subject) encoded by the grammar permit learning a hypertagging model which

Summary

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!