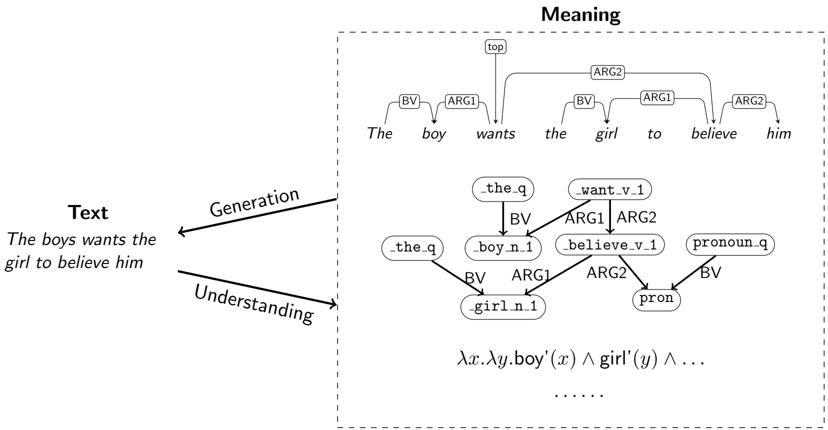
L98: Introduction to Computational Semantics

Lecture 8: Graph-Based Representations for Semantics

Weiwei Sun and Simone Teufel

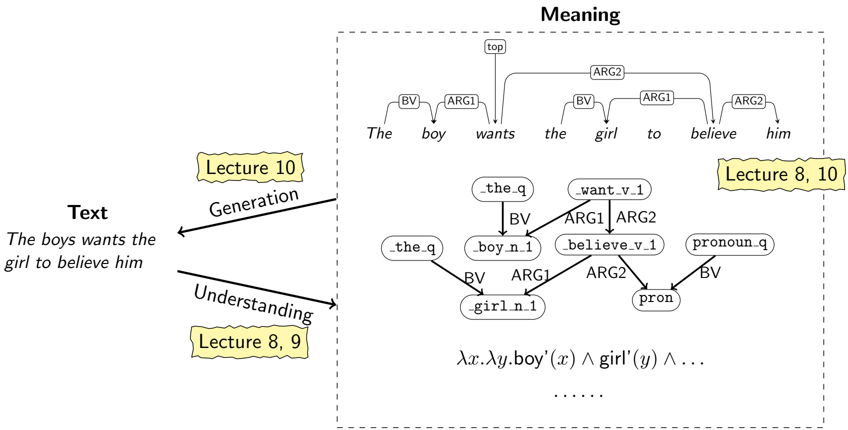
Natural Language and Information Processing Research Group
Department of Computer Science and Technology
University of Cambridge

Lent 2021/22



Lecture 8: Graph-Based Representations for Semantics

1. Generalised quantifiers
2. Logico-semantic graphs
3. Clause Union
4. Functor, argument and bilinearity



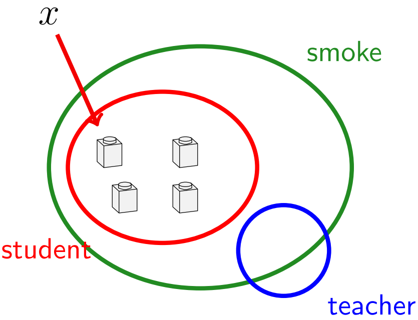
Lecture 8: Graph-Based Representations for Semantics

1. Generalised quantifiers
2. Logico-semantic graphs
3. Clause Union
4. Functor, argument and bilinearity

Generalised Quantifiers

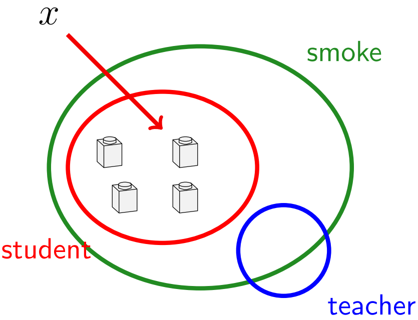
Quantification over individuals/sets

- What is ⟦*every student smokes*⟧? $\forall x(\text{student'}(x) \rightarrow \text{smoke'}(x))$
- What is ⟦*some students smoke*⟧? $\exists x(\text{student'}(x) \wedge \text{smoke'}(x))$



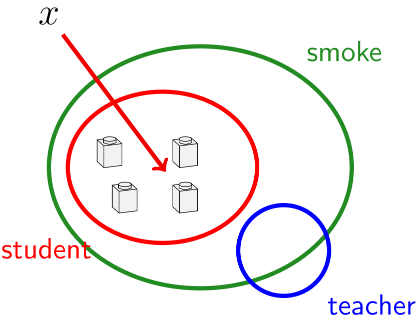
Quantification over individuals/sets

- What is ⟦*every student smokes*⟧? $\forall x(\text{student'}(x) \rightarrow \text{smoke'}(x))$
- What is ⟦*some students smoke*⟧? $\exists x(\text{student'}(x) \wedge \text{smoke'}(x))$



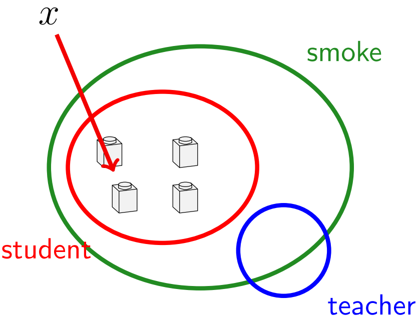
Quantification over individuals/sets

- What is ⟦*every student smokes*⟧? $\forall x(\text{student'}(x) \rightarrow \text{smoke'}(x))$
- What is ⟦*some students smoke*⟧? $\exists x(\text{student'}(x) \wedge \text{smoke'}(x))$



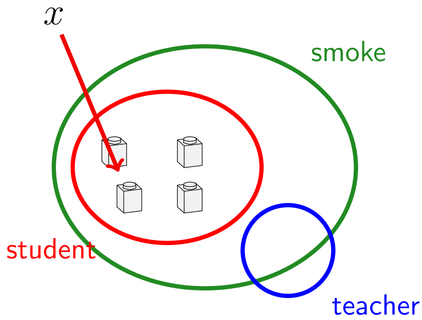
Quantification over individuals/sets

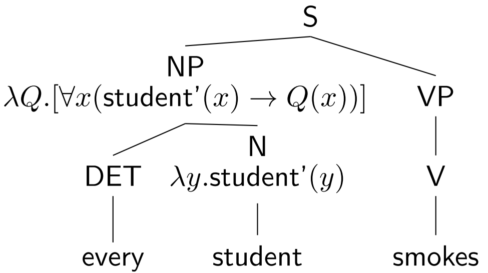
- What is ⟦*every student smokes*⟧? $\forall x(\text{student'}(x) \rightarrow \text{smoke'}(x))$
- What is ⟦*some students smoke*⟧? $\exists x(\text{student'}(x) \wedge \text{smoke'}(x))$



Quantification over individuals/sets

- What is ⟦*every student smokes*⟧? $\forall x(\text{student'}(x) \rightarrow \text{smoke'}(x))$
- What is ⟦*some students smoke*⟧? $\exists x(\text{student'}(x) \wedge \text{smoke'}(x))$



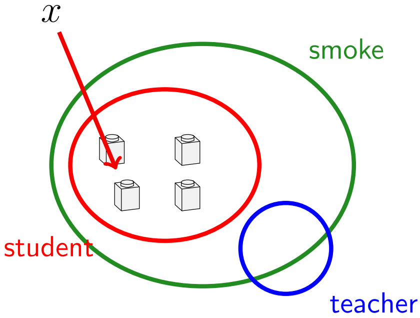


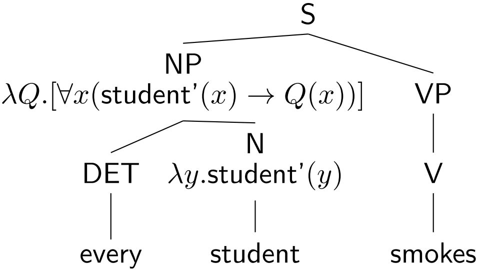
$$[\![\textit{every}]\!] = \lambda P.[\lambda Q.[\forall x(P(x) \rightarrow Q(x))]]$$

$$[\![\textit{some}]\!] = \lambda P.[\lambda Q.[\exists x(P(x) \wedge Q(x))]]$$

Quantification over individuals/sets

- What is $[\![$*every student smokes*$]\!]$? $\forall x(\text{student'}(x) \rightarrow \text{smoke'}(x))$
- What is $[\![$*some students smoke*$]\!]$? $\exists x(\text{student'}(x) \wedge \text{smoke'}(x))$





$$[\![\textit{every}]\!] = \lambda P.[\lambda Q.[\forall x(P(x) \rightarrow Q(x))]]$$

$$[\![\textit{some}]\!] = \lambda P.[\lambda Q.[\exists x(P(x) \wedge Q(x))]]$$

- what is the type of the NP (*every student*)?
- Is it $\langle \mathbf{e}, \langle \mathbf{e}, \mathbf{t} \rangle \rangle$? Or $\langle \langle \mathbf{e}, \mathbf{t} \rangle, \mathbf{t} \rangle$?

Generalised quantifiers

$$\langle\langle \mathbf{e}, \mathbf{t}\rangle, \mathbf{t}\rangle$$

- *Every student* smokes.
  the bucket associated with *student* is the only element in the bucket associated with *every student*.
- Assume we have two students in our world model:

$$[\![\textit{every student}]\!] = \begin{bmatrix} \begin{bmatrix} t & \mapsto & 1 \\ j & \mapsto & 1 \end{bmatrix} & \mapsto & 1 \\ \begin{bmatrix} t & \mapsto & 1 \\ j & \mapsto & 0 \end{bmatrix} & \mapsto & 0 \\ \begin{bmatrix} t & \mapsto & 0 \\ j & \mapsto & 1 \end{bmatrix} & \mapsto & 0 \\ \begin{bmatrix} t & \mapsto & 0 \\ j & \mapsto & 0 \end{bmatrix} & \mapsto & 0 \end{bmatrix}$$

Generalised quantifiers

- *At least three students* smoke.
  every bucket in the bucket associated with *at least three students* contains at least three students.
- *nothing*, *most*, *many*, *half* . . .
- FOPL is not expressive enough.

A convenient notation

- $\forall x$(student'$(x) \rightarrow$ smoke'(x))
- every'$(x,$ student'$(x),$ smoke'$(x))$
- $\exists x$(student'$(x) \wedge$ smoke'(x))
- some'$(x,$ student'$(x),$ smoke'$(x))$

at_least_three'$(x,$ student'$(x),$ smoke'$(x))$

Truth conditions for generalized determiners

Determiner	Truth conditions
$[\![\textit{every}]\!](P)(Q)$	$P \subseteq Q$
$[\![\textit{some}]\!](P)(Q)$	$P \cap Q \neq \emptyset$
$[\![\textit{no}]\!](P)(Q)$	$P \cap Q = \emptyset$
$[\![\textit{three}]\!](P)(Q)$	$\|P \cap Q\| = 3$
$[\![\textit{less than three}]\!](P)(Q)$	$\|P \cap Q\| < 3$
$[\![\textit{at least three}]\!](P)(Q)$	$\|P \cap Q\| \geq 3$
$[\![\textit{most}]\!](P)(Q)$	$\|P \cap Q\| \geq \|P - Q\|$
$[\![\textit{few}]\!](P)(Q)$	$\|P \cap Q\| \ll \|P - Q\|$

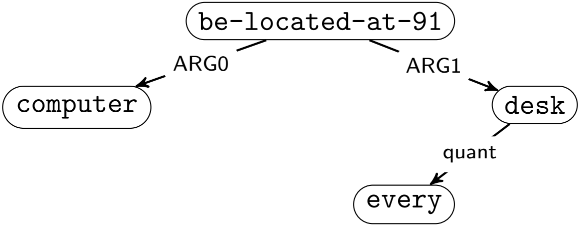
Truth conditions for generalized determiners

Determiner	Truth conditions
$[\![\textit{every}]\!](P)(Q)$	$P \subseteq Q$
$[\![\textit{some}]\!](P)(Q)$	$P \cap Q \neq \emptyset$
$[\![\textit{no}]\!](P)(Q)$	$P \cap Q = \emptyset$
$[\![\textit{three}]\!](P)(Q)$	$\|P \cap Q\| = 3$
$[\![\textit{less than three}]\!](P)(Q)$	$\|P \cap Q\| < 3$
$[\![\textit{at least three}]\!](P)(Q)$	$\|P \cap Q\| \geq 3$
$[\![\textit{most}]\!](P)(Q)$	$\|P \cap Q\| \geq \|P - Q\|$
$[\![\textit{few}]\!](P)(Q)$	$\|P \cap Q\| \ll \|P - Q\|$

$[\![\textit{the}]\!]$

Logico-Semantic Graphs

Abstract Meaning Representation



a computer is on every desk.

- There are several projects working on developing "conceptual graphs" as comprehensive meaning representations. We introduce Abstract Meaning Representation and English Resource Semantics.
- Basic units are "concepts" as well as asymmetric "links/dependency" between such concepts.

Abstract Meaning Representation

- AMR is a semantic representation aimed at large-scale human annotation in order to build a giant semantics bank.
- We do a practical, replicable amount of abstraction (limited canonicalization).
- Capture many aspects of meaning in a single simple data structure.
- AMR annotations are not tied to individual words or any syntactic derivation

PENMAN notation

The dog is eating a bone

```
(e / eat-01
   :ARG0 (d / dog)
   :ARG1 (b / bone))
```

slide from https://github.com/nschneid/amr-tutorial

Abstract Meaning Representation

- AMR is a semantic representation aimed at large-scale human annotation in order to build a giant semantics bank.
- We do a practical, replicable amount of abstraction (limited canonicalization).
- Capture many aspects of meaning in a single simple data structure.
- AMR annotations are not tied to individual words or any syntactic derivation

PENMAN notation

The dog is eating a bone

```
(e / eat-01
   :ARG0 (d / dog)
   :ARG1 (b / bone))
```

slide from https://github.com/nschneid/amr-tutorial

Inter-annotator agreement: 70–80% SMATCH

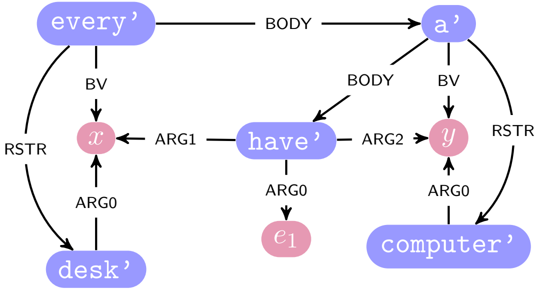
There is nothing as practical as a good theory.

Different representations of logical forms

- *Every desk has a computer* — more in lecture 13
- $\forall x(\text{desk'}(x) \rightarrow (\exists y(\text{computer'}(y) \wedge \text{have'}(e, x, y))))$
- $\text{every'}(x, \text{desk'}(x), \text{a'}(y, \text{computer'}(y), \text{have'}(e, x, y)))$

Different representations of logical forms

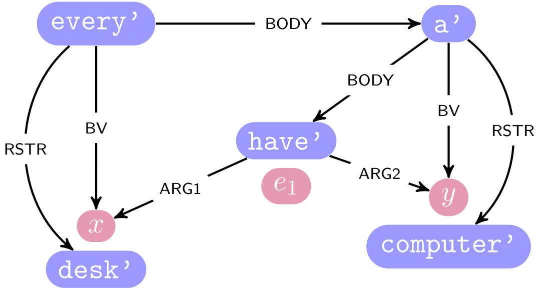
- *Every desk has a computer* more in lecture 13
- $\forall x(\text{desk'}(x) \rightarrow (\exists y(\text{computer'}(y) \wedge \text{have'}(e, x, y))))$
- $\text{every'}(x, \text{desk'}(x), \text{a'}(y, \text{computer'}(y), \text{have'}(e, x, y)))$



ARG0: which word "introduces" a variable.

Different representations of logical forms

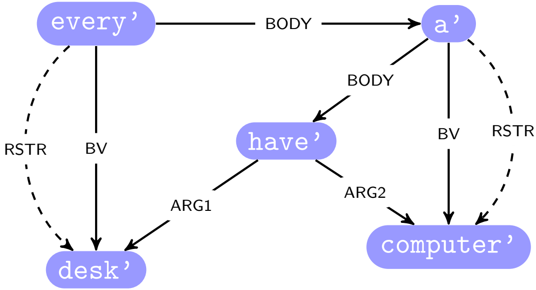
- *Every desk has a computer* — more in lecture 13
- $\forall x(\text{desk'}(x) \rightarrow (\exists y(\text{computer'}(y) \wedge \text{have'}(e, x, y))))$
- $\text{every'}(x, \text{desk'}(x), \text{a'}(y, \text{computer'}(y), \text{have'}(e, x, y)))$



ARG0: which word "introduces" a variable.

Different representations of logical forms

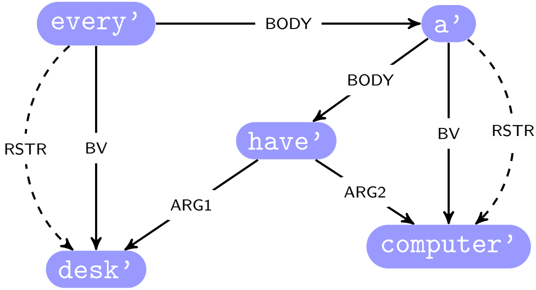
- *Every desk has a computer* — more in lecture 13
- $\forall x(\text{desk'}(x) \rightarrow (\exists y(\text{computer'}(y) \wedge \text{have'}(e, x, y))))$
- $\text{every'}(x, \text{desk'}(x), \text{a'}(y, \text{computer'}(y), \text{have'}(e, x, y)))$



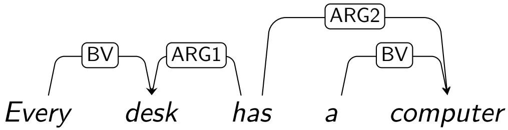
ARG0: which word "introduces" a variable.

Different representations of logical forms

- *Every desk has a computer* — more in lecture 13
- $\forall x(\text{desk'}(x) \rightarrow (\exists y(\text{computer'}(y) \wedge \text{have'}(e, x, y))))$
- $\text{every'}(x, \text{desk'}(x), \text{a'}(y, \text{computer'}(y), \text{have'}(e, x, y)))$



ARG0: which word "introduces" a variable.



Bi-lexical semantic dependency graphs

- Projecting "concept nodes" to "words".
- Relations between "concepts" $\Rightarrow$ bi-lexical semantic dependencies
- Reasonably good though not as expressive as conceptual graphs.

Bi-lexical semantic dependency graphs

- Projecting "concept nodes" to "words".
- Relations between "concepts" $\Rightarrow$ bi-lexical semantic dependencies
- Reasonably good though not as expressive as conceptual graphs.

Discussion on weakness of bi-lexical semantic dependency graphs

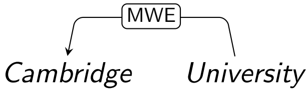
Bi-lexical semantic dependency graphs

- Projecting "concept nodes" to "words".
- Relations between "concepts" ⇒ bi-lexical semantic dependencies
- Reasonably good though not as expressive as conceptual graphs.

Discussion on weakness of bi-lexical semantic dependency graphs

What are the triggers of concepts?

- MWE:



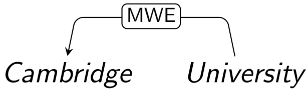
Bi-lexical semantic dependency graphs

- Projecting "concept nodes" to "words".
- Relations between "concepts" $\Rightarrow$ bi-lexical semantic dependencies
- Reasonably good though not as expressive as conceptual graphs.

Discussion on weakness of bi-lexical semantic dependency graphs

What are the triggers of concepts?

- MWE:



- Construction: *The emails won't reply themselves.*

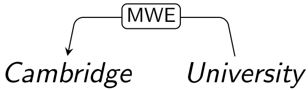
Bi-lexical semantic dependency graphs

- Projecting “concept nodes” to “words”.
- Relations between “concepts” ⇒ bi-lexical semantic dependencies
- Reasonably good though not as expressive as conceptual graphs.

Discussion on weakness of bi-lexical semantic dependency graphs

What are the triggers of concepts?

- MWE:



- Construction: *The emails won’t reply themselves.*

Modification

SemBanking in Natural Language Processing

- compositionally
  - comprehensiveness
    consistency
    scalability
  - manually-annotated
    - **PropBank** (Kingsbury & Palmer, 2002)
    - **FrameNet** (Baker et al., 1998)
  - grammar-based
    - **English Resource Semantics** (Oepen et al., 2004)
    - **Groningen Meaning Bank** (Basile et al., 2012)
- non-compositionally
  - **Abstract Meaning Representation** (Banarescu et al., 2013)
  - **QA-SRL** (He et al., 2015)

Bender, E.M., Flickinger, D., Oepen, S., Packard, W. and Copestake, A. Layers of interpretation: On grammar and compositionality. ICWS 2015.

English Resource Semantics (ERS)

LinGO English Resource Grammar (Flickinger, 2000; Flickinger et al., 2017)

- Hand-designed computational grammar for English based on Head-driven Phrase Structure Grammar;
- declarative, unification-based: parsing and realization; multiple engines;
- 25^+ person years; coverage of 85–95 % of running text across domains;
- underspecified meaning representation in Minimal Recursion Semantics

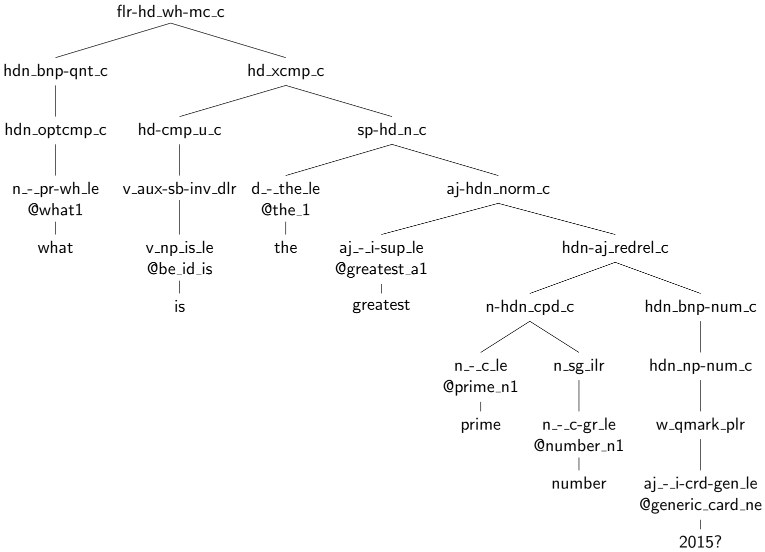
LinGO Redwoods Treebank (Oepen & Lønning, 2006; Flickinger et al., 2012)

- Grammar-based annotation: select rather than generate 'correct' analysis
- version 1214: some 85,000 annotated sentences, six^+ different domains;
- including Sections 00–21 from the venerable WSJ Corpus; sub-set of Brown Corpus; Wikipedia; tourism; ecommerce; transcribed speech;
- MRS plus various graph-based formats.
- inter-annotator agreement of 0.94 EDM (elementary dependency match);

Online demo

- https://delph-in.github.io/delphin-viz/demo/
- http://erg.delph-in.net/

Example: Derivation



Example: MRS

What is the greatest prime number below 2015?

$$
\begin{aligned}
\langle\, & h_1, \\
& \left|\begin{array}{l}
h_4\text{:thing}(\text{ARG0 } x_5), \\
h_6\text{:which\_q}(\text{ARG0 } x_5, \text{ RSTR } h_7, \text{ BODY } h_8), \\
h_2\text{:\_be\_v\_id}(\text{ARG0 } e_3, \text{ ARG1 } x_9, \text{ ARG2 } x_5), \\
h_{10}\text{:\_the\_q}(\text{ARG0 } x_9, \text{ RSTR } h_{12}, \text{ BODY } h_{11}), \\
h_{13}\text{:\_great\_a\_for}(\text{ARG0 } e_{14}, \text{ ARG1 } x_9), \\
h_{13}\text{:superl}(\text{ARG0 } e_{15}, \text{ ARG1 } e_{14}), \\
h_{13}\text{:compound}(\text{ARG0 } e_{17}, \text{ ARG1 } x_9, \text{ ARG2 } x_{16}\{\}), \\
h_{18}\text{:udef\_q}(\text{ARG0 } x_{16}, \text{ RSTR } h_{19}, \text{ BODY } h_{20}), \\
h_{21}\text{:\_prime\_n\_1}(\text{ARG0 } x_{16}), \\
h_{13}\text{:\_number\_n\_of}(\text{ARG0 } x_9), \\
h_{13}\text{:\_below\_p}(\text{ARG0 } e_{22}, \text{ ARG1 } x_9, \text{ ARG2 } x_{23}), \\
h_{24}\text{:number\_q}(\text{ARG0 } x_{23}, \text{ RSTR } h_{25}, \text{ BODY } h_{26}), \\
h_{27}\text{:card}(\text{ARG0 } x_{23}, \text{ ARG1 } i_{28}, \text{ CARG } \textit{2015})
\end{array}\right| \\
& \{\, h_{25} =_q h_{27},\ h_{19} =_q h_{21},\ h_{12} =_q h_{13},\ h_7 =_q h_4,\ h_1 =_q h_2 \,\}\,\rangle
\end{aligned}
$$

Example: MRS

What is the greatest prime number below 2015?

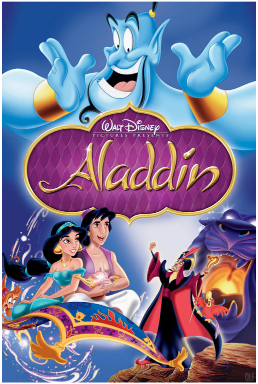
$$
\langle\, h_1,\; \Big|\; \begin{array}{l}
h_4\text{:thing(ARG0 } x_5\text{)},\\
h_6\text{:which\_q(ARG0 } x_5\text{, RSTR } h_7\text{, BODY } h_8\text{)},\\
h_2\text{:\_be\_v\_id(ARG0 } e_3\text{, ARG1 } x_9\text{, ARG2 } x_5\text{)},\\
h_{10}\text{:\_the\_q(ARG0 } x_9\text{, RSTR } h_{12}\text{, BODY } h_{11}\text{)},\\
h_{13}\text{:\_great\_a\_for(ARG0 } e_{14}\text{, ARG1 } x_9\text{)},\\
h_{13}\text{:superl(ARG0 } e_{15}\text{, ARG1 } e_{14}\text{)},\\
h_{13}\text{:compound(ARG0 } e_{17}\text{, ARG1 } x_9\text{, ARG2 } x_{16}\{\}\text{)},\\
h_{18}\text{:udef\_q(ARG0 } x_{16}\text{, RSTR } h_{19}\text{, BODY } h_{20}\text{)},\\
h_{21}\text{:\_prime\_n\_1(ARG0 } x_{16}\text{)},\\
h_{13}\text{:\_number\_n\_of(ARG0 } x_9\text{)},\\
h_{13}\text{:\_below\_p(ARG0 } e_{22}\text{, ARG1 } x_9\text{, ARG2 } x_{23}\text{)},\\
h_{24}\text{:number\_q(ARG0 } x_{23}\text{, RSTR } h_{25}\text{, BODY } h_{26}\text{)},\\
h_{27}\text{:card(ARG0 } x_{23}\text{, ARG1 } i_{28}\text{, CARG } \textit{2015}\text{)}
\end{array} \;\Big|\;
\{\, h_{25} =_q h_{27},\; h_{19} =_q h_{21},\; h_{12} =_q h_{13},\; h_7 =_q h_4,\; h_1 =_q h_2 \,\}\, \rangle
$$

Clause Union

Aladdin (1992 Disney film)

Three wishes

- to be a prince
- to be saved from drowning underwater
- to free the Genie



Fun with linguistics

- Coordination
  to be a prince and to be saved
- Subordination
  to be a prince who is saved
- Presupposition (lecture 11)
  to see my mother – Queen Elizabeth

Subordination

(1) a. David complained that Chris smoked.

b. David wondered who smoked.

c. David couldn't believe how big the house was.

Discussion

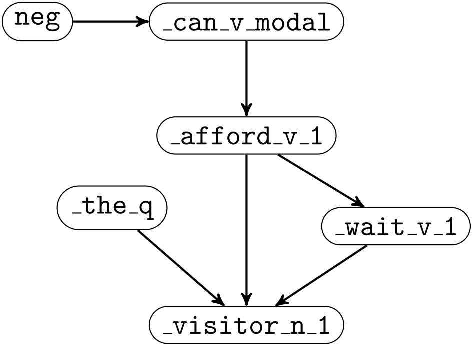
The visitor can't afford to wait.

- who afford?
- who wait?
- who can't?

Discussion

The visitor can't afford to wait.

- who afford?
- who wait? ▷*afford* and *wait* share an argument
- who can't?



Raising and control

Raising

[[Kim to be happy] seems]

$\Downarrow$

[Kim [seems to be happy]]

Control

[Sandy wants [Sandy to go]]

$\Downarrow$

[Sandy wants [PRO to go]]

- Embedded clause is missing its subject
- Subject or object (or PP-obj) of matrix clause (controller) is interpreted as subject of embedded clause.

Small clause

A small clause is a frequently occurring construction that has the semantic subject–predicate characteristics of a clause, but that lacks the tense of a finite clause and appears to lack the status of a constituent.

(2) a. Jim called me a liar.
  b. They named him Pedro.
  c. Fred wiped the table clean.
  d. Larry pounded the nail flat.
  e. Tracy proved the theorem false.
  f. Bo considered Lou a friend.
  g. We saw Fred leave.
  h. Did you hear them arrive?
  i. Dana preferred for Pat to get the job.
  j. Leslie wanted Chris to go.
  k. Lee believed Dominique to have made a mistake.

Adverbial clause

Open

(3) Stretching his arms, David yawned.

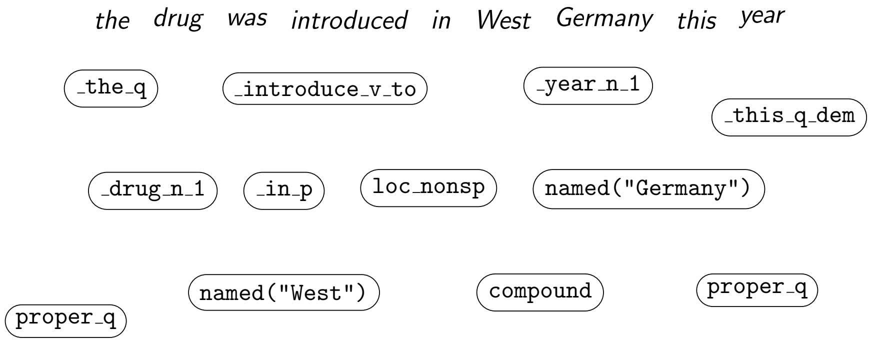
Close

Functor, Argument and Bilinearity

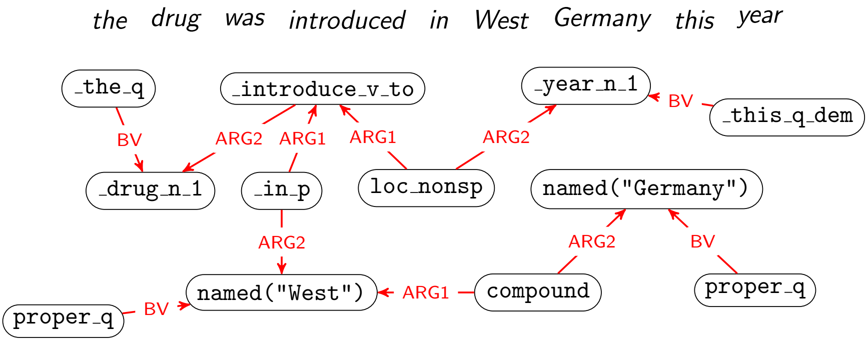
String-to-graph parsing

the drug was introduced in West Germany this year

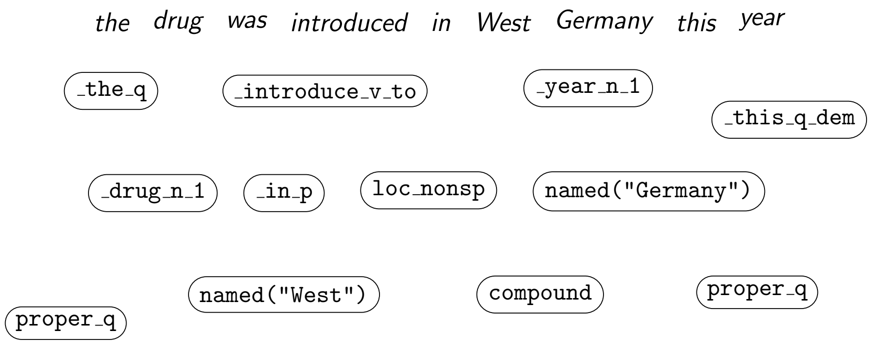
String-to-graph parsing



String-to-graph parsing

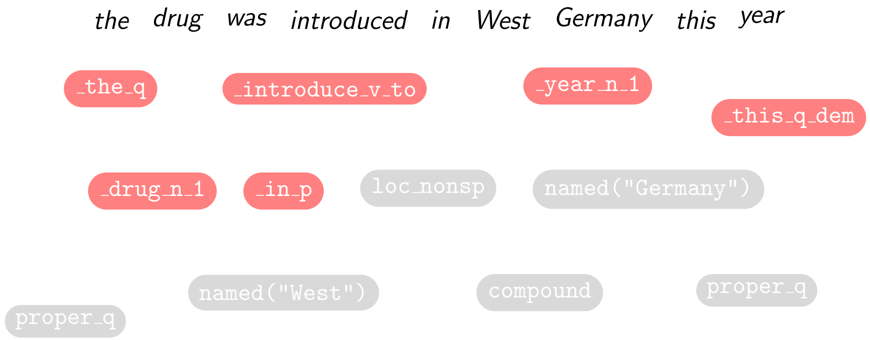


String-to-graph parsing



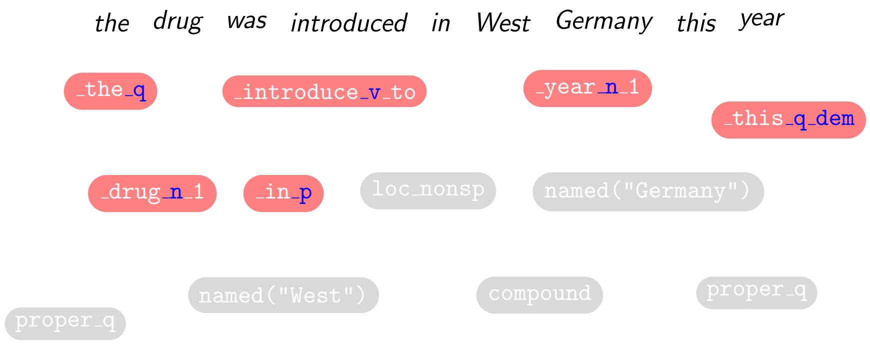
Task 1: Concept Identification

String-to-graph parsing



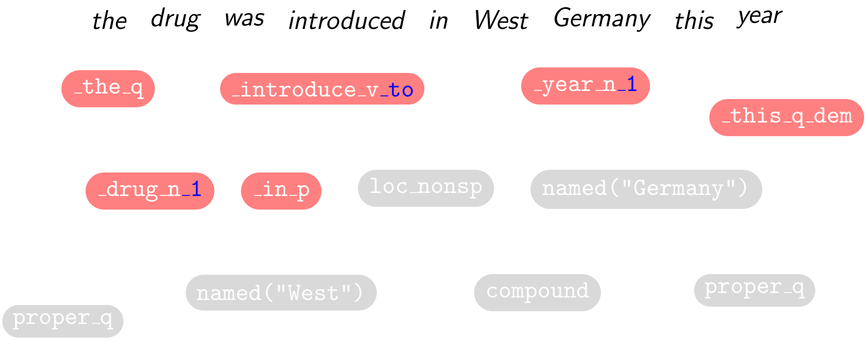
Task 1: Concept Identification

String-to-graph parsing



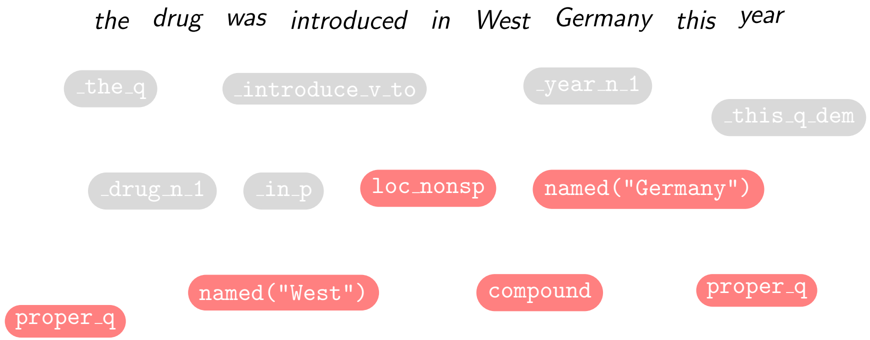
Task 1: Concept Identification

String-to-graph parsing



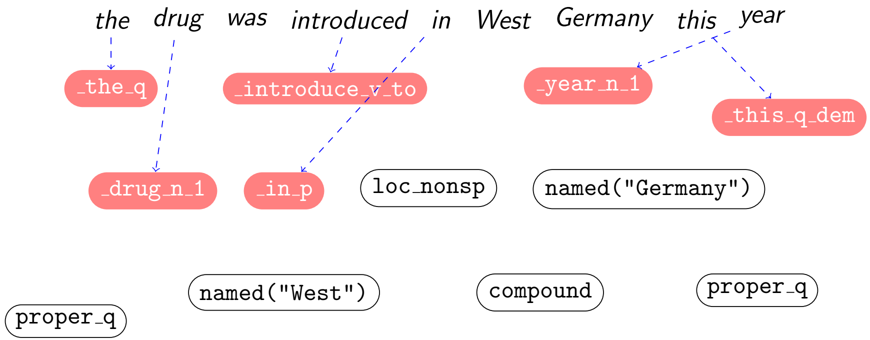
Task 1: Concept Identification

String-to-graph parsing



Task 1: Concept Identification

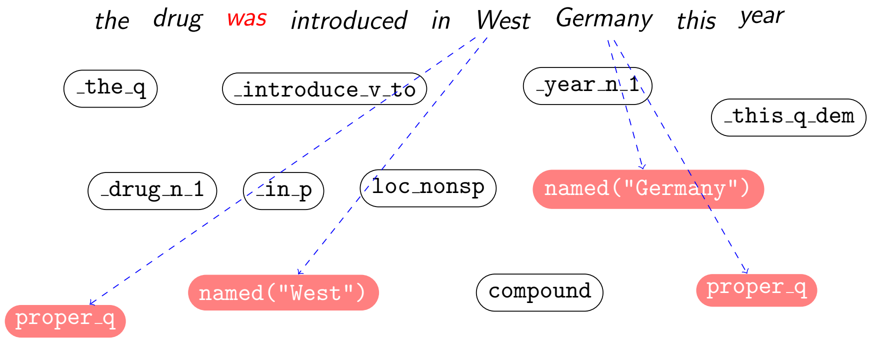
String-to-graph parsing



Task 0: Concept-to-word Alignment

Task 1: Concept Identification

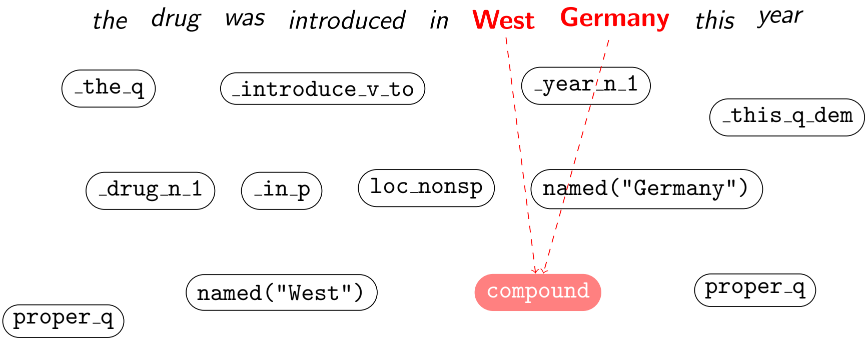
String-to-graph parsing



Task 0: Concept-to-word Alignment

Task 1: Concept Identification

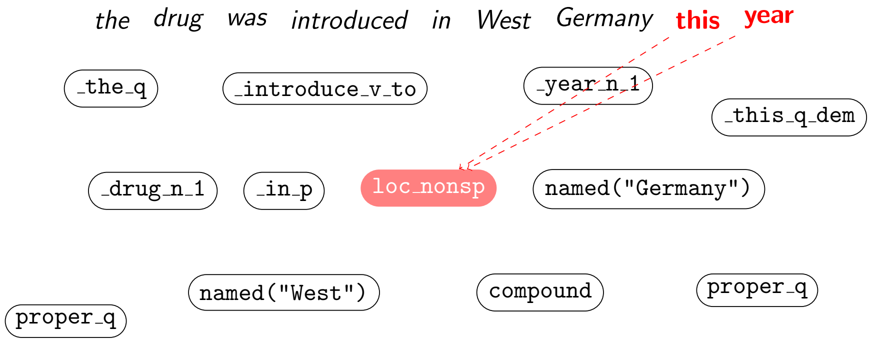
String-to-graph parsing



Task 0: Concept-to-word Alignment

Task 1: Concept Identification

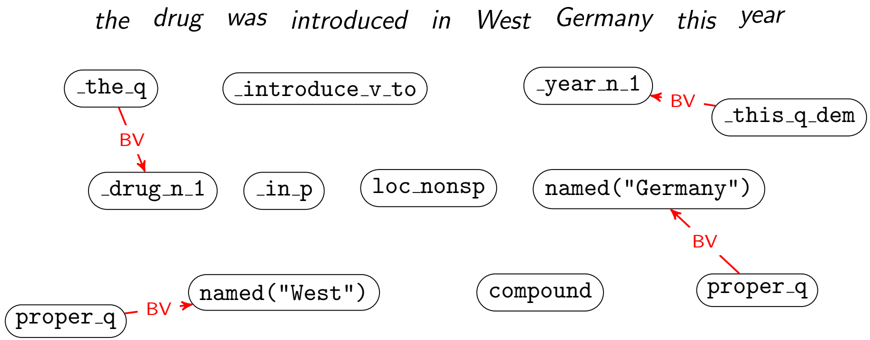
String-to-graph parsing



Task 0: Concept-to-word Alignment

Task 1: Concept Identification

String-to-graph parsing

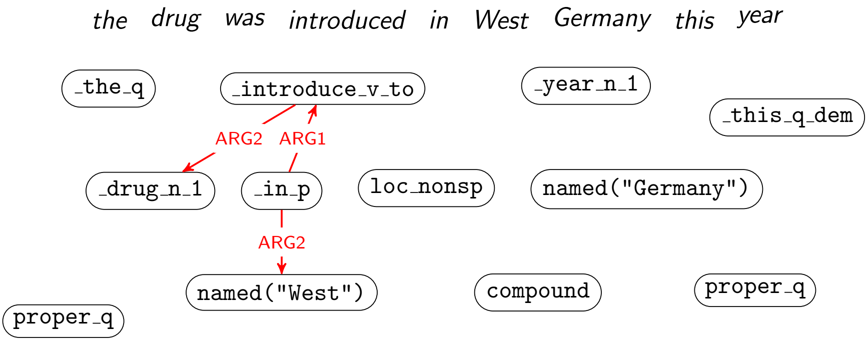


Task 0: Concept-to-word Alignment

Task 1: Concept Identification

Task 2: Relation Detection

String-to-graph parsing

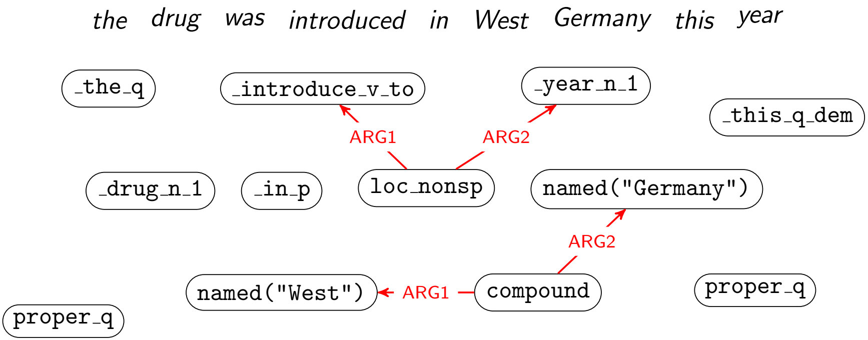


Task 0: Concept-to-word Alignment

Task 1: Concept Identification

Task 2: Relation Detection

String-to-graph parsing

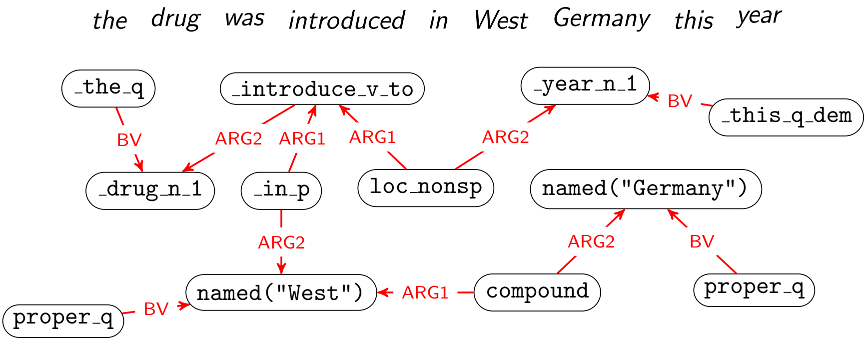


Task 0: Concept-to-word Alignment

Task 1: Concept Identification

Task 2: Relation Detection

String-to-graph parsing



Task 0: Concept-to-word Alignment

Task 1: Concept Identification

Task 2: Relation Detection

Relation detection

Functor–argument relation

Did you hear them arrive?

- arrive: functor?
- arrive: argument?

Relation detection

`word2vec`: define $p(w_{t+j}|w_t)$ as

$$p(o|c) = \frac{\exp(u_o^\top v_c)}{\sum_{w=1}^{|V|} \exp(u_w^\top v_c)}$$

Biaffine parsing

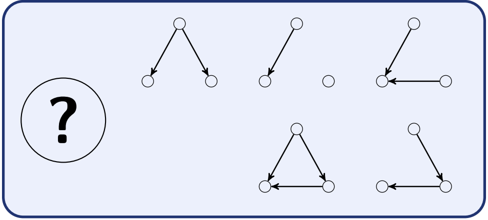
- dot product $\Rightarrow$ inner product

  inner product: a positive-definite symmetric bilinear function

  bilinear function:

  - $f(\alpha_1 + \alpha_2, \beta) = f(\alpha_1, \beta) + f(\alpha_2, \beta), \quad f(k\alpha, \beta) = kf(\alpha, \beta)$
  - $f(\alpha, \beta_1 + \beta_2) = f(\alpha, \beta_1) + f(\alpha, \beta_2), \quad f(\alpha, k\beta) = kf(\alpha, \beta)$
  - If $\{e_1, e_2, ...e_n\}$ is a basis, then $f(e_i, e_j)$ ($\forall i, j : 1 \leq i, j \leq n$) identifies f.
- bilinear $\Rightarrow$ biaffine: adding a prior
- u_i/v_i $\Rightarrow$ as functor/argument
- $+j$ (fixed window) $\Rightarrow$ the whole sentence

Representational: directly evaluate the target structure



Maximum Subgraph Parsing

Start from a directed graph $G = (V, E)$ and a score function that evaluates the *goodness* of a graph.

Search for a subgraph $G' = (V, E' \subseteq E)$ that maximizes:

$$G' = \arg\max_{G^* = (V, E^* \subseteq E)} \textsc{Score}(G^*)$$

First-order factorization

$$G' = \arg\max_{G^* = (V, E^* \subseteq E)} \sum_{e \in E^*} \textsc{ScorePart}(e)$$

Reading and exercise

- T. Dozat and C. Manning. Deep Biaffine Attention for Neural Dependency Parsing.
- S. Oepen, A. Koller and W. Sun. ACL Tutorial on Graph-Based Meaning Representations: Design and Processing.
- Pre-lecture 9 exercise: annotatng bi-lexical semantic graphs for the following sentences:
  - His words came after Ukraine's president urged calm, saying the biggest enemy was panic.
  - Moscow, with more than 100,000 troops near the border, has denied it plans to invade.

References I

Baker, C. F., Fillmore, C. J., & Lowe, J. B. (1998). The berkeley framenet project. In *Proceedings of the 17th international conference on computational linguistics-volume 1* (pp. 86–90).

Banarescu, L., Bonial, C., Cai, S., Georgescu, M., Griffitt, K., Hermjakob, U., . . . Schneider, N. (2013). Abstract meaning representation for sembanking. In *Proceedings of the 7th linguistic annotation workshop and interoperability with discourse* (pp. 178–186).

Basile, V., Bos, J., Evang, K., & Venhuizen, N. (2012). Developing a large semantically annotated corpus. In *Eighth international conference on language resources and evaluation* (pp. 3196–3200).

Flickinger, D. (2000). On building a more effcient grammar by exploiting types. *Natural Language Engineering*, *6*(1), 15–28.

Flickinger, D., Oepen, S., & Bender, E. M. (2017). Sustainable development and refinement of complex linguistic annotations at scale. In *Handbook of linguistic annotation* (pp. 353–377). Springer.

References II

Flickinger, D., Zhang, Y., & Kordoni, V. (2012). Deepbank: A dynamically annotated treebank of the wall street journal. In *Proceedings of the eleventh international workshop on treebanks and linguistic theories* (p. 85-96).

He, L., Lewis, M., & Zettlemoyer, L. S. (2015). Question-answer driven semantic role labeling: Using natural language to annotate natural language. In *Emnlp.*

Kingsbury, P., & Palmer, M. (2002). From treebank to propbank. In *Lrec* (pp. 1989–1993).

Oepen, S., Flickinger, D., Toutanova, K., & Manning, C. D. (2004). Lingo redwoods. *Research on Language and Computation*, *2*(4), 575–596.

Oepen, S., & Lønning, J. T. (2006). Discriminant-based mrs banking. In *Lrec* (pp. 1250–1255).