The continued potential of rule-based semantic parsing in the era of deep learning

The Universal Natural Language Understanding project

Jamie Y. Findlay

Syntax-Semantics Oberseminar,
Goethe Universität Frankfurt am Main
12 June 2023

Background

- **Semantic parsing:**
  text → symbolic meaning representation
- **State of the art:** train a neural network
  - Gets good results!
  - F1 scores in the high 80s
- **However:**
  - task not as easy as assumed
  - problems with robustness/hallucination
  - 'black boxes' not theoretically satisfying
- **So**: we are building a rule-based system.

Precision

$$\frac{\text{true positives}}{\text{true positives} + \text{false positives}}$$

e.g. prec. of 2/3 means 2 out of 3 +ves are true +ves

Recall/Sensitivity

$$\frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$

e.g. recall of 2/3 means 2 out of 3 relevant items found

F1 score

$$2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

i.e. the harmonic mean of precision and recall

Outline

1. Background

2. The Universal Natural Language Understanding project

3. Universal Dependencies and semantic interpretation

4. Glue Semantics for UD

5. (Universal) rules for semantic interpretation

6. Comparison with the state of the art

The Universal Natural Language Understanding project

Goals

- *Universal Natural Language Understanding* is an (ambitiously named) RCN-funded project on computational semantics.
    - Dag Haug (PI)
    - Jamie Findlay (postdoc)
    - Ahmet Yıldırım (senior engineer)
    - Saeedeh Salimifar (PhD fellow)
- It aims to build a system which can
    1. create rich, logic-based representations (specifically, DRSs); and
    2. do this for as many languages as possible.

Semantic parsing

- The state of the art involves pure machine learning.

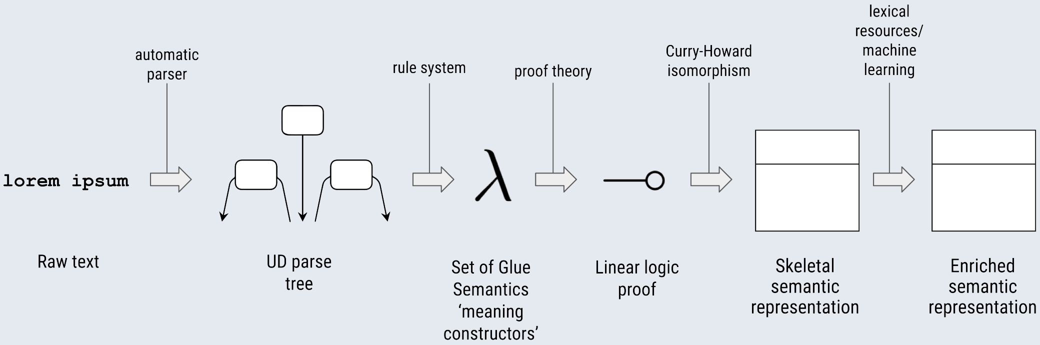
  (van Noord et al. 2018; van Noord 2019; Evang 2019)

- Our approach combines machine learning with a rule-based core.

  (Findlay et al. 2023)

  1. (shallow) syntactic parse (UD) (ML)
  2. symbolic meaning representation (DRS) (rule-based)
  3. further (e.g. lexical) enrichment (ML/rule-based)

Our pipeline



lorem ipsum

Raw text

automatic parser

UD parse tree

rule system

Set of Glue Semantics ‘meaning constructors’

proof theory

Linear logic proof

Curry-Howard isomorphism

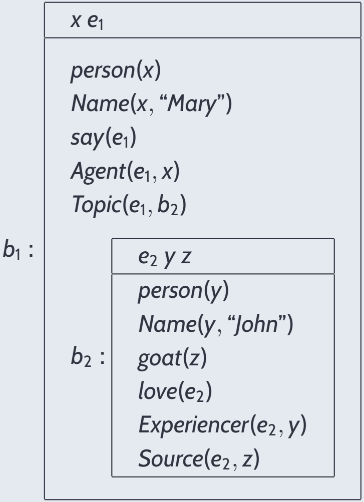
Skeletal semantic representation

lexical resources/ machine learning

Enriched semantic representation

Semantic representations

Mary says John likes goats.



$b_1 :$

$x\ e_1$

$person(x)$

$Name(x, \text{“}Mary\text{”})$

$say(e_1)$

$Agent(e_1, x)$

$Topic(e_1, b_2)$

$b_2 :$

$e_2\ y\ z$

$person(y)$

$Name(y, \text{“}John\text{”})$

$goat(z)$

$love(e_2)$

$Experiencer(e_2, y)$

$Source(e_2, z)$

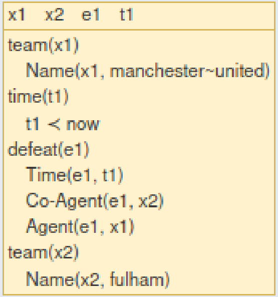
Discourse Representation Structures (DRSs)

- discourse referents + conditions on discourse referents
- DRSs can be embedded inside other DRSs (also under negation, disjunction, modal operators, …)
- Translatable to first-order logic for use with theorem provers.

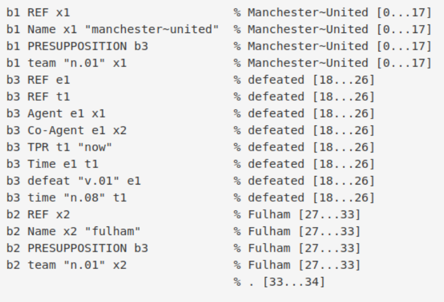
The Parallel Meaning Bank

- One good reason for choosing DRS representations is the **Parallel Meaning Bank** (PMB), which contains DRS representations for sentences in English, German, Dutch and Italian. (Abzianidze et al. 2017)

Manchester United defeated Fulham.



```
x1  x2  e1  t1
team(x1)
   Name(x1, manchester~united)
time(t1)
   t1 ≺ now
defeat(e1)
   Time(e1, t1)
   Co-Agent(e1, x2)
   Agent(e1, x1)
team(x2)
   Name(x2, fulham)
```

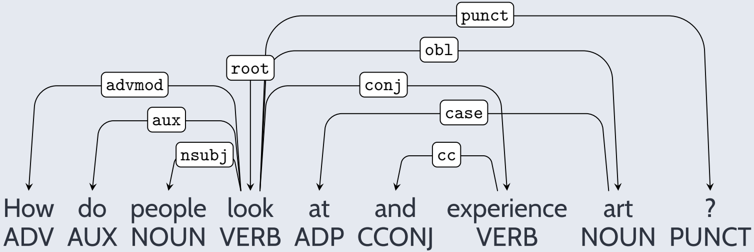


```
b1 REF x1                        % Manchester~United [0...17]
b1 Name x1 "manchester~united"   % Manchester~United [0...17]
b1 PRESUPPOSITION b3             % Manchester~United [0...17]
b1 team "n.01" x1                % Manchester~United [0...17]
b3 REF e1                        % defeated [18...26]
b3 REF t1                        % defeated [18...26]
b3 Agent e1 x1                   % defeated [18...26]
b3 Co-Agent e1 x2                % defeated [18...26]
b3 TPR t1 "now"                  % defeated [18...26]
b3 Time e1 t1                    % defeated [18...26]
b3 defeat "v.01" e1              % defeated [18...26]
b3 time "n.08" t1                % defeated [18...26]
b2 REF x2                        % Fulham [27...33]
b2 Name x2 "fulham"              % Fulham [27...33]
b2 PRESUPPOSITION b3             % Fulham [27...33]
b2 team "n.01" x2                % Fulham [27...33]
                                 % . [33...34]
```

Universal Dependencies and semantic interpretation

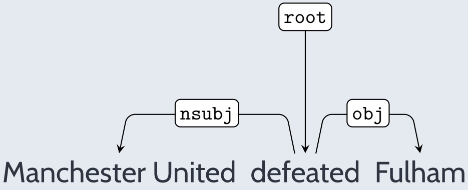
Universal Dependencies

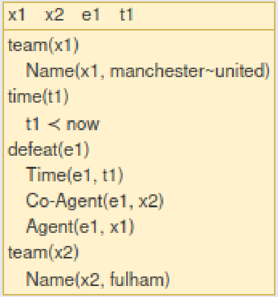
- We use a Universal Dependencies (UD) parse as our starting point. (Nivre et al. 2020)



- This is a shallow syntactic representation, which is hard to convert into a meaningful semantics.
    - *Enhanced* Universal Dependencies add more semantically-relevant information, but have their limitations. (Findlay & Haug 2021)
- On the plus side, structures like this are available for 100+ languages.

From UD tree to DRS





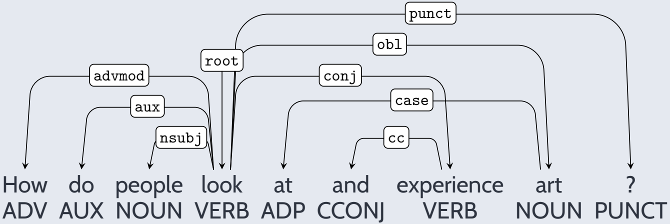
```
x1   x2   e1   t1
team(x1)
   Name(x1, manchester~united)
time(t1)
   t1 < now
defeat(e1)
   Time(e1, t1)
   Co-Agent(e1, x2)
   Agent(e1, x1)
team(x2)
   Name(x2, fulham)
```

- It looks like we only need to map tokens 1, 2, 3 to discourse referents x_1, e_1/t_1 and x_2, then do some labelling.
- There is some truth to this and it has inspired frameworks like UDepLambda.

(Reddy et al. 2017)

- But the question is whether we can do more as the syntactic and semantic representations grow in complexity!

What UD offers



1. Consistent representations of most predicate-argument structures

2. Separation of function words and content words

*e*1 *e*2 **t*1 **t*2 *x*1 *x*2 *x*3
(*x*3 = '?')
Manner(*e*1, *x*3)
Manner(*e*2, *x*3)
manner.*n*.01(*x*3)
Time(*e*1, *t*1)
Time(*e*2, *t*2)
person.*n*.01(*x*1)
Agent(*e*1, *x*1)
Theme(*e*1, *x*2)
look_at.*v*.02(*e*1)
Experiencer(*e*2, *x*1)
Stimulus(*e*2, *x*2)
experience.*v*.02(*e*2)
art.*n*.01(*x*2)
(*t*2 = '*now*')
**time*.*n*.08(*t*2)
(*t*1 = '*now*')
**time*.*n*.08(*t*1)

What UD lacks

1. Although we have the predicate-argument structures, the function words and the lexical items, we lack the corresponding meanings:
   - `nsubj` $\mapsto$ *Agent* or *Experiencer* or something else?
   - `aux` – encodes tense, modality or something else?
   - sense disambiguation of lexical items?

2. The shallow UD structure leaves some meaningful relations unexpressed or under-expressed, and raises questions about compositionality.

Problems for compositionality

- Montague-style compositionality poses two problems for a depedency grammar-style syntax:
  - the homomorphism problem
  - the lexical integrity problem (Haug & Findlay 2023)

Homomorphism Syntax and lexicon must jointly *determine* meaning (cf. QR), and syntax must be binary branching – not the case in a dependency grammar.

Lexical integrity Words are atoms, but sometimes we want them to contribute more than one meaning that can interact independently with other meanings.

Problems for compositionality

- Even if we can translate UD syntactic nodes into lambda terms, it is not clear that the UD syntax is specific enough to guide the composition of those terms!
- So we take the lead from another framework with similar issues: **LFG + Glue Semantics**

Glue Semantics for UD

What is Glue?

- A theory of the syntax-semantics interface, originally developed for **Lexical Functional Grammar (LFG)**, and now the mainstream in LFG.

  (Dalrymple et al. 1993; Dalrymple 1999; Asudeh 2022)
- Has been applied to other frameworks:
  - HPSG (Asudeh & Crouch 2002)
  - LTAG (Frank & van Genabith 2001)
  - Minimalism (Gotham 2018)
- Meanings are paired with a *composition logic* (a fragment of linear logic) to form a **meaning constructor**; semantic composition is deduction in the logic, mediated by the Curry-Horward correspondence. (Curry & Feys 1958; Howard 1980)

A crude characterisation would be that Glue Semantics is like categorial grammar and its semantics, but without the categorial grammar. (Crouch & van Genabith 2000)

The idea behind Glue semantics

The syntax is not (and should not be!) specific enough to guide composition of lambda terms, so we introduce a specific combinatory logic for lambda terms.

$$\dfrac{\dfrac{\lambda x.\lambda y.\mathbf{love}(x, y) : A \multimap B \multimap C \qquad \mathbf{naomi} : A}{\lambda y.\mathbf{love}(\mathbf{naomi}, y) : B \multimap C} \qquad \mathbf{jim} : B}{\mathbf{love}(\mathbf{naomi}, \mathbf{jim}) : C}$$

- This looks a lot like categorial grammar!
- But decoupled from surface syntax, which is good for universality
- By making *love* type $A \multimap B \multimap C$, did we just stipulate a particular composition order? No!

Inference rules

Function application : implication elimination

$$\frac{\mathbf{f} : A \multimap B \qquad \mathbf{a} : A}{\mathbf{f}(\mathbf{a}) : B} \multimap_{\mathcal{E}}$$

Lambda abstraction : implication introduction

$$\frac{\begin{array}{c}[\mathbf{a} : A]^1 \\ \vdots \\ \mathbf{f} : B\end{array}}{\lambda x.\mathbf{f} : A \multimap B} \multimap_{\mathcal{I},1}$$

Switching argument orders

$(A \multimap B \multimap C) \multimap (B \multimap A \multimap C)$ is a theorem of linear logic.

$$\dfrac{\dfrac{\dfrac{\dfrac{\lambda x.\lambda y.\mathbf{love}(x,y) : A \multimap B \multimap C \qquad [\mathbf{a} : A]^{1}}{\lambda y.\mathbf{love}(\mathbf{a},y) : B \multimap C} \qquad [\mathbf{b} : B]^{2}}{\mathbf{love}(\mathbf{a},\mathbf{b}) : C}}{\lambda x.\mathbf{love}(x,\mathbf{b}) : A \multimap C}\,{\multimap_{\mathcal{I},1}}}{\lambda y.\lambda x.\mathbf{love}(x,y) : B \multimap A \multimap C}\,{\multimap_{\mathcal{I},2}}$$

- There is also other stuff we get for free, like type raising operators: $(A \multimap ((A \multimap B) \multimap B))$ is also a theorem.

- Moreover, there is an efficient proof algorithm (based on CYK) and a good implementation (the Glue Semantics Workbench).

(Hepple 1996; Lev 2007; Messmer & Zymla 2018)

Quantifier scope ambiguities

(1) *Someone loves everyone.*

$\exists > \forall$ – someone has a lot of love to give

$\forall > \exists$ – everyone is loved

- Faced with the homomorphism problem, transformational theories treat this ambiguity as syntactic.
- Glue can treat it as genuinely semantic: different scopings correspond to different proofs from the same premises.

Quantifier scope ambiguity: surface scope

$$\dfrac{\dfrac{\dfrac{\boxed{\begin{array}{l}\textit{loves}\\ \lambda x.\lambda y.\mathbf{love}(x,y):\\ A\multimap B\multimap C\end{array}} \qquad [\mathbf{a}:A]^1}{\begin{array}{l}\lambda y.\mathbf{love}(\mathbf{a},y):\\ B\multimap C\end{array}} \qquad \boxed{\begin{array}{l}\textit{everyone}\\ \lambda P.\forall z.\mathbf{person}(z)\rightarrow P(z):\\ (B\multimap C)\multimap C\end{array}}}{\dfrac{\begin{array}{l}\forall z.\mathbf{person}(z)\rightarrow\mathbf{love}(\mathbf{a},z):\\ C\end{array}}{\begin{array}{l}\lambda x.\forall z.\mathbf{person}(z)\rightarrow\mathbf{love}(x,z):\\ A\multimap C\end{array}}\multimap_{\mathcal{I},1}} \qquad \boxed{\begin{array}{l}\textit{someone}\\ \lambda P.\exists x.\mathbf{person}(x)\wedge P(x):\\ (A\multimap C)\multimap C\end{array}}}{\begin{array}{l}\exists x.\mathbf{person}(x)\wedge(\forall z.\mathbf{person}(z)\rightarrow\mathbf{love}(z,x)):\\ C\end{array}}$$

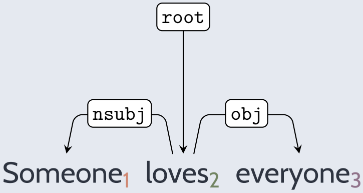
Quantifier scope ambiguity: inverse scope

$$
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\boxed{\begin{array}{l}\textit{loves}\\ \lambda x.\lambda y.\mathbf{love}(x,y):\\ A\multimap B\multimap C\end{array}}
\quad [\mathbf{a}:A]^1
}{\begin{array}{l}\lambda y.\mathbf{love}(\mathbf{a},y):\\ B\multimap C\end{array}}
\quad [\mathbf{b}:B]^2
}{\begin{array}{l}\mathbf{love}(\mathbf{a},\mathbf{b}):\\ C\end{array}}
}{\begin{array}{l}\lambda x.\mathbf{love}(x,\mathbf{b}):\\ A\multimap C\end{array}}\multimap_{\mathcal{I},1}
\quad
\boxed{\begin{array}{l}\textit{someone}\\ \lambda P.\exists x.\mathbf{person}(x)\wedge P(x):\\ (A\multimap C)\multimap C\end{array}}
}{\begin{array}{l}\exists x.\mathit{person}(x)\wedge \mathit{love}(x,\mathbf{b}):\\ C\end{array}}
}{\begin{array}{l}\lambda y.\exists x.\mathit{person}(x)\wedge \mathit{love}(x,y):\\ B\multimap C\end{array}}\multimap_{\mathcal{I},2}
\quad
\boxed{\begin{array}{l}\textit{everyone}\\ \lambda P.\forall z.\mathbf{person}(z)\rightarrow P(z):\\ (B\multimap C)\multimap C\end{array}}
}{\begin{array}{l}\forall z.\mathbf{person}(z)\rightarrow(\exists x.\mathbf{person}(x)\wedge \mathbf{love}(x,z)):\\ C\end{array}}
$$

Connecting up with the syntax

- The lexical entry of *love* cannot really be based on atomic categories like *A*, *B* and *C*.
- Instead, we use a first-order system where predicates are **type constructors** that apply to nodes as terms to yield **types**
- Writing $\hat{*}$ for the current node,
    - $E(\hat{*})$ is a type *e* meaning for that node
    - $T(\hat{*})$ is a type *t* meaning for that node
    - $E(\hat{*}) \multimap T(\hat{*})$ is a function type between the two (e.g. the type of a bare noun)
- Morever, we can use **path descriptions** based on syntactic labels:
    - $E(\hat{*}\ \texttt{nsubj})$ is a type *e* meaning for the current node's (nominal) subject
    - $E(\hat{*}\ \texttt{obj})$ is a type *e* meaning for the current node's object
    - $E(\hat{*}\ \texttt{subj}) \multimap E(\hat{*}\ \texttt{obj}) \multimap T(\hat{*})$ is the type we need for *love*

Another example



- Using ‘$\uparrow$’ to refer to the current node’s mother, we assign the following types:

someone	$[E(\hat{*}) \multimap T(\uparrow)] \multimap T(\uparrow)$	i.e. $[E(1) \multimap T(2)] \multimap T(2)$
love	$E(\hat{*}\ \texttt{subj}) \multimap E(\hat{*}\ \texttt{obj}) \multimap T(\hat{*})$	i.e. $E(1) \multimap E(3) \multimap T(2)$
everyone	$[E(\hat{*}) \multimap T(\uparrow)] \multimap T(\uparrow)$	i.e. $[E(3) \multimap T(2)] \multimap T(2)$

- This is isomorphic to the atomic types we used: $E(1) \mapsto A$, $E(3) \mapsto B$, $T(2) \mapsto C$, and so we get the same proofs

Glue summary

- Glue allows us to build logical representations off dependency trees without requiring arbitrary binarization or lexical decomposition.
- The syntax can underspecify the semantics – this also makes it easier to “offload” work to the interface.
  - E.g. restrictions on gaps in relative clauses. (Haug & Findlay 2023: 28–30)

(Universal) rules for semantic interpretation

Rules for semantic interpretation

- Rules have two parts:
  `Condition(s) -> Meaning constructor`
- UD tree inspected node by node, via custom Haskell script.
- Meanings expressed in format of Python NLTK package. (Garrette & Klein 2009)
- Sample ruleset available at `https://github.com/Universal-NLU/UNLU`.

Based on part of speech

```
coarsePos = PROPN -> \X.(([], [Name(X, ‘:LEMMA:‘)])) : e(!) -o t(!)
```

$$\lambda X.\begin{array}{|c|}\hline \quad \\ \hline Name(X, \text{“ : }LEMMA\text{ : ”}) \\ \hline\end{array} : E(\hat{\ast}) \multimap T(\hat{\ast})$$

Based on UD relation

```
relation = nsubj; ^ {coarsePos = VERB} ->
\Q.\V.\F.(Q(\X.(V(\E.(([], [nsubj(E,X)]) + F(E)))))) :
((e(!) -o t(^)) -o t(^)) -o (x(^) -o x(^))
```

$$\lambda Q \lambda V \lambda F.Q(\lambda X.(V(\lambda E.\begin{array}{|c|}\hline \\ \hline \mathit{nsubj}(E,X) \\ \hline \end{array} + F(E)))) : [[(E(\hat{*}) \multimap T(\uparrow)] \multimap T(\uparrow)] \multimap X(\uparrow) \multimap X(\uparrow)$$

$(x \equiv \langle\langle v, t\rangle, t\rangle)$

Based on features

```
coarsePos = VERB; ~ aux.*; Tense = Pres ->
\V.\F.(V(\E.(([T], [time(T), EQ(T, ‘now‘), Time(E, T)]) + F(E)))) :
x(!) -o x(!)
```

$$\lambda V \lambda F.V(\lambda E.\begin{array}{|l|}\hline T \\ \hline time(T) \\ T = \text{‘}now\text{’} \\ Time(E,T) \\ \hline \end{array} + F(E)) : X(\hat{*}) \multimap X(\hat{*})$$

Dealing with structure-meaning interactions

Some meanings affect the structure of the DRS:

(2) *A politician lies.*

$$\begin{array}{|l|}\hline x\ e \\ \hline \mathit{politician}(x) \\ \mathit{lie}(e, x) \\ \hline \end{array}$$

(3) *Every politician lies.*

$$\begin{array}{|c|}\hline \\ \hline \begin{array}{|l|}\hline x \\ \hline \mathit{politician}(x) \\ \hline \end{array} \Rightarrow \begin{array}{|l|}\hline e \\ \hline \mathit{lie}(e, x) \\ \hline \end{array} \\ \hline \end{array}$$

- We therefore allow some lemma-based rules for ‘logic words’ like quantifiers, negation, etc.
- Ideally these would be identifiable from features alone, but this is not the case.

Lemma-specific rules

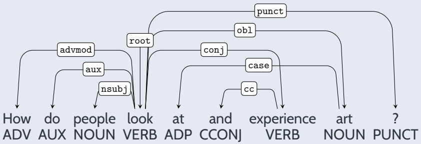
```
relation = det; lemma = "every" ->
\P.\Q.([ ],[ ( ( ([X],[ ]) + P(X) ) => (Q(X)) ) ]) :
(e(^) -o t(^)) -o (e(^) -o t(^ ^)) -o t(^ ^)
```

$$\lambda P.\lambda Q.\begin{array}{|c|}\hline \\ \hline \left(\begin{array}{|c|}\hline X \\ \hline \\ \hline\end{array} + P(X)\right) \Rightarrow Q(X) \\ \hline\end{array} : [E(\uparrow) \multimap T(\uparrow)] \multimap [E(\uparrow) \multimap T(\uparrow\uparrow)] \multimap T(\uparrow\uparrow)$$

Language parametrisation

```
"eng": {
    "stanza_lang_code":"en",
    "typological_features": {
        "negative_concord" : "no",
        "sequence_of_tense": "yes",
        "grammatical_gender": "no"
      },
    "lexical_items": {
        "future_aux": "will",
        "definite_det": "(the|this|that)",
        "indefinite_det": "(a|some)",
        "universal_quantifier": "(every|each)",
        "infinitive_marker": "to",
        "conjunction": "and",
        "disjunction": "or"
      },
  }
```

How far do we get?



$e_1\ e_2\ x_1\ x_2\ x_3\ {}^*t_1$
$person(x_1)$
$art(x_2)$
$look(e_1)$
$experience(e_2)$
$at(e_1, x_2)$
$(x_3 = '?')$
$how(e_1, x_3)$
$nsubj(e_1, x_1)$
$Time(e_1, t_1)$
${}^*time(t_1)$
$t_1 = 'now'$

Missing information:

- “look at” as MWE
- purely syntactic roles
- no roles for second conjunct

$e1\ e2\ {}^*t1\ {}^*t2\ x1\ x2\ x3$
$(x3 = '?')$
$Manner(e1, x3)$
$Manner(e2, x3)$
$manner.n.01(x3)$
$Time(e1, t1)$
$Time(e2, t2)$
$person.n.01(x1)$
$Agent(e1, x1)$
$Theme(e1, x2)$
$look\_at.v.02(e1)$
$Experiencer(e2, x1)$
$Stimulus(e1, x2)$
$experience.v.02(e2)$
$art.n.01(x2)$
$(t2 = 'now')$
${}^*time.n.08(t2)$
$(t1 = 'now')$
${}^*time.n.08(t1)$

Comparison with the state of the art

At first sight, hopeless!

	PMB 2.2.0		PMB 3.0.0		PMB 4.0.0		
	dev	test	dev	test	dev	test	eval
van Noord et al. (2020)	86.1	88.3	88.4	89.3	–	–	–
Liu et al. (2021)	–	88.7	–	–	–	–	–
Yıldırım & Haug (2023)	**87.5**	**89.2**	**89.8**	**90.3**	**88.1**	**89.0**	**86.9**

Table 1: Recently reported F1 scores for PMB 2.2.0, 3.0.0, and 4.0.0 datasets

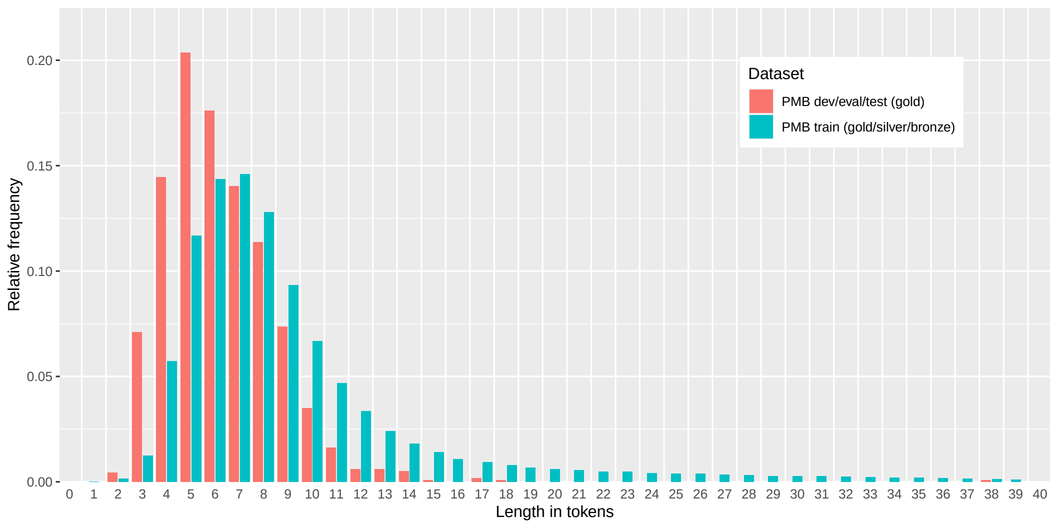
Language	Raw	Structural
German (de)	30.8	59.6
English (en)	46.7	63.4
Italian (it)	30.7	58.9
Dutch (nl)	28.6	58.4
AVERAGE	34.2	60.1

Table 2: F1 scores from the rule-based system on PMB 4.0.0 test set

But DRS parsing isn't as easy as it seems …

1. PMB test set only has short sentences.
2. Neural network parsers not as robust/reliable as rule-based systems.
3. Neural network parsers might overfit to data.

Sentence length in the PMB



The DRASTIC corpus

- To provide more representative test data, we annotated ∼3000 words of naturally-occuring texts, taken from the GUM corpus.

  (Zeldes 2017)

- We call this the DRASTIC corpus (‘Discourse Representation Annotations with Sentence Texts of Increased Complexity’).[1]

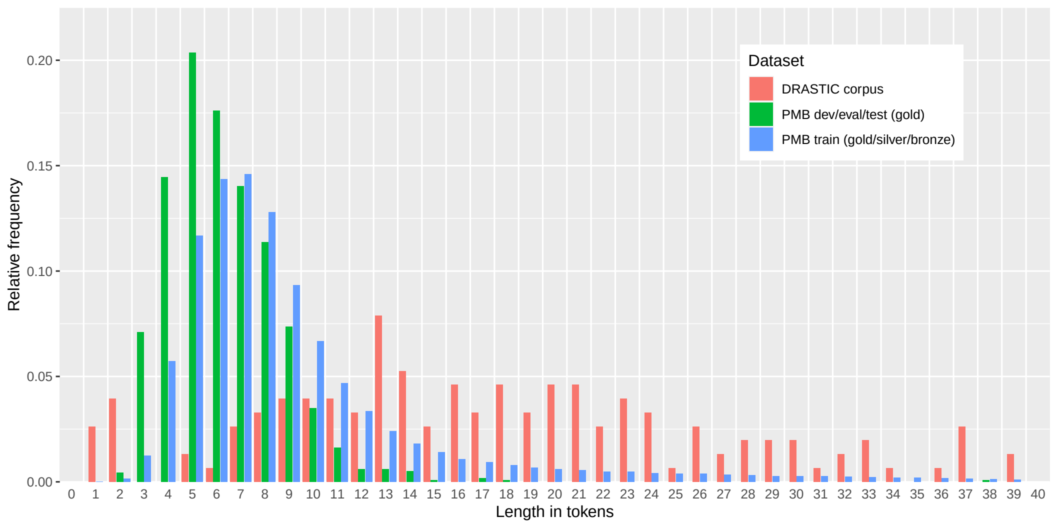
  (Haug et al. 2023)

(Sub-)corpus	Median	Mean	St.dev.
`dvorak`	23	23.9	9.68
`marbles`	17	19.6	12.4
`nida`	18	19.1	11.1
`short-texts`	13	12.8	4.29
DRASTIC (all)	17	18.5	10.6
PMB (all)	8	10.0	9.53
PMB (test only)	6	6.60	2.08

Table 3: Sentence length across (sub-)corpora

[1] Available here: `https://github.com/Universal-NLU/DRASTIC`

Sentence length in the PMB and DRASTIC

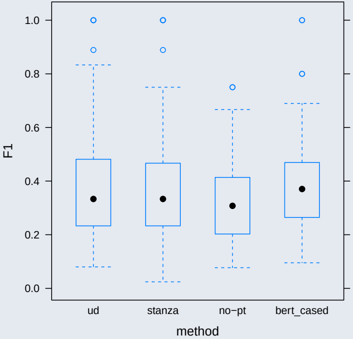


Performance on DRASTIC

- The advantage of neural networks drops off almost entirely!

  (cf. Yao & Koller 2022; Donatelli & Koller 2023)

- The best neural parser from the PMB experiments achieves an F1 score of only **36.2** on the DRASTIC texts.



Robustness – complexity

- Rule-based systems scale better than stochastic models.
- The neural nets often struggle with complex sentences involving negation and other scopal operators.

(4) *While the impact of a translation may be close to the original, there can be no identity in detail.*

Correct: $\Diamond[\ldots] \wedge \neg\Diamond[\ldots]$

ML: $\Diamond\Diamond\neg[\ldots]$

Robustness – coverage

- Lambda calculus, like other rule-based approaches, is brittle.
- If there is a type clash, the result is undefined.
- Here is the non-reduced lambda term for *How do people look at and experience art?*

```
λV.(V(λF.([],[])))(λX1.(λQ.λV.λF.(Q(λX.(V(λE.(([],[at(E,X)]) +
F(E))))))(λX.(λP.λQ.(([X],[]) + P(X) + Q(X)
)(λY.(λX.([],[art(X)])(Y)))(λZ.(X(Z)))))(X1))(λX.(λV.λU.λF.(U(F) +
V(λG.([],[])))(λF.(([E],[experience(E)]) + F(E)
))(X))(λX1.(λQ.λV.λF.(Q(λX.(V(λE.(([],[nsubj(E,X)]) +
F(E))))))(λZ.(λP.λQ.(([X],[]) + P(X) + Q(X) )(λX.(λX.([],[person(X)]
)(X)))(λY.(Z(Y)))))(X1))(λP.λF.P(λE(([X], [(X='?'), how(E,X)]) +
F(E)))(λV.λF.(V(λE.((([],[PRESUPPOSITION(([T],[ time(T), (T='now')])),
Time(E,T)])) + F(E))))(λF.(([E],[ look(E)]) + F(E) )))))))
```

Increasing robustness in our system

- There are still many instances our rules don’t cover.
- We do not want a failure in one place to cause the whole proof to fail.
- But in many cases, even if we don’t know what the meaning of some subconstituent is, we do know *what type* it should be.
  - For example, we consistently treat arguments as event modifiers; and we know that conjuncts should have the type of their heads.
- So we can provide a default “placeholder” semantics of the correct type in case the computation fails.

Example

Dvořák’s own style has been described as "the fullest recreation of a national idiom with that of the symphonic tradition, absorbing folk influences and finding effective ways of using them".

$$\begin{array}{|l|}
\hline
E\ F3\ F6\ F8\ {}^{*}X\ {}^{*}T \\
\hline
{}^{*}Name(X, dvořák) \\
poss(F3, X) \\
own(F6) \\
Attribute(F3, F6) \\
style(F3) \\
describe(E) \\
{}^{*}time(T) \\
TPR(T, now)) \\
Time(E, T) \\
nsubj\text{-}pass(E, F3) \\
obl(E, F8) \\
DUMMY(F8) \\
\hline
\end{array}$$

Reliability

- The rules are much more likely to miss things out than put extraneous things in.
- By contrast, the ML systems, as always, are prone to 'hallucinations'.
  - teenager, what teenager?
  - female YouTube
- The rules will consistently get names right (they just lift the lemmas); the ML output goes very wrong here.
  - E.g. *(Jenna) Marbles* rendered as 'georgia strawberry', 'marau', 'margis', 'name', etc.
  - Evidence that models overfit to pecularities of the PMB training data? (e.g. at least 15% of the sentences across the PMB subsets contain the name *Tom* …)

Summing up

- DRS parsing is not as easy as the PMB test set would lead us to believe.
- Definitely a role for rule-based systems:
    - increased robustness when it comes to length and complexity.
    - more reliable when it comes to systematic correspondences (e.g. names)
    - more theoretically illuminating
- With improved rules performance will also improve.

References i

Abzianidze, Lasha, Johannes Bjerva, Kilian Evang, Hessel Haagsma, Rik van Noord, Pierre Ludmann, Duc-Duy Nguyen & Johan Bos. 2017. The Parallel Meaning Bank: Towards a multilingual corpus of translations annotated with compositional meaning representations. In *Proceedings of the 15th conference of the European chapter of the association for computational linguistics: Volume 2, short papers*, 242–247. Valencia, Spain: Association for Computational Linguistics. `https://www.aclweb.org/anthology/E17-2039`.

Asudeh, Ash. 2022. Glue Semantics. *Annual Review of Linguistics* 8. 321–341. `https://doi.org/10.1146/annurev-linguistics-032521-053835`.

Asudeh, Ash & Richard Crouch. 2002. Glue semantics for HPSG. In Frank Van Eynde, Lars Hellan & Dorothee Beermann (eds.), *Proceedings of the 8th International Conference on Head-Driven Phrase Structure Grammar*, 1–19. Stanford, CA: CSLI Publications. `http://web.stanford.edu/group/cslipublications/cslipublications/HPSG/2001/Ash-Crouch-pn.pdf`.

Bos, Johan & Katja Markert. 2005. Recognising textual entailment with logical inference. In *Proceedings of human language technology conference and conference on empirical methods in natural language processing*, 628–635. Vancouver, British Columbia, Canada: Association for Computational Linguistics. `https://aclanthology.org/H05-1079`.

References ii

Bowman, Samuel R., Gabor Angeli, Christopher Potts & Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, 632–642. Lisbon, Portugal: Association for Computational Linguistics. DOI: 10.18653/v1/D15-1075.

Crouch, Dick & Josef van Genabith. 2000. Linear logic for linguists: ESSLLI-2000 course notes. Unpublished ms. available at `http://www.ling.ohio-state.edu/~pollard/681/crouch.pdf`.

Curry, Haskell B. & Robert Feys. 1958. *Combinatory logic: volume I*. Amsterdam: North Holland.

Dalrymple, Mary (ed.). 1999. *Semantics and syntax in Lexical Functional Grammar: the resource logic approach*. Cambridge, MA: MIT Press.

Dalrymple, Mary, John Lamping & Vijay Saraswat. 1993. LFG semantics via constraints. In Steven Krauwer, Michael Moortgat & Louis des Tombe (eds.), *Proceedings of the Sixth Conference of the European Chapter of the Association for Computational Linguistics (EACL 1993)*, 97–105. `https://www.aclweb.org/anthology/E93-1013.pdf`.

Donatelli, Lucia & Alexander Koller. 2023. Compositionality in computational linguistics. *Annual Review of Linguistics* 9. 463–481. `https://doi.org/10.1146/annurev-linguistics-030521-044439`.

Evang, Kilian. 2019. Transition-based DRS parsing using stack-LSTMs. In *Proceedings of the IWCS shared task on semantic parsing*, Gothenburg, Sweden: Association for Computational Linguistics. DOI: 10.18653/v1/W19-1202.

References iii

Findlay, Jamie Y. & Dag T. T. Haug. 2021. How useful are Enhanced Universal Dependencies for semantic interpretation? In *Proceedings of the Sixth International Conference on Dependency Linguistics (Depling, SyntaxFest 2021)*, 22–34. Sofia: Association for Computational Linguistics. `https://aclanthology.org/2021.depling-1.3`.

Findlay, Jamie Y., Ahmet Yıldırım, Saeedeh Salimifar & Dag T. T. Haug. 2023. Rule-based semantic interpretation for Universal Dependencies. In *Proceedings of the Sixth Workshop on Universal Dependencies (UDW, GURT/SyntaxFest 2023)*, 47–57. Association for Computational Linguistics. `https://aclanthology.org/2023.udw-1.6/`.

Frank, Anette & Josef van Genabith. 2001. GlueTag: linear logic based semantics for LTAG—and what it teaches us about LFG and LTAG. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG01 Conference*, 104–126. Stanford, CA: CSLI Publications. `http://web.stanford.edu/group/cslipublications/cslipublications/LFG/6/pdfs/lfg01frankgenabith.pdf`.

Garrette, Dan & Ewan Klein. 2009. An extensible toolkit for computational semantics. In *Proceedings of the eight international conference on computational semantics*, 116–127. Tilburg, The Netherlands: Association for Computational Linguistics. `https://aclanthology.org/W09-3712`.

Gotham, Matthew. 2018. Making Logical Form type-logical: Glue semantics for Minimalist syntax. *Linguistics and Philosophy* 41(5). 511–556. DOI: 10.1007/s10988-018-9229-z.

References iv

Haug, Dag T. T. & Jamie Y. Findlay. 2023. Formal semantics for Dependency Grammar. In *Proceedings of the Seventh International Conference on Dependency Linguistics (Depling 2023)*, Association for Computational Linguistics.

Haug, Dag T. T., Jamie Y. Findlay & Ahmet Yıldırım. 2023. The long and the short of it: DRASTIC, a semantically annotated dataset containing sentences of more natural length. In *Proceedings of the Fourth International Workshop on Designing Meaning Representations (DMR 2023)*, Association for Computational Linguistics.

Hepple, Mark. 1996. A compilation-chart method for linear categorial deduction. In *COLING '96: Proceedings of the 16th conference on computational linguistics*, vol. 1, 537–542. `https://doi.org/10.3115/992628.992721`.

Howard, William A. 1980. The formulae-as-types notion of construction. In *To H. B. Curry: essays on combinatory logic, lambda calculus, and formalism*, 479–490. London: Academic Press. Circulated in unpublished form from 1969.

Jiang, Nanjiang & Marie-Catherine de Marneffe. 2019. Evaluating BERT for natural language inference: A case study on the CommitmentBank. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 6086–6091. Hong Kong, China: Association for Computational Linguistics. DOI: 10.18653/v1/D19-1630.

Lev, Iddo. 2007. *Packed computation of exact meaning representations*. Stanford, CA. Ph.D. thesis, Stanford University.

References v

Liu, Jiangming, Shay B. Cohen, Mirella Lapata & Johan Bos. 2021. Universal discourse representation structure parsing. *Computational Linguistics* 47(2). 445–476. DOI: 10.1162/coli_a_00406.

McCoy, R. Thomas, Ellie Pavlick & Tal Linzen. 2019. Right for the wrong reasons: diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 3428–3448. Association for Computational Linguistics. DOI: 10.18653/v1/P19-1334.

Messmer, Moritz & Mark-Matthias Zymla. 2018. The Glue Semantics Workbench: a modular toolkit for exploring linear logic and Glue Semantics. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG'18 Conference*, 249–263. Stanford, CA: CSLI Publications. `http://cslipublications.stanford.edu/LFG/2018/lfg2018-messmer-zymla.pdf`.

Nivre, Joakim, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajic, Christopher Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers & Daniel Zeman. 2020. Universal Dependencies v2: an evergrowing multilingual treebank collection. In *Proceedings of the 12th International Conference on Language Resources and Evaluation (LREC 2020)*, 4034–4043. Marseille: European Language Resources Association. `https://aclanthology.org/2020.lrec-1.497`.

References vi

van Noord, Rik. 2019. Neural boxer at the IWCS shared task on DRS parsing. In *Proceedings of the IWCS shared task on semantic parsing*, Gothenburg, Sweden: Association for Computational Linguistics. DOI: 10.18653/v1/W19-1204.

van Noord, Rik, Lasha Abzianidze, Antonio Toral & Johan Bos. 2018. Exploring neural methods for parsing discourse representation structures. *Transactions of the Association for Computational Linguistics* 6. 619–633.

van Noord, Rik, Antonio Toral & Johan Bos. 2020. Character-level representations improve DRS-based semantic parsing even in the age of BERT. In *Proceedings of the 2020 conference on empirical methods in natural language processing (emnlp)*, 4587–4603. Online: Association for Computational Linguistics. DOI: 10.18653/v1/2020.emnlp-main.371.

Reddy, Siva, Oscar Täckström, Slav Petrov, Mark Steedman & Mirella Lapata. 2017. Universal semantic parsing. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, 89–101. Copenhagen, Denmark: Association for Computational Linguistics. DOI: 10.18653/v1/D17-1009. `https://aclanthology.org/D17-1009`.

Yao, Yuekun & Alexander Koller. 2022. Structural generalization is hard for sequence-to-sequence models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 5048–5062. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics. `https://aclanthology.org/2022.emnlp-main.337`.

References vii

Yıldırım, Ahmet & Dag Trygve Truslew Haug. 2023. Experiments in training transformer sequence-to-sequence DRS parsers. In *Proceedings of the 15th International Conference on Computational Semantics (IWCS 2023)*, Nancy, France: Association for Computational Linguistics.

Zeldes, Amir. 2017. The GUM corpus: creating multilayer resources in the classroom. *Language Resources and Evaluation* 51(3). 581–612. DOI: 10.1007/s10579-016-9343-x.

Why symbolic representations?

- 30 years ago, computational semantics would just have been formal semantics implement on a computer; not the case any longer.
- Why care about combining computational approaches and formal semantics?
  - From the computational side, **natural language inference** remains an unsolved task, where ML models are often wrong (or right for the wrong reasons …)
  - From the linguistic side, we get to study large-scale rule systems and discover unexpected interactions – as well as just how much we take for granted as input to the semantics (e.g. syntactic and lexical information)

Natural language inference

- **Natural language inference:** an in-demand NLP task – given a text, what follows?
    - Especially important for question-answering systems.

- Modern computational approaches treat NLI as a classification problem over sentence pairs, without explicit modelling of semantic structure.

(e.g. Bowman et al. 2015; Jiang & de Marneffe 2019)

- Results can be good, but seem often to rely on quirks of the datasets to extract incorrect generalisations that are effective, but brittle and easily misled (e.g. by passivisation).

(McCoy et al. 2019)

Natural Language Inference

- NLI can be straightforward given an suitable logical representation of the premise:

(5) Before John left, Mary slept.
$\Rightarrow$ John left.

(6) $\mathit{leave}(e_1, \mathrm{j}) \wedge \mathit{sleep}(e_2, \mathrm{m}) \wedge \mathit{before}(e_2, e_1)$

(7) $A \wedge B \vdash A$

(8) If Mary shouted, John ran.
$\not\Rightarrow$ Mary shouted.

(9) $\mathit{shout}(\mathrm{m}) \rightarrow \mathit{run}(\mathrm{j})$

(10) $A \rightarrow B \nvdash A$

- At the same time, purely logical entailments are rare – we often need access to e.g. lexical information. (Bos & Markert 2005)

Post-processing

- Output of rule-system is underspecified; language-specific information added when available, such as
  - Translation of UD relations to thematic roles

Jim loves Naomi.

$$\begin{array}{|l|}\hline e\ x\ y \\ \hline person(x) \\ Name(x, \text{‘}Jim\text{’}) \\ person(y) \\ Name(x, \text{‘}Naomi\text{’}) \\ love(e) \\ \mathbf{nsubj}(\mathbf{e}, \mathbf{x}) \\ \mathbf{obj}(\mathbf{e}, \mathbf{y}) \\ \hline \end{array} \Rightarrow \begin{array}{|l|}\hline e\ x\ y \\ \hline person(x) \\ Name(x, \text{‘}Jim\text{’}) \\ person(y) \\ Name(x, \text{‘}Naomi\text{’}) \\ love(e) \\ \mathbf{Experiencer}(\mathbf{e}, \mathbf{x}) \\ \mathbf{Source}(\mathbf{e}, \mathbf{y}) \\ \hline \end{array}$$

- Also (to come):
  - Idioms/multiword expressions
  - Anaphora resolution

What can/can't we achieve?

- We can't achieve any universality in the labels, as there is no universal approach to
  - word sense disambiguation
  - mapping from syntactic functions to semantic roles
  - mapping subordinators to adjunct clause functions
- Here, the ML is just much better
- So our focus is on building the correct semantic structure, i.e. the right graph of semantic relations
- We try to use universal rules, so should generalize across languages, though unfortunately the PMB has only English, German, Dutch and Italian → we need test data for more diverse languages

More challenging constructions

- Relativization: many languages have no overt indication of the gap site
    - we generate possible gap sites
- Control: no subject vs. object control distinction
    - underspecified relation
- Coordination: collapse of first conjunct and coordination as a whole, always assumes like function coordination