Extracting & Learning a Dependency-Enhanced Grammar for Dutch

Konstantinos Kogkalidis

June 2019

Overview

- Parsing as Deduction
- Extraction
- Supertagging
- Parsing

Parsing as Deduction

Intro: Categorial Grammars

- Lexicon $\mathcal{L}$: words $\rightarrow$ categories
- Categories defined by some inductive scheme
  - Atomic Categories: full phrases $\{\textsc{np}, \textsc{s}, \dots\}$
  - Complex Categories: fractionals $\{\textsc{np}\backslash\textsc{s}, \textsc{np}\backslash(\textsc{s}/\textsc{np}), \dots\}$
- Category Interactions: Combination Rules
- Parsing: Rule Application

Intro: Typelogical Grammars

- Lexicon $\mathcal{L}$: words $\rightarrow$ types
- Types defined by some inductive scheme
  - Atomic Types: full phrases $\{\textsc{np}, \textsc{s}, \ldots\}$
  - Complex Types: fractionals $\{\textsc{np}\backslash\textsc{s}, \textsc{np}\backslash(\textsc{s}/\textsc{np}), \ldots\}$
- Type Interactions: Logical Rules
- Parsing: Proof Search

Typelogical Grammars: Example (Lambek & co)

$$
\begin{aligned}
\mathcal{L} := \text{ducks} &: \textsc{np}, \\
\text{fish} &: \textsc{np}, \\
\text{fly} &: \textsc{np}\backslash\textsc{s}, \\
\text{eat} &: \textsc{np}\backslash(\textsc{s}/\textsc{np}), \\
\text{majestically} &: (\textsc{np}\backslash\textsc{s})\backslash(\textsc{np}\backslash\textsc{s})
\end{aligned}
$$

$$\frac{\Gamma \vdash B/A \quad \Delta \vdash A}{\Gamma, \Delta \vdash B}\ (/E) \qquad \frac{\Delta \vdash A \quad \Gamma \vdash A\backslash B}{\Delta, \Gamma \vdash B}\ (\backslash E)$$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash B/A}\ (/I) \qquad \frac{A, \Gamma \vdash B}{\Gamma \vdash A\backslash B}\ (\backslash I)$$

Directionality & Lexical Ambiguity

$$\begin{aligned}\mathcal{L} := \text{eten}_1 &: \textsc{np}\backslash(\textsc{s}/\textsc{np}), \\ \text{eten}_2 &: (\textsc{s}/\textsc{np})/\textsc{np} \\ \text{eten}_3 &: (\textsc{s}/\textsc{np}), \\ \text{eten}_{4,5} &: \textsc{np}\backslash(\textsc{np}\backslash\textsc{s}),\end{aligned}$$

eenden eten$_1$ vis (SVO)

eten$_2$ eenden vis? (VSO)

eten$_3$ vis! (VO)

eenden die vis eten$_{4,5}$ (SOV / OSV)

...

Directionality & Lexical Ambiguity

$$\begin{aligned}\mathcal{L} := \text{eten}_1 &: \text{NP}\backslash(\text{S}/\text{NP}), \\ \text{eten}_2 &: (\text{S}/\text{NP})/\text{NP} \\ \text{eten}_3 &: (\text{S}/\text{NP}), \\ \text{eten}_{4,5} &: \text{NP}\backslash(\text{NP}\backslash\text{S}),\end{aligned}$$

eenden eten_1 vis (SVO)

eten_2 eenden vis? (VSO)

eten_3 vis! (VO)

eenden die vis $\text{eten}_{4,5}$ (SOV / OSV)

...

☹

Beyond Directionality: MILL

Can we abstract directionality away?

Beyond Directionality: MILL

Can we abstract directionality away?

Yes: Multiplicative Intuionistic Linear Logic (ACG, LP ...)

Beyond Directionality: MILL

Can we abstract directionality away?

Yes: Multiplicative Intuionistic Linear Logic (ACG, LP . . .)

Types:

$$\mathrm{T} := \mathrm{A} \mid \mathrm{T}_1 \rightarrow \mathrm{T}_2$$

Beyond Directionality: MILL

Can we abstract directionality away?

Yes: Multiplicative Intuionistic Linear Logic (ACG, LP ...)

Types:

$$\mathrm{T} := \mathrm{A} \mid \mathrm{T}_1 \rightarrow \mathrm{T}_2$$

Logical Rules:

$$\frac{\Gamma \vdash A \rightarrow B \quad \Delta \vdash A}{\Gamma, \Delta \vdash B} \ (\rightarrow \mathrm{E}) \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \rightarrow B} \ (\rightarrow I)$$

MILL - The Good (1): Syntax-Semantics Interface

Curry-Howard Correspondence:

$$\text{MILL} \equiv \text{Simply-typed Linear } \lambda\text{-calculus}$$

MILL - The Good (1): Syntax-Semantics Interface

Curry-Howard Correspondence:

$$\text{MILL} \equiv \text{Simply-typed Linear } \lambda\text{-calculus}$$

$$\frac{\Gamma : f \vdash A \to B \quad \Delta : x \vdash A}{\Gamma, \Delta \vdash f(x) : B} \qquad \frac{\Gamma, x : A \vdash u : B}{\Gamma \vdash \lambda x.u : A \to B}$$

☺

MILL: The good (2): Lesser Lexical Ambiguity

$$\begin{aligned}
\mathcal{L} := \mathsf{eten}_1 &: \textsc{np}\backslash(\textsc{s}/\textsc{np}),\\
\mathsf{eten}_2 &: (\textsc{s}/\textsc{np})/\textsc{np}\\
\mathsf{eten}_3 &: (\textsc{s}/\textsc{np}),\\
\mathsf{eten}_{4,5} &: \textsc{np}\backslash(\textsc{np}\backslash\textsc{s}),
\end{aligned}$$

MILL: The good (2): Lesser Lexical Ambiguity

$$\begin{aligned} \mathcal{L} := \mathsf{eten}_1 &: \textsc{np}\backslash(\textsc{s}/\textsc{np}), \\ \mathsf{eten}_2 &: (\textsc{s}/\textsc{np})/\textsc{np} \\ \mathsf{eten}_3 &: (\textsc{s}/\textsc{np}), \\ \mathsf{eten}_{4,5} &: \textsc{np}\backslash(\textsc{np}\backslash\textsc{s}), \end{aligned}$$

$$\begin{aligned} \mathcal{L}' := \mathsf{eten}_{1,2,4,5} &: \mathrm{NP} \to \mathrm{NP} \to \mathrm{S}, \\ \mathsf{eten}_3 &: \mathrm{NP} \to \mathrm{S} \end{aligned}$$

☺

MILL - The Bad: Structural Ambiguity

$$\dfrac{\dfrac{\dfrac{}{\text{eten} \vdash \mathrm{NP} \rightarrow \mathrm{NP} \rightarrow \mathrm{S}} \rightarrow \mathit{Ax.} \quad \dfrac{}{\text{vis} \rightarrow \mathrm{NP}} \rightarrow \mathit{Ax.}}{\text{eten}, \text{vis} \vdash \mathrm{NP} \rightarrow \mathrm{S}} \rightarrow E \quad \dfrac{}{\text{eenden} \vdash \texttt{eenden} : \mathrm{NP}} \mathit{Ax.}}{\text{eenden}, \text{eten}, \text{vis} \vdash \mathrm{S}} \rightarrow E$$

`(eten vis) eenden`

$$\dfrac{\dfrac{\dfrac{}{\text{eten} \vdash \mathrm{NP} \rightarrow \mathrm{NP} \rightarrow \mathrm{S}} \rightarrow \mathit{Ax.} \quad \dfrac{}{\text{enden} \rightarrow \mathrm{NP}} \rightarrow \mathit{Ax.}}{\text{eten}, \text{eenden} \vdash \mathrm{NP} \rightarrow \mathrm{S}} \rightarrow E \quad \dfrac{}{\text{vis} \vdash \mathrm{NP}} \mathit{Ax.}}{\text{eenden}, \text{eten}, \text{vis} \vdash \mathrm{S}} \rightarrow E$$

`(eten eenden) vis`

☹

Dependency Decorations

Replace $\rightarrow$ with dependency-decorated variants:

$$\{\xrightarrow{\textsf{su}}, \xrightarrow{\textsf{obj}}, \xrightarrow{\textsf{predc}}, \xrightarrow{\textsf{mod}}, \ldots\}$$

$$\textsf{eten} : \mathrm{NP} \xrightarrow{\textsf{su}} \mathrm{NP} \xrightarrow{\textsf{obj}} \mathrm{S}$$

Dependency Decorations

Replace $\rightarrow$ with dependency-decorated variants:

$$\{\xrightarrow{\textsf{su}}, \xrightarrow{\textsf{obj}}, \xrightarrow{\textsf{predc}}, \xrightarrow{\textsf{mod}}, \ldots\}$$

$$\textsf{eten} : \mathrm{NP} \xrightarrow{\textsf{su}} \mathrm{NP} \xrightarrow{\textsf{obj}} \mathrm{S}$$

Lexical preferences + decorations $\implies$ reduced ambiguity

Dependency Decorations

Replace $\rightarrow$ with dependency-decorated variants:

$$\{\xrightarrow{\text{su}}, \xrightarrow{\text{obj}}, \xrightarrow{\text{predc}}, \xrightarrow{\text{mod}}, \dots\}$$

$$\text{eten} : \mathrm{NP} \xrightarrow{\text{su}} \mathrm{NP} \xrightarrow{\text{obj}} \mathrm{S}$$

Lexical preferences + decorations $\implies$ reduced ambiguity

Formally:

Unary modality $\Diamond^d$ for $d \in \{\text{su}, \text{obj}, \text{predc}, \text{mod}, \dots\}$

$$\frac{\Gamma \vdash A}{\langle \Gamma \rangle^d \vdash \Diamond^d A} \ (\Diamond^d I) \qquad \frac{\Delta \vdash \Diamond^d A \quad \Gamma[\langle A \rangle^d] \vdash B}{\Gamma[\Delta] \vdash B} \ (\Diamond^d E)$$

Extraction

Extraction: Intro

Goal

Syntactically-Annotated Corpus $\rightarrow$ Type Lexicon

Extraction: Intro

Goal

Syntactically-Annotated Corpus $\rightarrow$ Type Lexicon

Corpus

$$\text{Lassy-Small:} \quad \begin{cases} 65\,000 \text{ Sentences} \\ 1\,000\,000 \text{ Words} \\ \sim 30 \text{ Dependency Labels} \\ \sim 30 \text{ POS \& Phrasal Category Tags} \end{cases}$$

Extraction: Parameters

Parameters

- Translation Tables
  - Atomic Types: POS & Phrasal Category Tags
    $\{np: \textsc{np}, vnw: \textsc{vnw}, \ldots\}$
  - Implications: Dependency Labels
    $\{su: \xrightarrow{\text{su}}, obj: \xrightarrow{\text{obj}}, \ldots\}$
- Head Dependencies
  $\{hd, rhd, whd, crd, cmp\}$
- Modifier Dependencies
  $\{mod, app, predm\}$

Extraction: Algorithm

General Idea

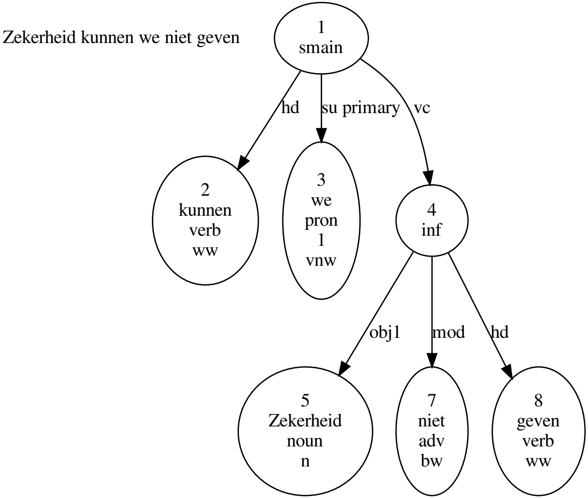
For each branch

- Find head, arguments, modifiers
- Collect & arrange argument types
- Type head as a functor from argument types to parent type
- Type modifiers as endomorphisms from parent type to itself

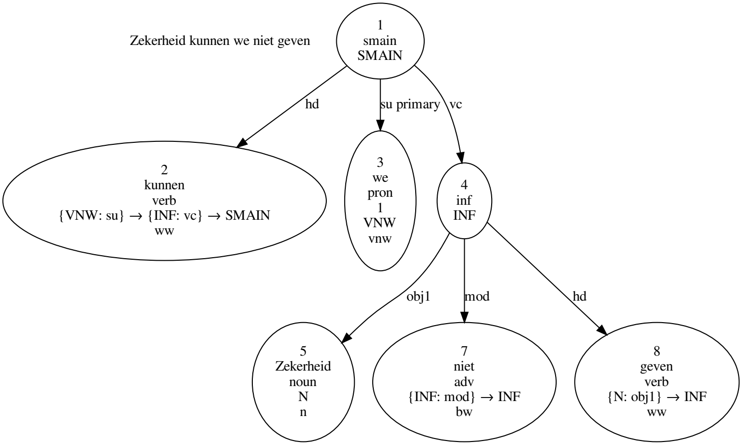
Hypothetical Reasoning

When type assigning arguments, consider internal “gaps”

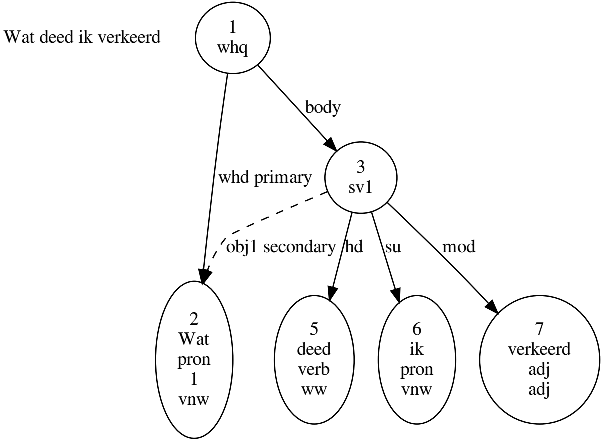
Extraction: Example (1)



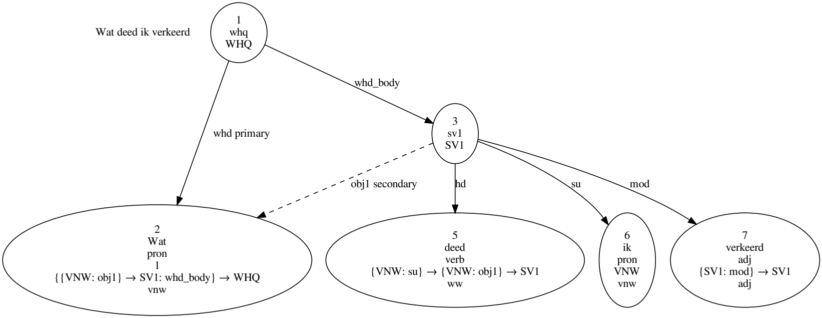
Extraction: Example (1)



Extraction: Example (2)



Extraction: Example (2)

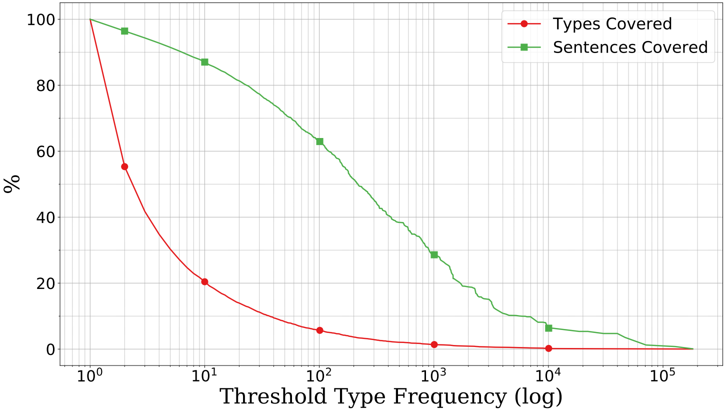


Extraction: Transformations

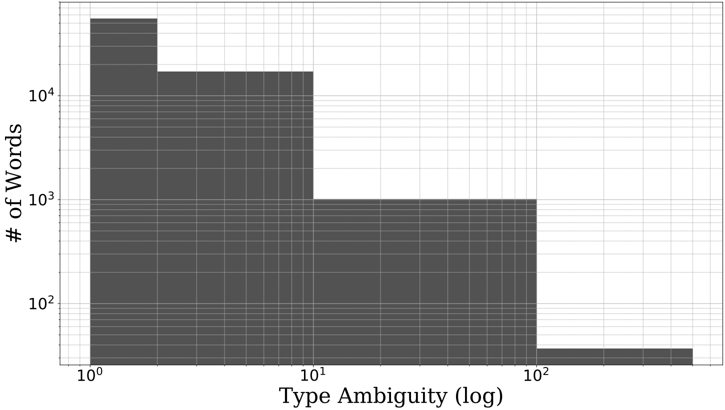
- Majority Voting
- Headless Branches
- Ellipses
- Determiners

Extraction: Results (1)

$\sim 5\,700$ unique types



Extraction: Results (2)



Supertagging

Supertagging: Standard Approach

Sequence Classification

Given input data sequence (word vectors)

predict a class for each sequence item (types)

Supertagging: Standard Approach

Sequence Classification

Given input data sequence (word vectors)
predict a class for each sequence item (types)

The problem

- Can't predict unseen items
- Difficulty predicting rare items

Supertagging: An Alternative

Type Syntax

A CFG of two meta-rules

$$\{(S \implies A) \;\forall\; A \in \mathcal{A}\}$$
$$\{(S \implies d\; S\; S) \;\forall\; d \in \mathcal{D}\}$$

Supertagging: An Alternative

Type Syntax

A CFG of two meta-rules

$$\{(S \implies A) \; \forall \; A \in \mathcal{A}\}$$
$$\{(S \implies d \; S \; S) \; \forall \; d \in \mathcal{D}\}$$

CFGs: learnable

Supertagging: An Alternative

Type Syntax

A CFG of two meta-rules

$$\{(S \implies A) \; \forall \; A \in \mathcal{A}\}$$
$$\{(S \implies d \; S \; S) \; \forall \; d \in \mathcal{D}\}$$

CFGs: learnable

Supertagging: learnable

Supertagging: An Alternative

Type Syntax

A CFG of two meta-rules

$$\{(S \implies A) \,\forall\, A \in \mathcal{A}\}$$
$$\{(S \implies d\ S\ S) \,\forall\, d \in \mathcal{D}\}$$

CFGs: learnable

Supertagging: learnable

CFG embedded within supertagging $\implies$ unbounded co-domain

Supertagging: Unbounded co-domain

Reformulation

Given input data sequence (word vectors)

generate an output sequence (atomic types & binary connectives)

Supertagging: Unbounded co-domain

Reformulation

Given input data sequence (word vectors)
generate an output sequence (atomic types & binary connectives)

Requirements

- Global Receptive Field (long-distance dependencies)
- Auto-Regressive (output conditional on prior output)

Options

- RNN encoder-decoder

Supertagging: Unbounded co-domain

Reformulation

Given input data sequence (word vectors)

generate an output sequence (atomic types & binary connectives)

Requirements

- Global Receptive Field (long-distance dependencies)
- Auto-Regressive (output conditional on prior output)

Options

- RNN encoder-decoder ☹

Supertagging: Unbounded co-domain

Reformulation

Given input data sequence (word vectors)
generate an output sequence (atomic types & binary connectives)

Requirements

- Global Receptive Field (long-distance dependencies)
- Auto-Regressive (output conditional on prior output)

Options

- RNN encoder-decoder ☹
- RNN encoder-separable decoder

Supertagging: Unbounded co-domain

Reformulation

Given input data sequence (word vectors)

generate an output sequence (atomic types & binary connectives)

Requirements

- Global Receptive Field (long-distance dependencies)
- Auto-Regressive (output conditional on prior output)

Options

- RNN encoder-decoder ☹
- RNN encoder-separable decoder ☹

Supertagging: Unbounded co-domain

Reformulation

Given input data sequence (word vectors)

generate an output sequence (atomic types & binary connectives)

Requirements

- Global Receptive Field (long-distance dependencies)
- Auto-Regressive (output conditional on prior output)

Options

- RNN encoder-decoder ☹
- RNN encoder-separable decoder ☹
- Attentive encoder-decoder

Supertagging: Unbounded co-domain

Reformulation

Given input data sequence (word vectors)

generate an output sequence (atomic types & binary connectives)

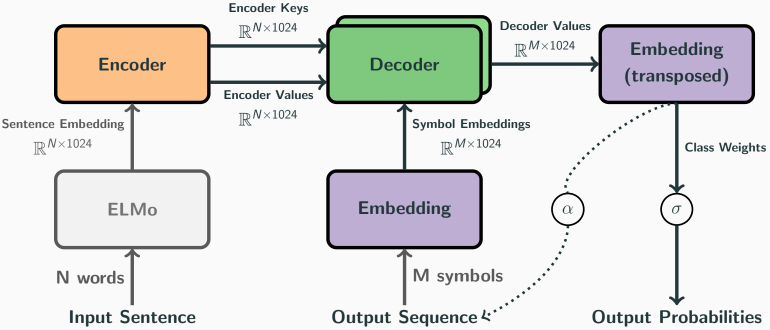
Requirements

- Global Receptive Field (long-distance dependencies)
- Auto-Regressive (output conditional on prior output)

Options

- RNN encoder-decoder ☹
- RNN encoder-separable decoder ☹
- Attentive encoder-decoder ☺

Supertagging: Model & Results



	Overall	Frequency: Unseen	Rare	Mid	High
Accuracy	88.05	19.2	45.68	65.62	89.93

Parsing

Parsing: Intro

Goal

From abstract syntax to surface syntax

Parsing: Intro

Goal

From abstract syntax to surface syntax

Parse $\equiv$ Proof

How can we navigate the proof space?

Parsing: General Framework

Parse State

A logical judgement

Word associations for (some) premises

A 1-step lookback

Algorithm

Given a parse state

1. Decide between introduction and elimination
2. Perform either
3. Update state(s)
4. Repeat

Parsing: Elimination

Given a sequence of word & type pairs

Split into two disjoint sequences..

Parsing: Elimination

Given a sequence of word & type pairs

Split into two disjoint sequences..

..by assigning each item one of two labels