From Parser to Query

Day 1. Introduction to Dependency Queries

Tom Rainsford Mathilde Regnault
thomas.rainsford@ling.uni-stuttgart.de
mathilde.regnault@ling.uni-stuttgart.de

10–11 June 2024

Outline

Outline

Plan of this workshop

- Day 1 : Parsed corpora and queries
  1. Introduction to Historical Parsed Corpora
  2. From constituency to dependency
  3. Querying UD corpora with Grew Match
- Day 2 : Parsing and querying your own texts
  1. Parsing with UD Pipe and HOPS Parser
  2. Examining parser results with GREW match
  3. Techniques for getting the best results

Plan of this workshop

- Day 1 : Parsed corpora and queries
  1. Introduction to Historical Parsed Corpora
  2. From constituency to dependency
  3. Querying UD corpora with Grew Match
- Day 2 : Parsing and querying your own texts
  1. Parsing with UD Pipe and HOPS Parser
  2. Examining parser results with GREW match
  3. Techniques for getting the best results

GitHub repository

`https://github.com/ILR-Stuttgart/from-parser-to-query`

- step-by-step guides for all exercises using the command-line
- sample data
- these slides

Outline

Practical Information about this workshop

Introduction to Historical Parsed Corpora

From Constituency to Dependency

What is a parsed corpus?

Parsed Corpus (Treebank)

An electronic corpus divided into words and sentences in which the syntactic relations between the words within each sentence are represented.

(1) John gave the book to his father.

- *John* is a nominal and is the subject of *gave*.
- *the book* is the object of *gave*, *the* determines *book*.
- *to his father* is the indirect object of *gave*, *his* determines *father* [and refers to *John*]

What is a parsed corpus?

Parsed Corpus (Treebank)

An electronic corpus divided into words and sentences in which the syntactic relations between the words within each sentence are represented.

(1) John gave the book to his father.

- *John* is a nominal and is the subject of *gave*.
- *the book* is the object of *gave*, *the* determines *book*.
- *to his father* is the indirect object of *gave*, *his* determines *father* [and refers to *John*]

What is a parsed corpus?

Parsed Corpus (Treebank)

An electronic corpus divided into words and sentences in which the syntactic relations between the words within each sentence are represented.

(1) John gave the book to his father.

- *John* is a nominal and is the subject of *gave*.
- *the book* is the object of *gave*, *the* determines *book*.
- *to his father* is the indirect object of *gave*, *his* determines *father* [and refers to *John*]

From Text to Query

1. Text

raw text

2. Parse

- 2a. add syntactic annotation
- 2b. verify annotation

3. Query

search for structures

How are gold corpora created ?

1. Text

raw text

2. Parse

- 2a. add syntactic annotation
- 2b. verify annotation

3. Query

search for structures

1. Annotation scheme and data format
   - Penn format (constituency, generative)
   - Universal Dependencies
2. Semi-automated or automated initial parsing
   - if resources available
3. Manual annotation and/or error correction
   - resource- and time-intensive

How are gold corpora created ?

1. Text

raw text

2. Parse

- 2a. add syntactic annotation
- 2b. verify annotation

3. Query

search for structures

1. Annotation scheme and data format
   - Penn format (constituency, generative)
   - Universal Dependencies
2. Semi-automated or automated initial parsing
   - if resources available
3. Manual annotation and/or error correction
   - resource- and time-intensive

How are gold corpora created ?

1. Text

raw text

2. Parse

- 2a. add syntactic annotation
- 2b. verify annotation

3. Query

search for structures

1. Annotation scheme and data format
   - Penn format (constituency, generative)
   - Universal Dependencies
2. Semi-automated or automated initial parsing
   - if resources available
3. Manual annotation and/or error correction
   - resource- and time-intensive

How are gold corpora created ?

1. Text

raw text

2. Parse

- 2a. add syntactic annotation
- 2b. verify annotation

3. Query

search for structures

1. Annotation scheme and data format
   - Penn format (constituency, generative)
   - Universal Dependencies
2. Semi-automated or automated initial parsing
   - if resources available
3. Manual annotation and/or error correction
   - resource- and time-intensive

How are parsed corpora used ?

1. Text

raw text

2. Parse

- 2a. add syntactic annotation
- 2b. verify annotation

3. Query

search for structures

- specialized corpus query programmes
  - CorpusSearch `https://corpussearch.sourceforge.net/`
  - ANNIS `https://corpus-tools.org/annis/`
  - GREW Match, etc... `https://match.grew.fr/`
- tailor-made scripts applied to parsed data files

How are parsed corpora used ?

1. Text

raw text

2. Parse

- 2a. add syntactic annotation
- 2b. verify annotation

3. Query

search for structures

- specialized corpus query programmes
  - CorpusSearch https://corpussearch.sourceforge.net/
  - ANNIS https://corpus-tools.org/annis/
  - GREW Match, etc... https://match.grew.fr/
- tailor-made scripts applied to parsed data files

How can we work with unparsed data?

1. Text

raw text

2. Parse

- 2a. add syntactic annotation
- 2b. verify annotation

3. Query

search for structures

1. manual annotation is extremely time-intensive...
   - ... so use a parser.
2. parsers are not perfect...
   - ... so correct (the worst) errors
   - ... and/or write more robust queries.
3. there's not always a trained model for historical texts...
   - ... so train your own from existing data
   - ... or try the model closest to the data.

How can we work with unparsed data ?

1. Text

raw text

2. Parse

- 2a. add syntactic annotation
- 2b. verify annotation

3. Query

search for structures

1. manual annotation is extremely time-intensive...
   - ... so use a parser.
2. parsers are not perfect...
   - ... so correct (the worst) errors
   - ... and/or write more robust queries.
3. there's not always a trained model for historical texts...
   - ... so train your own from existing data
   - ... or try the model closest to the data.

How can we work with unparsed data?

1. Text

raw text

2. Parse

- 2a. add syntactic annotation
- 2b. verify annotation

3. Query

search for structures

1. manual annotation is extremely time-intensive...
   - ... so use a parser.
2. parsers are not perfect...
   - ... so correct (the worst) errors
   - ... and/or write more robust queries.
3. there's not always a trained model for historical texts...
   - ... so train your own from existing data
   - ... or try the model closest to the data.

How can we work with unparsed data ?

1. Text

raw text

2. Parse

- 2a. add syntactic annotation
- 2b. verify annotation

3. Query

search for structures

1. manual annotation is extremely time-intensive...
   - ... so use a parser.
2. parsers are not perfect...
   - ... so correct (the worst) errors
   - ... and/or write more robust queries.
3. there's not always a trained model for historical texts...
   - ... so train your own from existing data
   - ... or try the model closest to the data.

How can we work with unparsed data?

1. Text

raw text

2. Parse

- 2a. add syntactic annotation
- 2b. verify annotation

3. Query

search for structures

1. manual annotation is extremely time-intensive...
   - ... so use a parser.
2. parsers are not perfect...
   - ... so correct (the worst) errors
   - ... and/or write more robust queries.
3. there's not always a trained model for historical texts...
   - ... so train your own from existing data
   - ... or try the model closest to the data.

How can we work with unparsed data?

1. Text

raw text

2. Parse

- 2a. add syntactic annotation
- 2b. verify annotation

3. Query

search for structures

1. manual annotation is extremely time-intensive...
   - ... so use a parser.
2. parsers are not perfect...
   - ... so correct (the worst) errors
   - ... and/or write more robust queries.
3. there's not always a trained model for historical texts...
   - ... so train your own from existing data
   - ... or try the model closest to the data.

How can we work with unparsed data?

1. Text

raw text

2. Parse

- 2a. add syntactic annotation
- 2b. verify annotation

3. Query

search for structures

1. manual annotation is extremely time-intensive...
   - ... so use a parser.
2. parsers are not perfect...
   - ... so correct (the worst) errors
   - ... and/or write more robust queries.
3. there's not always a trained model for historical texts...
   - ... so train your own from existing data
   - ... or try the model closest to the data.

How can we work with unparsed data ?

1. Text

raw text

2. Parse

- 2a. add syntactic annotation
- 2b. verify annotation

3. Query

search for structures

1. manual annotation is extremely time-intensive...
   - ... so use a parser.
2. parsers are not perfect...
   - ... so correct (the worst) errors
   - ... and/or write more robust queries.
3. there's not always a trained model for historical texts...
   - ... so train your own from existing data
   - ... or try the model closest to the data.

How can we work with unparsed data?

1. Text

raw text

2. Parse

- 2a. add syntactic annotation
- 2b. verify annotation

3. Query

search for structures

1. manual annotation is extremely time-intensive...
   - ... so use a parser.
2. parsers are not perfect...
   - ... so correct (the worst) errors
   - ... and/or write more robust queries.
3. there's not always a trained model for historical texts...
   - ... so train your own from existing data
   - ... or try the model closest to the data.

Outline

Practical Information about this workshop

Introduction to Historical Parsed Corpora

From Constituency to Dependency

Historical corpora in dependency and constituency

- **Constituency (Penn format)**

  https://www.ling.upenn.edu/hist-corpora/other-corpora.html
  - Germanic : English, German, Saxon, Icelandic, Faroese
  - Romance : French, Portuguese
  - Japanese
- **Dependency (UD)** https://universaldependencies.org/
  - Akkadian, Classical Chinese, French, Gothic, Ancient Greek, Icelandic and Faroese (converted), Irish, Latin, Sanskrit, Old East Slavic, Old Church Slavonic, Turkish

Historical corpora in dependency and constituency

- **Constituency (Penn format)**
  https://www.ling.upenn.edu/hist-corpora/other-corpora.html
  - Germanic : English, German, Saxon, Icelandic, Faroese
  - Romance : French, Portuguese
  - Japanese
- **Dependency (UD)** https://universaldependencies.org/
  - Akkadian, Classical Chinese, French, Gothic, Ancient Greek, Icelandic and Faroese (converted), Irish, Latin, Sanskrit, Old East Slavic, Old Church Slavonic, Turkish

Sample sentence

Song of Roland, l. 1–2

Carles li reis, nostre emperere magnes,
Set anz tuz pleins ad estet en Espaigne.
'Charles the King, our great emperor, has been in Spain for seven full long years'

Penn Format (constituency)

```
( (IP-MAT (NP-SBJ (NPRS Carles)
          (NP-PRN (D li) (NCS reis))
          (PON ,)
          (NP-PRN (DZ nostre)
              (NCS empere)
              (ADJP (ADJ magnes))))
      (PON ,)
      (NP-MSR (ADJNUM Set)
          (NCPL anz)
          (ADJP (Q tuz) (ADJ pleins)))
      (AJ ad)
      (VPP estet)
      (PP (P en)
          (NP (NPRS Espaigne)))
      (PONFP :))
  (ID 1100-ROLAND-MCVF-V,1.3))
```

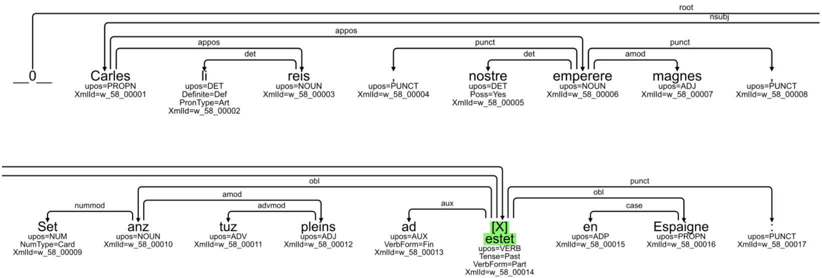
Quelle : Martineau, France, Paul Hirschbühler, Anthony Kroch, and Yves Charles Morin, eds. 2021. MCVF Corpus, parsed, version 2.0. https://github.com/beatrice57/mcvf-plus-ppchf

Conllu Format (dependency)

1	Carles	_	PROPN	NOMpro	_	14	nsubj	_	XmlId=w_58_00001
2	li	_	DET	DETdef	Definite=Def|	3	det	_	XmlId=w_58_00002
3	reis	_	NOUN	NOMcom	_	1	appos	_	SpaceAfter=No|XmlId=w_58_00003
4	,	_	PUNCT	PONfbl	_	6	punct	_	XmlId=w_58_00004
5	nostre	_	DET	DETpos	Poss=Yes	6	det	_	XmlId=w_58_00005
6	emperere	_	NOUN	NOMcom	_	1	appos	_	XmlId=w_58_00006
7	magnes	_	ADJ	ADJqua	_	6	amod	_	SpaceAfter=No|XmlId=w_58_00007
8	,	_	PUNCT	PONfbl	_	6	punct	_	XmlId=w_58_00008
9	Set	_	NUM	DETcar	NumType=C	10	nummod	_	XmlId=w_58_00009
10	anz	_	NOUN	NOMcom	_	14	obl	_	XmlId=w_58_00010
11	tuz	_	ADV	ADVgen	_	12	advmod	_	XmlId=w_58_00011
12	pleins	_	ADJ	ADJqua	_	10	amod	_	XmlId=w_58_00012
13	ad	_	AUX	VERcjg	VerbForm=F	14	aux	_	XmlId=w_58_00013
14	estet	_	VERB	VERppe	Tense=Past|	0	root	_	XmlId=w_58_00014
15	en	_	ADP	PRE	_	16	case	_	XmlId=w_58_00015
16	Espaigne	_	PROPN	NOMpro	_	14	obl	_	XmlId=w_58_00016
17	:	_	PUNCT	PONfbl	_	14	punct	_	XmlId=w_58_00017

Quelle : UD_Old_French-Profiterole2.14, `https://gitlab.huma-num.fr/profiterole/srcmf-ud`

Graph Format (dependency)



Source : GREW Match `https://universal.grew.fr`

Why do we prefer dependency?

- simpler, language-universal annotation scheme (Universal Dependencies)
- easier to parse, parsers more widely available
  - large community of developers
  - Berkeley Neural Parser for constituency

    https ://github.com/nikitakit/self-attentive-parser
- conversion constituency > dependency

Why do we prefer dependency?

- simpler, language-universal annotation scheme (Universal Dependencies)
- easier to parse, parsers more widely available
  - large community of developers
  - Berkeley Neural Parser for constituency

    https ://github.com/nikitakit/self-attentive-parser
- conversion constituency > dependency

Why do we prefer dependency?

- simpler, language-universal annotation scheme (Universal Dependencies)
- easier to parse, parsers more widely available
  - large community of developers
  - Berkeley Neural Parser for constituency

    https ://github.com/nikitakit/self-attentive-parser
- conversion constituency > dependency

Why do we prefer dependency?

- simpler, language-universal annotation scheme (Universal Dependencies)
- easier to parse, parsers more widely available
  - large community of developers
  - Berkeley Neural Parser for constituency

    https ://github.com/nikitakit/self-attentive-parser
- conversion constituency > dependency

Why do we prefer dependency?

- simpler, language-universal annotation scheme (Universal Dependencies)
- easier to parse, parsers more widely available
  - large community of developers
  - Berkeley Neural Parser for constituency

    https ://github.com/nikitakit/self-attentive-parser
- conversion constituency > dependency