Dependency Parsing

Seminar Question Answering

Cindy Fähnrich
Introduction

Dependency Grammar
  - Concept
  - Dependency Graphs

Dependency Parsing
  - Concept
  - Dynamic Programming
  - Constraint Satisfaction
  - Deterministic Parsing
  - Data-Driven Dependency Parsing

Conclusion
QA System - Placement

- Question
- Question Analysis
- Question Classification
- Query Construction
- Document Retrieval
- Sentence Retrieval
- Sentence Annotation
- Answer Extraction
- Answer Validation

- Named Entity Recognition
- **Dependency Parsing**
- Semantic Role Labeling
Dependency Parsing – What is it for?

- Belongs to natural language processing
- Aim: identify grammatical structure and syntactical relations
- Played only a marginal role until recently
The syntactical structure of a sentence consists of lexical items (words), linked by binary asymmetric relations between the words.

- Relations form the sentence structure.
- Relations create the dependencies between two words.
- Dependencies consist of a head and a dependent.
Dependency Grammar

- Dependency types are grammatical functions
- Verbs considered most often as head
- Criteria for identifying dependencies
- Various theories establish different criteria!
A dependency structure can be defined as a *directed graph* $G$:
- With a set $V$ of nodes
- With a set $E$ of edges

The edges in a *labeled graph* have dependency types, and nodes have word forms as labels.

This is an example sentence:

```
sentence
   an example
```

Has to fulfill some properties:
- **Connected** (weakly): all nodes are connected in the graph
- **Acyclic**: no cycles in graph
- **Single-head**: at most one head for each word
- **Projective**: no crossing of edges
- Computational analysis of syntactic sentence structure

- Parser implementations closely related to corresponding theory

- Non-projective dependency grammars are special case

- Three main approaches
  - Dynamic programming
  - Constraint satisfaction
  - Deterministic parsing
Grammar is regarded context-free

Use and modify parsers for context-free grammars like CYK

Dependencies are constituents, store subtrees for later reuse

Split main problem into smaller subproblems:
- Start by combining adjacent words and build dependencies
- Combine these subtrees again and build dependencies

Only production rules of $X \rightarrow u$, $X$ is a single non-terminal symbol

(an example) (example sentence)
- Grammar is regarded context-free

- Use and modify parsers for context-free grammars like CYK

- Dependencies are constituents, store subtrees for later reuse

- Split main problem into smaller subproblems:
  - Start by combining adjacent words and build dependencies
  - Combine these subtrees again and build dependencies

(An example sentence)
Constraint Satisfaction

- Uses constraint dependency grammar
  - Set of constraints
  - Can also be weighted

- Parsing **eliminates** representatives that contradict constraints

- Parsing has become constraint satisfaction problem
  - In general **NP complete**!
  - Parser design must ensure practical efficiency
This is an example sentence.
1. Verbs are roots.
1. Verbs are roots.
2. Nouns only modify verbs.
1. Verbs are roots.
2. Nouns only modify verbs.
3. Determiners only modify nouns on their right.
1. Verbs are roots.
2. Nouns only modify verbs.
3. Determiners only modify nouns on their right.
4. "example" always modifies nouns on its right.
Deterministic Parsing

- Try to derive a single dependency graph
- Deterministic sequence of parsing actions
- Incremental approach:
  - Try to link each word to its preceding one
  - Provide a function that determines valid dependencies
- Shift-reduce-type algorithms
Deterministic Parsing – Shift-Reduce

- **Shift**: Take a word on stack
- **Reduce**: Combine word with current words on stack

This is an example sentence.
Deterministic Parsing – Shift-Reduce

- **Shift**: Take a word on stack
- **Reduce**: Combine word with current words on stack

This is an example sentence.
Deterministic Parsing – Shift-Reduce

- **Shift**: Take a word on stack
- **Reduce**: Combine word with current words on stack

This is an example sentence.
Deterministic Parsing – Shift-Reduce

- **Shift:** Take a word on stack
- **Reduce:** Combine word with current words on stack

```
This is an example sentence.
```
Deterministic Parsing – Shift-Reduce

- Shift: Take a word on stack

- Reduce: Combine word with current words on stack

This is an example sentence.
Deterministic Parsing – Shift-Reduce

- **Shift**: Take a word on stack

- **Reduce**: Combine word with current words on stack

```
This is an example sentence.
```
Deterministic Parsing – Shift-Reduce

- **Shift**: Take a word on stack

- **Reduce**: Combine word with current words on stack

```
This is an example sentence.
```
Data-Driven Dependency Parsing

- Disambiguation of syntactical compositions?
  - Trainee corpus
  - Probabilistic models

- Deterministic Parsing
  - Requires an oracle for decisions → classifier
  - Train classifier with learning methods
  - Make parsing problem a classifier problem
Advantages

- Dependency links are close to semantic relation
- Reduced complexity: Parsing task becomes straight-forward
- Processing word by word instead of sentence
- Adequate treatment for languages with variable word orders

Disadvantages

- Non-projective dependency grammars are special case
- Dependency grammar is less expressive
  → What about dependencies between multiple words?
Questions
Sources
