Parsing of natural language sentences to syntactic and semantic graph representations

Abschlussvortrag zum Forschungsprojekt

Pius Meinert

April 13, 2018
Overview

Graph Representations

Corpora

Parsing Techniques

Parser
Overview

Graph Representations

Corpora

Parsing Techniques

Parser

Semantic: AMR, UCCA, dependency graphs

Syntactic: Constituency tree derived, Use of syntactic information
Overview

Graph Representations

Corpora

AMR, UCCA, SemEval-2014/-2015: dependency graphs, Penn Treebank, TIGER Corpus

Parsing Techniques

Parser
Overview

Graph Representations

Corpora

Parsing Techniques

Parser

Maximum Subgraph
Transition-Based
Synchronous HRG
Overview

- Graph Representations
- Corpora
- Parsing Techniques
- Parser
Abstract Meaning Representation (AMR) [Ban+13]

Possible say contrast include

- arg0-of
- arg1
- arg2
- arg3

You I person only

dream
Abstract Meaning Representation (AMR) [Ban+13]

rooted, directed, edge-labeled and leaf-labeled

contrast

possible

include

say

arg0-of

arg1

arg2

arg3

person

only

—

You

dream
A tree is a directed graph $G = (V, A)$ that has a vertex $r$, named root, such that every vertex $v \in V$ is reachable from $r$ via a unique directed path. [KJ15; KO16]
The gene thus can prevent a plant from fertilizing itself.
unconnected

**Connectedness:**
There exists an undirected path between every two pairs of vertices. Nodes with in- and out-degree zero are called singletons. [KO16]
unconnected, multi-rooted

Top nodes:
Nodes of in-degree zero, a graph’s equivalent to the unique root in a tree. [KO16]
unconnected, multi-rooted, reentrancy

Reentrant nodes:
Nodes with in-degree greater than one. [WXP15; DCS17; BB17]
The gene thus can prevent a plant from fertilizing itself.
Coverage ranges from 48% to 78% for various graph banks (CCGbank, Prage Semantic Dependencies, etc.). [KJ15; SCW17]
Das mer em Hans es huus hälfed aastriche
Account for 95.7 – 97.7% of the dependency structures that are used in [Cao+17].
The company that Mark wants to buy
The company that Mark wants to buy 

```
The company that Mark wants to buy
```
The company that Mark wants to buy
The company that Mark wants to buy
Evaluation Metric - AMR Representations

AMR graph

instance

want-01

ARG0

ARG1

instance

ARG0

instance

go-1

boy
Evaluation Metric - AMR Representations

AMR graph

PENMAN notation

\[(\text{w / want-01} :\text{arg0 (b / boy)} :\text{arg1 (g / go-01)} :\text{arg0 b})\]
Evaluation Metric - AMR Representations

AMR graph

logic format

\[
\text{instance}(a, \text{want-01}) \land \\
\text{instance}(b, \text{boy}) \land \\
\text{instance}(c, \text{go-01}) \land \\
\text{ARG0}(a, b) \land \\
\text{ARG1}(a, c) \land \\
\text{ARG0}(c, b)
\]
The boy wants the football

\[
\text{instance}(x, \text{want-01}) \land \\
\text{instance}(y, \text{boy}) \land \\
\text{instance}(z, \text{football}) \land \\
\text{ARG0}(x, y) \land \\
\text{ARG1}(x, z)
\]

The boy wants to go

\[
\text{instance}(a, \text{want-01}) \land \\
\text{instance}(b, \text{boy}) \land \\
\text{instance}(c, \text{go-01}) \land \\
\text{ARG0}(a, b) \land \\
\text{ARG1}(a, c) \land \\
\text{ARG0}(c, b)
\]
The boy wants the football

\[
\begin{align*}
\text{instance}(x, \text{want-01}) \land \\
\text{instance}(y, \text{boy}) \land \\
\text{instance}(z, \text{football}) \land \\
\text{ARG0}(x, y) \land \\
\text{ARG1}(x, z)
\end{align*}
\]

The boy wants to go

\[
\begin{align*}
\text{instance}(a, \text{want-01}) \land \\
\text{instance}(b, \text{boy}) \land \\
\text{instance}(c, \text{go-01}) \land \\
\text{ARG0}(a, b) \land \\
\text{ARG1}(a, c) \land \\
\text{ARG0}(c, b)
\end{align*}
\]

inter-annotator agreement study:

Smatch score ranges from 0.79 to 0.83.
Graph Parsing Techniques

Maximum Subgraph
“all pairs” approach [BM06] - Consider all possible (weighted) arcs and find the maximum spanning connected subgraph.

Transition-based
“stepwise” approach [BM06] - Build the graph step by step by applying transitions to the current configuration.

Synchronous Hyperedge Replacement Grammar (SHRG)
HRGs as “an intuitive generalization of context free grammars (CFGs) from strings to hypergraphs.” [Jon+12; Hab92]
Maximum Subgraph - Problem Definition [SCW17]

**Input**
directed, weighted graph \( G = (V, A) \) (complete)

**Implicit**
sentence \( s \), class of graphs \( \mathcal{G} \)

**Output**
subgraph \( G' = (V, A' \subseteq A) \) with maximum total weight such that \( G' \) belongs to \( \mathcal{G} \)

\[
G'(s) = \arg \max_{H \in \mathcal{G}(s, \mathcal{G})} \sum_{p \in H} \text{SCOREPART}(s, p)
\]

**Example**
if class of graphs \( \mathcal{G} \) is the class of all trees, Maximum Subgraph = Maximum Spanning Tree
\[ G'(s) = \arg \max_{H \in G(s,G)} \sum_{p \in H} \text{SCOREPART}(s, p) \]
Global learning

Optimize entire graph score, not only single arc attachments.

\[ G'(s) = \arg \max_{H \in G(s, G)} \sum_{p \in H} \text{SCOREPART}(s, p) \]
Global learning
Optimize entire graph score, not only single arc attachments.

\[ G'(s) = \arg \max_{H \in g(s, G)} \sum_{p \in H} \text{SCOREPART}(s, p) \]

Local features
Restricted to a limited number of arcs (to keep inference and learning tractable).
First published AMR parser. It solves the task by means of two phases:

**Concept identification**: Match spans of words to concept graph fragments.

**Relation identification**: Find the maximum spanning connected subgraph over those graph fragments.
The boy wants to visit New York City.
The boy wants to visit New York City.

Score function:

\[
\text{score}(b, c; \theta) = \sum_{i=1}^{k} \theta^{\top} f(w_{b_{i-1}:b_i}, b_{i-1}, b_i, c_i)
\]

Solve by dynamic programming: \(\mathcal{O}(n^2)\).
1. Initialization:
Include all edges and vertices given by the concept identification phase.

2. Pre-processing:
Reduce the set of edges considered to one edge per pair of nodes: Either the edge given by the first phase or the highest scoring one.

3. Core algorithm:
First, add all positive edges and then greedily add the least negative edge that connects two components until the graph is connected.
A transition system for parsing is a tuple $S = (S, T, s_0, S_t)$ where

- $S$ is a set of parsing **states** (configurations).
- $T$ is a set of parsing **actions** (transitions), each of which is a function $t : S \rightarrow S$.
- $s_0$ is an **initialization function**, mapping each input sentence $w$ to an **initial state**.
- $S_t \subseteq S$ is a set of **terminal states**.
**Transition-Based - Parsing Algorithm [WXP15]**

**Input:** sentence \( w = w_0 ... w_n \)

**Output:** parsed graph \( G \)

1: \( s \leftarrow s_0(w) \)

2: while \( s \notin S_t \) do

3: \( \mathcal{T} \leftarrow \) all possible actions according to \( s \)

4: \( bestT \leftarrow \arg \max_{t \in \mathcal{T}} \text{score}(t, s) \)

5: \( s \leftarrow \) apply \( bestT \) to \( s \)

6: end while

7: return \( G \)
bestT ← arg max_{t ∈ T} score(t, s)
Local learning  Optimization only for single transitions, not transition sequences.

\[ \text{bestT} \leftarrow \arg \max_{t \in \mathcal{T}} \text{score}(t, s) \]
Local learning

Optimization only for single transitions, not transition sequences.

$$\text{bestT} \leftarrow \arg \max_{t \in T} \text{score}(t, s)$$

Global features

Features may be based on whole graph built so far/entire transition history.
Idea: Use similarities between an AMR and the dependency structure of a sentence.

Two-stage framework:

1) **dependency parser** to generate dependency tree for the sentence

2) **transition-based** algorithm to transform dependency tree to an AMR graph

The dependency parser can be trained on a much larger data set.
REENTRANCE action

\[
\text{want} \rightarrow \text{police} \rightarrow \text{arrest} \\
\text{reentrance}
\]

\[
\text{want} \rightarrow \text{police} \rightarrow \text{arrest}
\]

REPLACE-HEAD action

\[
\text{live} \rightarrow \text{in} \rightarrow \text{Singapore} \\
\Rightarrow
\]

\[
\text{live} \rightarrow \text{Singapore}
\]
Towards a Catalogue of Linguistic Graph Banks [KO16]

- graph structures are of growing relevance to much NLP research
- provide common terminology and transparent statistics for different (collections of) graphs
- propose to establish shared community resource: https://aclweb.org/aclwiki/Graph_Parsing_(State_of_the_Art)

