Natural Language Processing

Transition-Based Dependency Parsing
Transition-based dependency parsing

- Eisner’s algorithm runs in time $O(|w|^3)$. This may be too much if a lot of data is involved.

- **Idea:** Design a dumber but really fast algorithm and let the machine learning do the rest.

- Eisner’s algorithm searches over many different dependency trees at the same time.

- A transition-based dependency parser only builds *one* tree, in *one* left-to-right sweep over the input.
Transition-based dependency parsing

• The parser starts in an initial configuration.

• At each step, it asks a guide to choose between one of several transitions (actions) into new configurations.

• Parsing stops if the parser reaches a terminal configuration.

• The parser returns the dependency tree associated with the terminal configuration.
Configuration c = parser.getInitialConfiguration(sentence)

while c is not a terminal configuration do

    Transition t = guide.getNextTransition(c)

    c = c.makeTransition(t)

return c.getGraph()
Guides

- We need a guide that tells us what the next transition should be.
- The task of the guide can be understood as **classification**: Predict the next transition (class), given the current configuration.
Training a guide

• We let the parser run on gold-standard trees.
• Every time there is a choice to make, we simply look into the tree and do ‘the right thing’™.
• We collect all (configuration, transition) pairs and train a classifier on them.
• When parsing unseen sentences, we use the trained classifier as a guide.
Training a guide

• The number of (configuration, transition) pairs is far too large.

• We define a set of features of configurations that we consider to be relevant for the task of predicting the next transition.

  Example: word forms of the topmost two words on the stack and the next two words in the buffer

• We can then describe every configuration in terms of a feature vector.
Transition-based dependency parsing

Training a guide

configurations in which we want to do la

configurations in which we want to do ra
Transition-based dependency parsing

Training a guide

Score for feature 1

Score for feature 2

Classification function learned by the classifier
Training a guide

• In practical systems, we have thousands of features and hundreds of transitions.

• There are several machine-learning paradigms that can be used to train a guide for such a task.

  Examples: perceptron, decision trees, support-vector machines
The arc-standard algorithm

- The arc-standard algorithm is a simple algorithm for transition-based dependency parsing.
- It is very similar to shift–reduce parsing as it is known for context-free grammars.
- It is implemented in most practical transition-based dependency parsers, including MaltParser.
The arc-standard algorithm

**Configurations**

A configuration for a sentence $w = w_1 \ldots w_n$ consists of three components:

- a **buffer** containing words of $w$
- a **stack** containing words of $w$
- the **dependency graph** constructed so far
Configurations

- **Initial configuration:**
  - All words are in the buffer.
  - The stack is empty.
  - The dependency graph is empty.

- **Terminal configuration:**
  - The buffer is empty.
  - The stack contains a single word.
The arc-standard algorithm

Possible transitions

• **shift (sh):** push the next word in the buffer onto the stack

• **left-arc (la):** add an arc from the topmost word on the stack, $s_1$, to the second-topmost word, $s_2$, and pop $s_2$

• **right-arc (ra):** add an arc from the second-topmost word on the stack, $s_2$, to the topmost word, $s_1$, and pop $s_1$
The arc-standard algorithm

Example run

Stack

Buffer

I booked a flight from LA
The arc-standard algorithm

Example run

Stack

Buffer

I booked a flight from LA

I booked a flight from LA

sh
Example run

Stack

Buffer

I booked a flight from LA
The arc-standard algorithm

Example run

Stack

Buffer

I

booked

a

flight

from LA

I booked a flight from LA

sh
The arc-standard algorithm

Example run

Stack

Buffer

I booked a flight from LA
The arc-standard algorithm

Example run

Stack

Buffer

I booked a flight from LA

I booked a flight from LA

la-subj
Example run

The arc-standard algorithm

Stack

Buffer

booked

a
flight
from LA

I booked a flight from LA
The arc-standard algorithm

Example run

Stack

Buffer

I booked a flight from LA

sh
The arc-standard algorithm

Example run

Stack
- booked
- a

Buffer
- flight
- from LA

subj

I booked a flight from LA
The arc-standard algorithm

Example run

Stack
booked  a

Buffer
flight  from LA

subj

I booked a flight from LA

sh
The arc-standard algorithm

Example run

I booked a flight

subj

from LA
The arc-standard algorithm

Example run

Stack

Buffer

I booked a flight from LA

la-det
Example run

The arc-standard algorithm

Stack

Buffer

I booked a flight from LA

subj det

I booked a flight from LA
The arc-standard algorithm

Example run

Stack

Buffer

subj

det

I booked

a flight from LA

sh
Example run

The arc-standard algorithm

Stack

| booked | flight | from LA |

Buffer

subj

I booked

det

a flight from LA
The arc-standard algorithm

Example run

I booked a flight from LA

ra-pmod
The arc-standard algorithm

Example run

Stack

Buffer

I booked a flight

I booked a flight from LA
The arc-standard algorithm

Example run

Stack

Buffer

ra-dobj

I booked a flight from LA

subj det pmod

l booked a flight from LA
The arc-standard algorithm
The arc-standard algorithm

Example run

UPPSALA UNIVERSITET

Stack

Buffer

booked

I booked a flight from LA

done!