Natural Language Processing

Dependency Parsing
Dependency grammar

- The term ‘dependency grammar’ does not refer to a specific grammar formalism.
- Rather, it refers to a specific way to describe the syntactic structure of a sentence.
The notion of dependency

- The basic observation behind **constituency** is that groups of words may act as one unit.

  *Example:* noun phrase, prepositional phrase

- The basic observation behind **dependency** is that words have grammatical functions with respect to other words in the sentence.

  *Example:* subject, modifier
I booked a flight from LA.
Dependency trees

- In an arc $h \rightarrow d$, the word $h$ is called the head, and the word $d$ is called the dependent.
- The arcs form a rooted tree.
The history of dependency grammar

- The notion of dependency can be found in some of the earliest formal grammars.
- Modern dependency grammar is attributed to Lucien Tesnière (1893–1954).
- Recent years have seen a revived interest in dependency-based description of natural language syntax.
Linguistic resources

• Descriptive dependency grammars exist for some natural languages.

• Dependency treebanks exist for a wide range of natural languages.

• These treebanks can be used to train accurate and efficient dependency parsers.
Just like phrase structure parsing, dependency parsing has to deal with ambiguity.
Just like phrase structure parsing, dependency parsing has to deal with ambiguity.
Disambiguation

• We need to disambiguate between alternative analyses.

• We develop mechanisms for scoring dependency trees, and disambiguate by choosing a dependency tree with the highest score.
Distinguish two aspects:

• **Scoring model:**
  How do we want to score dependency trees?

• **Parsing algorithm:**
  How do we compute a highest-scoring dependency tree under the given scoring model?
The arc-factored model

• To score a dependency tree, score the individual arcs, and combine the score into a simple sum.

\[ \text{score}(t) = \text{score}(a_1) + \ldots + \text{score}(a_n) \]

• Define the score of an arc \( h \rightarrow d \) as the weighted sum of all features of that arc:

\[ \text{score}(h \rightarrow d) = f_1w_1 + \ldots + f_nw_n \]
Examples of features

• ‘The head is a verb.’
• ‘The dependent is a noun.’
• ‘The head is a verb
  and the dependent is a noun.’
• ‘The head is a verb
  and the predecessor of the head is a pronoun.’
• ‘The arc goes from left to right.’
• ‘The arc has length 2.’
Training using structured prediction

• Take a sentence $w$ and a gold-standard dependency tree $g$ for $w$.

• Compute the highest-scoring dependency tree under the current weights; call it $p$.

• Increase the weights of all features that are in $g$ but not in $p$.

• Decrease the weights of all features that are in $p$ but not in $g$. 
Collins’ algorithm: Straightforward adaptation of CKY to dependency trees. Runs in $O(w^5)$ time.

Eisner’s algorithm: Improves complexity by building the left and right halves of trees independently. Runs in $O(w^3)$ time.