Maximum Spanning Tree Algorithm for Non-projective Labeled Dependency Parsing

Nobuyuki Shimizu (清水伸幸)
Ph D student
Computer Science
State University of New York at Albany
Outline

• Preprocessing before running CLE
  – CLE does not label edges.

• Features
  – We used 27 very simple features
Labeling Edges Before CLE

- Create a complete directed graph for each sentence.
  - Exactly one edge from i to j, and one from j to i
  - No self pointing edge
- Label the directed edges with the highest scoring dependency relation.
  - For each edge, for each dependency relation, find the score. If it is higher than the previous score, record it.
Labeling Edges Before CLE

• Advantage
  - Dependency Relation can be used as a feature while we execute CLE.

• Disadvantage
  - The structure of the dependency tree is not known while determining the edge labels.
Features

- We used the combinations of:
  - Dependency Relation
  - Coarse POS
  - Word Token
  - Direction of the edge (left or right)
  - Distance between the start and end point of the edge
- Nothing else
Features

• Every feature contains
  − Dependency Relation
  − Direction of the edge (left or right)

• Only 27 combinations of the above
  − Well under 4 GB of memory
  − Very simple and easy on your machine
Results

• The results are around the average
  - Despite one iteration of training
  - Despite the minimal features used
    • We don't use fine-grained POS tags
Question

• Given the same features, should we determine edge labels before the CLE, or after the CLE?