Tree Edit Distance for Recognizing Textual Entailment: Estimating the Cost of Insertion

Milen Kouylekov & Bernardo Magnini

University of Trento,
ITC-irst, Centro per la Ricerca Scientifica e Tecnologica,
Trento, Italy

Venezia, April 10, 2005
Our Goals

- Continue the development of a system based on Tree Edit Distance
- Investigate the cost of Insertion
- Combining different system settings using learning algorithm
The Textual Entailment Framework


- An **Entailment Relation** holds between two text fragments (i.e. **text T** and **hypothesis H**) when the meaning of H, as interpreted in the context of T, can be inferred from the meaning of T.

- **Entailment Rules (patterns)** is directional relation between two parse sub-trees with variables, where the first one entails the second.
Entailment and Tree Edit Distance

- Represent $T$ and $H$ as dependency trees
- The probability of an entailment relation between $T$ and $H$ is related to the mappings between $H$ and $T$
- Mappings can be described as a sequence of editing operations that transform $T$ into $H$
- Each edit operation has a cost assigned to it
- Entailment holds if the overall transformation cost is below a certain threshold, estimated over the training data.
Tree Edit Distance on Dependency Trees

- (Zhang and Shasha, 1990) Tree Edit Distance algorithm has been implemented.
- Edit operations (Insertion, Deletion, Substitution) are allowed on single nodes only.
- Parsing is performed with Minipar (Lin 1998).
- Node order is relevant: nodes are re-arranged according to: subj --> obj --> mods.
- The original algorithm does not consider labels on edges: relations names are concatenated to node names.
- E.g. eat [subj] John       eat --> John#subj
Cost functions

- **Insertion**: the cost of inserting a node $w$ in $T$ should be proportional to the relevance of $w$ in the context of $H$.

- **Deletion**: the cost of deleting a node $w$ in $T$ should be proportional to the relevance of $w$ in the context of $T$.

- **Substitution**: the cost of substituting a node $w_1#rel1$ in $T$ with a node $w_2#rel2$ in $H$ is proportional to the strength of the entailment relation between the two nodes and relevant to the context of $H$ and $T$. 

Example

T: The Statue of Liberty is so big it had to be built in 300 sections.
H: The Statue of Liberty was built in the year 300.
Example (2)

T: Mount Olympus towers up from the center of the earth.
H: Mount Olympus is in the center of the earth.
System Settings

- System 1: Insertion as IDF
- System 2: Fixed Insert cost
- System 3: Number of Parents.
- System 4: Number of Children.
- System 5: Number of Children + Number of Parents
- System 6: Combined

- All previous systems as features of the sequential minimal optimization (SMO) algorithm – training a support vector classifier
## System Settings - Performance

<table>
<thead>
<tr>
<th></th>
<th>development</th>
<th>cross-validation</th>
<th>test</th>
</tr>
</thead>
<tbody>
<tr>
<td>idf</td>
<td>0.581</td>
<td>0.578</td>
<td>0.572</td>
</tr>
<tr>
<td>fixed</td>
<td>0.591</td>
<td>0.560</td>
<td>0.570</td>
</tr>
<tr>
<td>#parents</td>
<td>0.600</td>
<td>0.590</td>
<td>0.582</td>
</tr>
<tr>
<td>#children</td>
<td>0.579</td>
<td>0.579</td>
<td>0.541</td>
</tr>
<tr>
<td>#ch + #par</td>
<td>0.598</td>
<td>0.590</td>
<td>0.571</td>
</tr>
<tr>
<td>combined</td>
<td>0.637</td>
<td>0.613</td>
<td>0.605</td>
</tr>
</tbody>
</table>

- Combined run is the best performing
- Additional resources for calculating the insertion cost are not needed
Our system performs well on the Summarization task.
IE requires a large resource of complex entailment rules.
Combined run is accurate but less precise.
Thank You!