Unsupervised Generation of Parallel Treebanks through Sub-Tree Alignment

Ventsislav Zhechev
contact@ventsislavzhechev.eu
Talk Outline

• What is a Parallel Treebank and Why do we Need One?
• System Design and Features
• Software Package Details
• Avenues for Improvement
• Conclusions
What is a Parallel Treebank?
English
I do not think it is necessary for classic cars to be part of the directive.
I am not looking for such rigidly high recycling quotas when it comes to special-purpose vehicles either.
I want special-purpose vehicles such as ambulances to have high recovery quotas.
This is my main concern in this matter.

German
Ich halte es nicht für notwendig, daß Oldtimer Bestandteil dieser Richtlinie sind.
Auch bei Sonderfahrzeugen strebe ich nicht so unbedingt hohe Recyclingquoten an.
Ich habe den Wunsch, daß Sonderfahrzeuge wie Krankenwagen hohe Rettungsquoten haben.
Das ist meine Hauptsorge in diesem Bereich.
What is a Parallel Treebank?

English

This is my main concern.

German

Das ist meine Hauptsorge in diesem Bereich.
What is a Parallel Treebank?

English

This is my main concern.

German

Das ist meine Hauptsorge in diesem Bereich.
This is my main concern in this matter.

Das ist meine Hauptersorg in diesem Bereich.
What is a Parallel Treebank?

English

German

This is my main concern.

Das ist meine Hauptsorge in diesem Bereich.
Uses of Parallel Treebanks
Uses of Parallel Treebanks

- Hiero (Chiang, 2007)
- Probabilistic Synchronous Tree-Insertion Grammars (Nesson et al., 2006)
- Data-Oriented Translation (Hearne and Way, 2006)
- Stat-XFER (Lavie, 2008)
Uses of Parallel Treebanks

English

German

This is my main concern in this matter.

Das ist meine Hauptsorge in diesem Bereich.
Uses of Parallel Treebanks
What do we need to generate Parallel Treebanks?
What do we need to generate Parallel Treebanks?

- We already have:
  - Parallel Corpora
  - Parsers
- What’s missing?
  - A Sub-Tree Aligner
- for existing research see
  (Tinsley et al., 2007), MT Summit XI
  (Zhechev, to appear), PhD Thesis
What do we want in a Sub-Tree Aligner?
What do we want in a Sub-Tree Aligner?

- Independence
- Preservation
- Minimal External Resources
- Guided Lexical Alignments
How does the Sub-Tree Aligner Work?
How does the Sub-Tree Aligner Work?

- Prerequisites:
  - sentence-aligned parallel text
  - monolingual parsers for both languages
  - source-to-target and target-to-source word-alignment probability tables
- Both sides of the parallel text need to be parsed in advance
- The sub-tree aligner operates on one parsed sentence pair at a time
How does the Sub-Tree Aligner Work?

• Initialisation
  • extract the relevant word-alignment data
  • calculate scores for all possible links between nodes in the source and target tree
    • only nonzero scores are stored

• Selection
  • select the best combination of links for the sentence pair
  • two selection algorithms
Translational Equivalence
Translational Equivalence

\[
\begin{align*}
\text{inside} & \quad s_l = \langle b \ c \rangle \\
& \quad t_l = \langle x \ y \rangle \\
\text{outside} & \quad \bar{s}_l = \langle a \rangle \\
& \quad \bar{t}_l = \langle w \ z \rangle
\end{align*}
\]

\[
\gamma(\langle s, t \rangle) = \alpha(s_l|t_l) \cdot \alpha(t_l|s_l) \cdot \alpha(\bar{s}_l|\bar{t}_l) \cdot \alpha(\bar{t}_l|\bar{s}_l)
\]

\[
\text{score1} \quad \alpha(y|x) = \prod_i \sum_j P(y_j|x_i)
\]
**Translational Equivalence**

\[
\gamma(\langle s, t \rangle) = \alpha(s_l|t_l) \cdot \alpha(t_l|s_l) \cdot \alpha(\bar{s}_l|\bar{t}_l) \cdot \alpha(\bar{t}_l|\bar{s}_l)
\]

**inside**
\[
s_l = \langle b, c \rangle \\
t_l = \langle x, y \rangle
\]

**outside**
\[
\bar{s}_l = \langle a \rangle \\
\bar{t}_l = \langle w, z \rangle
\]

**score**
\[
\alpha(y|x) = \prod_j \sum_i P(y_j|x_i) / |x|
\]
Greedy-Search Selection
while unprocessed hypotheses remain do
link the highest-scoring hypothesis
discard all incompatible hypotheses
end while
Two hypotheses are incompatible if
• they share a node

**Greedy-Search Selection**

```
while unprocessed hypotheses remain do
  link the highest-scoring hypothesis
  discard all incompatible hypotheses
end while
```

- Two hypotheses are incompatible if
  - they share a node
Two hypotheses are incompatible if

- they share a node

while unprocessed hypotheses remain do

- link the highest-scoring hypothesis
- discard all incompatible hypotheses

end while
Greedy-Search Selection

while unprocessed hypotheses remain do
    link the highest-scoring hypothesis
    discard all incompatible hypotheses
end while

- Two hypotheses are incompatible if
  - they share a node
  - descendants / ancestors of the source node are linked to non-descendants / non-ancestors of the target node
Two hypotheses are incompatible if
• they share a node
• descendants / ancestors of the source node are linked to non-descendants / non-ancestors of the target node

Greedy-Search Selection

\[
\text{while unprocessed hypotheses remain do } \begin{align*}
\text{link the highest-scoring hypothesis} \\
\text{discard all incompatible hypotheses} \\
\text{end while}
\end{align*}
\]
while unprocessed hypotheses remain do
  link the highest-scoring hypothesis
  discard all incompatible hypotheses
end while

Two hypotheses are incompatible if
• they share a node
• descendants / ancestors of the source node
  are linked to non-descendants / non-ancestors of
  the target node
while unprocessed hypotheses remain
do link the highest-scoring hypothesis
discard all incompatible hypotheses
end while

while unprocessed hypotheses remain do
link the highest-scoring hypothesis
discard all incompatible hypotheses
end while

Two hypotheses are incompatible if
• they share a node
• descendants / ancestors of the source node
are linked to non-descendants / non-ancestors of 
the target node
while unprocessed hypotheses remain do
  link the highest-scoring hypothesis
  discard all incompatible hypotheses
end while

• Two hypotheses are incompatible if
  • they share a node
  • descendants / ancestors of the source node
    are linked to non-descendants / non-ancestors of
    the target node
while unprocessed hypotheses remain do
  link the highest-scoring hypothesis
  discard all incompatible hypotheses
end while

• Two hypotheses are incompatible if
• they share a node
• descendants / ancestors of the source node
  are linked to non-descendants / non-ancestors of
  the target node
while unprocessed hypotheses remain do
  link the highest-scoring hypothesis
  discard all incompatible hypotheses
end while

Two hypotheses are incompatible if
• they share a node
• descendants / ancestors of the source node are linked to non-descendants / non-ancestors of the target node
while unprocessed hypotheses remain do
    link the highest-scoring hypothesis
    discard all incompatible hypotheses
end while
while unprocessed hypotheses remain do
  link the highest-scoring hypothesis
  discard all incompatible hypotheses
end while

Two hypotheses are incompatible if
• they share a node
• descendants / ancestors of the source node are linked to non-descendants / non-ancestors of the target node
while unprocessed hypotheses remain do
    link the highest-scoring hypothesis
    discard all incompatible hypotheses
end while
while unprocessed hypotheses remain
  do
    link the highest-scoring hypothesis
  discard all incompatible hypotheses
end while
Greedy-Search Selection

\[
\text{while unprocessed hypotheses remain do} \\
\quad \text{link the highest-scoring hypothesis} \\
\quad \text{discard all incompatible hypotheses} \\
\text{end while}
\]

- Two hypotheses are incompatible if
  - they share a node
  - descendants / ancestors of the source node are linked to non-descendants / non-ancestors of the target node
- Several hypotheses may share the highest translational equivalence score!
while unprocessed hypotheses with no tied competitors remain do
    while the highest-scoring hypothesis has tied competitors do
        skip all tied competitors
    end while
    link the highest-scoring hypothesis
    discard all incompatible hypotheses
end while

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>score_1</th>
</tr>
</thead>
<tbody>
<tr>
<td>a ⇔ b</td>
<td>a ⇔ c</td>
<td>score_1</td>
</tr>
<tr>
<td>d ⇔ c</td>
<td>e ⇔ f</td>
<td>score_2</td>
</tr>
<tr>
<td>g ⇔ b</td>
<td>h ⇔ i</td>
<td>score_3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>score_4</td>
</tr>
</tbody>
</table>
while unprocessed hypotheses with no tied competitors remain do
while the highest-scoring hypothesis has tied competitors do
    skip all tied competitors
end while
link the highest-scoring hypothesis
discard all incompatible hypotheses
end while

<table>
<thead>
<tr>
<th>a ↔ b</th>
<th>score_1</th>
</tr>
</thead>
<tbody>
<tr>
<td>a ↔ c</td>
<td>score_1</td>
</tr>
<tr>
<td>d ↔ c</td>
<td>score_2</td>
</tr>
<tr>
<td>e ↔ f</td>
<td>score_3</td>
</tr>
<tr>
<td>g ↔ b</td>
<td>score_4</td>
</tr>
<tr>
<td>h ↔ i</td>
<td>score_4</td>
</tr>
</tbody>
</table>
while unprocessed hypotheses with no tied competitors remain do
  while the highest-scoring hypothesis has tied competitors do
    skip all tied competitors
  end while
  link the highest-scoring hypothesis
  discard all incompatible hypotheses
end while
while unprocessed hypotheses with no tied competitors remain do
if the highest-scoring hypothesis has tied competitors do
mark the constituents of all tied competitors
end if
while the highest-scoring hypothesis has a marked constituent do
skip
end while
link the highest-scoring hypothesis
discard all incompatible hypotheses
unmark all constituents
end while

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>a ⇔ b</td>
<td>score_1</td>
<td></td>
</tr>
<tr>
<td>a ⇔ c</td>
<td>score_1</td>
<td></td>
</tr>
<tr>
<td>d ⇔ c</td>
<td>score_2</td>
<td></td>
</tr>
<tr>
<td>e ⇔ f</td>
<td>score_3</td>
<td></td>
</tr>
<tr>
<td>g ⇔ b</td>
<td>score_4</td>
<td></td>
</tr>
<tr>
<td>h ⇔ i</td>
<td>score_4</td>
<td></td>
</tr>
</tbody>
</table>
while unprocessed hypotheses with no tied competitors remain do
if the highest-scoring hypothesis has tied competitors do
mark the constituents of all tied competitors
end if
while the highest-scoring hypothesis has a marked constituent do
skip
end while
link the highest-scoring hypothesis
discard all incompatible hypotheses
unmark all constituents
end while

| a ↔ b   | score_1 |
| a ↔ c   | score_1 |
| d ↔ c   | score_2 |
| e ↔ f   | score_3 |
| g ↔ b   | score_4 |
| h ↔ i   | score_4 |
while unprocessed hypotheses with no tied competitors remain do
if the highest-scoring hypothesis has tied competitors do
mark the constituents of all tied competitors
end if
while the highest-scoring hypothesis has a marked constituent do
skip
end while
link the highest-scoring hypothesis
discard all incompatible hypotheses
unmark all constituents
end while
while unprocessed hypotheses with no tied competitors remain do
if the highest-scoring hypothesis has tied competitors do
mark the constituents of all tied competitors
end if
while the highest-scoring hypothesis has a marked constituent do
skip
end while
link the highest-scoring hypothesis
discard all incompatible hypotheses
unmark all constituents
end while

'a ⇔ b' is incompatible with 'e ⇔ f'

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b</td>
<td>score_1</td>
</tr>
<tr>
<td>a</td>
<td>c</td>
<td>score_1</td>
</tr>
<tr>
<td>d</td>
<td>c</td>
<td>score_2</td>
</tr>
<tr>
<td>e</td>
<td>f</td>
<td>score_3</td>
</tr>
<tr>
<td>g</td>
<td>b</td>
<td>score_4</td>
</tr>
<tr>
<td>h</td>
<td>i</td>
<td>score_4</td>
</tr>
</tbody>
</table>
while unprocessed hypotheses with no tied competitors remain do
if the highest-scoring hypothesis has tied competitors do
mark the constituents of all tied competitors
end if
while the highest-scoring hypothesis has a marked constituent do
skip
end while
link the highest-scoring hypothesis
discard all incompatible hypotheses
unmark all constituents
end while
span1
• High-scoring improper lexical links prevent the production of good non-lexical links

• Split the set of nonzero links into two sets:
  • lexical links
  • non-lexical links
High-scoring improper lexical links prevent the production of good non-lexical links.

Split the set of nonzero links into two sets:
- lexical links
- non-lexical links

\[
\begin{array}{c|c}
A & W \\
B & X \\
B & Z \\
C & Y \\
BC & XY \\
BC & Z \\
BC & WZ \\
AB & WZ \\
\end{array}
\]
High "scoring improper lexical links prevent the production of good non"lexical links

Split the set of nonzero links into two sets:

- lexical links
- non"lexical links

\[
\begin{array}{c|c|c|c|c|c|c}
\text{AB} & \text{BC} & \text{WZ} & \text{XY} & \text{W} & \text{X} & \text{Y} \\
\text{A} & \text{B} & \text{C} & \text{W} & \text{X} & \text{Y} & \text{Z} \\
a & b & c & w & x & y & z \\
\end{array}
\]
• High-scoring improper lexical links prevent the production of good non-lexical links

• Split the set of nonzero links into two sets:
  • lexical links
  • non-lexical links

• Perform selection
  • first amongst the non-lexical links
  • then amongst the lexical links
Full-Search Selection
Full-Search Selection

- Backtracking Recursive Algorithm
  - enumerate all possible combinations of non-crossing links
  - store all maximal combinations of non-crossing links
- The best combination of links is the highest-scoring maximal combination of links
- Ambiguity again...
String-to-String Alignment
String-to-String Alignment

- The sub-tree aligner operates on parsed data
- For many languages no parsers are available
  - Retraining existing parsers for new languages may require significant resources
- The string-to-string aligner operates on plain sentences
String-to-String Alignment

algorithm

- Generate all possible binary trees for each sentence in the sentence pair
- Calculate scores for all possible links
- Select the best set of links as before
  - Two links are incompatible if they contain nodes that are part of incompatible trees
String-to-String Alignment Algorithm
String-to-String Alignment algorithm

Two links are incompatible if they contain nodes that are part of incompatible trees.
String-to-String Alignment algorithm

Two links are incompatible if they contain nodes that are part of incompatible trees.
String-to-String Alignment

algorithm

Diagram:

- $X_1$
- $X_2$
- $X_3$
- $X_4$
- $X_e$
- $X_f$

Links between nodes indicate alignment.

Steps:

1. Generate all possible binary trees for each sentence in the sentence pair.
2. Calculate scores for all possible links.
3. Select the best set of links as before.
4. Two links are incompatible if they contain nodes that are part of incompatible trees.
String-to-String Alignment

Algorithm

- Generate all possible binary trees for each sentence in the sentence pair
- Calculate scores for all possible links
- Select the best set of links as before
- Two links are incompatible if they contain nodes that are part of incompatible trees
String-to-String Alignment algorithm

- Generate all possible binary trees for each sentence in the sentence pair
- Calculate scores for all possible links
- Select the best set of links as before
- Two links are incompatible if they contain nodes that are part of incompatible trees
String-to-String Alignment algorithm

- Generate all possible binary trees for each sentence in the sentence pair
- Calculate scores for all possible links
- Select the best set of links as before
  - Two links are incompatible if they contain nodes that are part of incompatible trees
- Only output linked nodes
  - and nodes needed for structural integrity
String-to-String Alignment

- Tree-to-string and string-to-tree modules
  - used when a parser exists for one of the languages in question
  - based on the string-to-string module
  - preserve original parse structures
Re-Scoring Algorithm
• Use already fixed links to establish the context for word alignments
Re-Scoring Algorithm
Re-Scoring Algorithm
Re-Scoring Algorithm
Re-Scoring Algorithm
Re-Scoring Algorithm
Algorithm Complexity
Algorithm Complexity

- Tree-to-Tree Alignment
  - space: $O(n)$
  - time: $O(m^2)$

- String-to-String Alignment
  - space: $O(n^2)$
  - time: $O(m^4)$

- Full-Search Algorithm
  - related to the TSP, albeit highly restricted
  - Score calculation may be executed in parallel
Contents of the Distribution
Contents of the Distribution

• Available at
  http://ventsislavzhechev.eu/Home/Software/Software.html

• an RSS feed is available for update notifications
  http://ventsislavzhechev.eu/Home/Software/rss.xml

• GPL licence

• README file

• C++ code of the sub-tree aligner

• Configuration and compilation scripts
Required Software
Required Software

- GCC 4.0+
  - GCC 4.2+ required for the compilation of parallel code
- boost 1.34+ for string-to-string, tree-to-string and string-to-tree modules
- Should compile on any UNIX system
Avenues for Improvement
Avenues for Improvement

• Add new scoring algorithms (eg. maximum-entropy-based)
• Research ways to reduce the number of scores that need to be calculated
• Store word-alignment data in memory in a more optimal way
• Research ways to integrate POS-tag data into the structures generated by the string-based modules
• Make the main features selectable at run time, rather than at compile time
Conclusions
Conclusions

- Developed a novel platform for the fast and robust generation of parallel treebanks
- The aligner can handle very large amounts of data
- There are many underresourced languages
  - String-to-string, string-to-tree and tree-to-string algorithms have been developed
- Please send your bug reports to bugs@ventsislavzhechev.eu
Thank you!