DAESO Corpus and Tools

Erwin Marsi, Emiel Krahmer

Universiteit van Tilburg

June 10, 2009
Outline

DAESO Corpus

Evaluation

Automatic alignment

Briefly mentioned

Plan
Step 1: conversion to XML

1. Autocue-subtitle
   - pairs from NOS and VRT Journals
   - converted from HTML to ad hoc XML

2. Books
   - parts of two alternative translations of:
     2.1 Darwin - Origin of Species
     2.2 Montaigne - Essays
     2.3 Saint-Exupery - Le Petit Prince
   - converted from PDF, Doc, OCR to TEI Lite

3. Headlines
   - headlines from similar news articles from Google News
   - converted to ad hoc XML
Step 1: conversion to XML (cont’d)

1. News
   ▶ sentences from press releases by ANP and Novum about the same news event
   ▶ obtained from news feeds
   ▶ converted from HTML to ad hoc XML

2. QA
   ▶ alternative answers to the same question in a QA context
   ▶ obtained from IMIX QA reference corpus
   ▶ modified from original XML
Step 2: Tokenization

- tokenize texts with DCOI tokenizer
- add <s id="..."> ... </s> tags
Step 3: Sentence alignment

- align similar sentences
- many-to-many alignments possible
- partly automatic, followed by manual correction
- using Hitaext, a general tool for manually aligning XML documents
- store alignment in *parallel text corpus files* (.ptc)
Hitaext: text alignment tool

Text view

Document tree view
Parallel Text Corps XML format

```xml
<?xml version="1.0" encoding="utf-8"?>
<hitaext>
  <from>
    <render>...
    </render>
  </from>
  <to>
    <render>...
    </render>
  </to>
  <alignment method="id">
    <link from_id="r10" from_tag="release" to_id="r10" to_tag="release" />
    <link from_id="s10" from_tag="s" to_id="s10" to_tag="s" />
    <link from_id="s30" from_tag="s" to_id="s30" to_tag="s" />
    ...
    <link from_id="r20" from_tag="release" to_id="r20" to_tag="release" />
    <link from_id="s90" from_tag="s" to_id="s180" to_tag="s" />
    <link from_id="s110" from_tag="s" to_id="s200" to_tag="s" />
    <link from_id="s120" from_tag="s" to_id="s210" to_tag="s" />
    ...
  </alignment>
</hitaext>
```
Step 4: Parsing

- parse all sentences with Alpino parser
- collect parse trees in graph bank files
Graph Bank XML format

<?xml version="1.0" encoding="utf-8"?>
<treebank>
  <alpino_ds id="s10" version="1.2">
    <node begin="0" cat="top" end="7" id="0" rel="top">
      <node begin="0" cat="smain" end="7" id="1" rel="--">
        <node begin="0" end="1" id="2" pos="name" rel="su" root="Zeeland" word="Zeeland"/>
        <node begin="1" end="2" id="3" pos="verb" rel="hd" root="verwijderen" word="verwijdert"/>
        <node begin="2" cat="np" end="7" id="4" rel="obj1">
          <node begin="2" end="3" id="5" pos="noun" rel="hd" root="poster" word="posters"/>
          <node begin="3" cat="mwu" end="7" id="6" rel="app">
            <node begin="3" end="4" id="7" pos="name" rel="mwp" root="Partij" word="Partij"/>
            <node begin="4" end="5" id="8" pos="name" rel="mwp" root="voor" word="voor"/>
            <node begin="5" end="6" id="9" pos="name" rel="mwp" root="de" word="de"/>
            <node begin="6" end="7" id="10" pos="name" rel="mwp" root="Dieren" word="Dieren"/>
          </node>
        </node>
      </node>
      <sentence>Zeeland verwijdert posters Partij voor de Dieren</sentence>
      <comments>
        <comment>Q#1|Zeeland verwijdert posters Partij voor de Dieren|1|1|-0.02610407259382212</comment>
      </comments>
    </node>
  </alpino_ds>
  <alpino_ds id="s20" version="1.2">
  ...
  </alpino_ds>
</treebank>
Step 5: Tree alignment

- align similar nodes from a pair of parse trees
- label edges according to a set of five similarity relations
- partly automatic, with manual correction
- using Algraeph, a tool for manual graph alignment
- store alignment in *parallel graph corpus files* (.ptc)
Algraeph: graph alignment tool
Parallel Graph Corpus XML format

<?xml version="1.0" encoding="utf-8"?>
<parallel_graph_corpus>
  <comment>Annotator: HS</comment>
  <graphbanks format="alpino">
    <file format="alpino" id="1">../../..//gb/2006-11/news-novum-2006-11-part-00-treebank.xml</file>
    <file format="alpino" id="2">../../..//gb/2006-11/news-anp-2006-11-part-00-treebank.xml</file>
  </graphbanks>
  <node_relations>
    <relation>equals</relation>
    <relation>restates</relation>
    <relation>specifies</relation>
    <relation>generalizes</relation>
    <relation>intersects</relation>
  </node_relations>
  <aligned_graphs>
    <graph_pair from_bank_id="1" from_graph_id="s10" to_bank_id="2" to_graph_id="s10">
      <aligned_nodes>
        <node_pair from_node_id="5" relation="equals" to_node_id="8" />
        <node_pair from_node_id="2" relation="equals" to_node_id="5" />
        <node_pair from_node_id="3" relation="equals" to_node_id="6" />
        ...
      </aligned_nodes>
    </graph_pair>
    <graph_pair from_bank_id="1" from_graph_id="s30" to_bank_id="2" to_graph_id="s30">
      ...
    </graph_pair>
    ...
  </aligned_graphs>
</parallel_graph_corpus>
How to count alignments?

- There are many ways to count the number of aligned words ...
- Our counting method is rather strict
- We essentially count the number *words* in *fully aligned graphs*, discounting
  - punctuation tokens
  - unaligned sentences
  - failed parses
  - incompletely aligned graphs
## Alignment counts

<table>
<thead>
<tr>
<th>Source</th>
<th>#Graphpairs</th>
<th>#Nodepairs</th>
<th>#Tokens</th>
</tr>
</thead>
<tbody>
<tr>
<td>Autocue-subtitle</td>
<td>9 851</td>
<td>135 798</td>
<td>217 956</td>
</tr>
<tr>
<td>Books</td>
<td>3 430</td>
<td>63 874</td>
<td>114 485</td>
</tr>
<tr>
<td>Headlines</td>
<td>13 084</td>
<td>89 086</td>
<td>97 681</td>
</tr>
<tr>
<td>News</td>
<td>8 248</td>
<td>86 227</td>
<td>162 361</td>
</tr>
<tr>
<td>QA</td>
<td>186</td>
<td>1 503</td>
<td>2 230</td>
</tr>
<tr>
<td><strong>Totals:</strong></td>
<td><strong>34 799</strong></td>
<td><strong>376 488</strong></td>
<td><strong>594 713</strong></td>
</tr>
</tbody>
</table>
# Distribution of relations per corpus segment

<table>
<thead>
<tr>
<th>Segment</th>
<th>Eq</th>
<th>Re</th>
<th>Spec</th>
<th>Gen</th>
<th>Inter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Autocue-subtitle</td>
<td>67.46</td>
<td>11.48</td>
<td>2.58</td>
<td>14.12</td>
<td>4.37</td>
</tr>
<tr>
<td>Books</td>
<td>57.17</td>
<td>21.87</td>
<td>3.82</td>
<td>4.31</td>
<td>12.84</td>
</tr>
<tr>
<td>Headlines</td>
<td>54.56</td>
<td>11.03</td>
<td>9.48</td>
<td>10.43</td>
<td>14.49</td>
</tr>
<tr>
<td>News</td>
<td>55.59</td>
<td>8.32</td>
<td>7.58</td>
<td>7.05</td>
<td>21.46</td>
</tr>
<tr>
<td>QA</td>
<td>59.28</td>
<td>6.05</td>
<td>5.59</td>
<td>4.79</td>
<td>24.28</td>
</tr>
<tr>
<td>Overall:</td>
<td>58.89</td>
<td>12.13</td>
<td>6.33</td>
<td>10.02</td>
<td>12.64</td>
</tr>
</tbody>
</table>
Final result

- corpus in XML format (marked-up source text, graphbanks, parallel text corpus, parallel graph corpus files)
- annotation tools for manual text and graph alignment
- software library and command line scripts for using the corpus
- annotation manual
- corpus documentation
- corpus statistics
Goal of the evaluation is to determine the consistency of the annotation.

We measured the inter-annotator agreement, for both node alignment and relation labeling.

On 4 different text sources.

Using 6 different annotators.
## Evaluation samples

<table>
<thead>
<tr>
<th>Source</th>
<th>#Graphpairs</th>
<th>#Words</th>
<th>#Nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Autocue-subtitle</td>
<td>50</td>
<td>1537</td>
<td>2598</td>
</tr>
<tr>
<td>Montaigne</td>
<td>25</td>
<td>1352</td>
<td>1528</td>
</tr>
<tr>
<td>News</td>
<td>48</td>
<td>2674</td>
<td>3952</td>
</tr>
<tr>
<td>Saint</td>
<td>51</td>
<td>973</td>
<td>1591</td>
</tr>
</tbody>
</table>
Precision, recall and F-score on alignment

True alignment

Predicted alignment

\[ \text{Precision} = \frac{|\text{True} \cap \text{Pred}|}{|\text{Pred}|} = \frac{|\{<a,v>\}|}{|\{<a,v>,<c,x>\}|} = \frac{1}{2} = 0.5 \]
Precision, recall and F-score on alignment

**True alignment**

**Predicted alignment**

\[
\text{Precision} = \frac{|\text{True} \cap \text{Pred}|}{|\text{Pred}|} = \frac{|\{<a,v>, <c,x>\}|}{|\{<a,v>, <c,x>\}|} = \frac{1}{2} = 0.5
\]

\[
\text{Recall} = \frac{|\text{True} \cap \text{Pred}|}{|\text{True}|} = \frac{|\{<a,v>\}|}{|\{<a,v>, <b,x>, <c,z>\}|} = \frac{1}{3} = 0.33
\]
Precision, recall and F-score on alignment

True alignment

![True alignment diagram]

Predicted alignment

![Predicted alignment diagram]

- **Precision** = \( \frac{|\text{True} \cap \text{Pred}|}{|\text{Pred}|} = \frac{|\{<a,v>,<c,x>\}|}{|\{<a,v>,<c,x>\}|} = \frac{1}{2} = 0.5 \)

- **Recall** = \( \frac{|\text{True} \cap \text{Pred}|}{|\text{True}|} = \frac{|\{<a,v>\}|}{|\{<a,v>,<b,x>,<c,z>\}|} = \frac{1}{3} = 0.33 \)

- **F₁ score** = \( \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} = \frac{2 \times 1/2 \times 1/3}{1/2 + 1/3} = \frac{2}{5} = 0.4 \)
Cross-evaluation

- Given annotations $A_1, \ldots, A_n$:
  - for each $A_i$ evaluate against the other $n - 1$ annotations
  - and calculate the average precision, recall and F-score.
Precision, recall and F-score on relation labeling

True alignment

Predicted alignment

To calculate scores on relation R:
1. restrict to alignments labeled with relation R
2. calculate precision, recall and F-score as before

Repeat for each relation R
Perform cross-evaluation as before
Interannotator agreement on alignment only

Autocue-subtitle sample

Montaigne sample

News sample

Saint-Exupery sample

Scores

Autocue-subtitle sample

Montaigne sample

News sample

Saint-Exupery sample

Scores

Scores

Scores

Scores
Interannotator agreement on *Equals* relation

Autocue-subtitle sample

Montaigne sample

News sample

Saint-Exupery sample
Interannotator agreement on *Restates* relation

![Autocue-subtitle sample](chart1)

![Montaigne sample](chart2)

![News sample](chart3)

![Saint-Exupery sample](chart4)
Interannotator agreement on *Specifies* relation

**Autocue-subtitle sample**

- **Scores** (Precision, Recall, F-score)
- **BB**, **FvD**, **HS**, **KvL**, **NE**, **VN**

**Montaigne sample**

- **Scores** (Precision, Recall, F-score)
- **BB**, **FvD**, **HS**, **KvL**, **NE**, **VN**

**News sample**

- **Scores** (Precision, Recall, F-score)
- **BB**, **FvD**, **HS**, **KvL**, **NE**, **VN**

**Saint-Exupery sample**

- **Scores** (Precision, Recall, F-score)
- **BB**, **FvD**, **HS**, **KvL**, **NE**, **VN**
Interannotator agreement on *Generalizes* relation

![Graphs showing precision, recall, and F-score for different samples: Autocue-subtitle, Montaigne, News, Saint-Exupery. Each graph compares different annotators (BB, FvD, HS, KvL, NE, VN).]
Interannotator agreement on *Intersects* relation

![Graphs showing precision, recall, and F-score for various samples.](image-url)
Automatic alignment of syntax trees

- Tree alignment is a hard problem
  - knowledge from many sources comes into play
  - corpus data is noisy
- Our initial approach using DP-based tree alignment algorithms did not work very well
  - restrictions on matching are too strict
  - difficult to combine knowledge sources
- Hence matching needs to be more relaxed (error-tolerant graph matching)
- Further attempts suffered from
  - lack of integration between alignment and relation labeling
  - knowledge-based hacks per relation/domain
  - need for manual tuning of somewhat arbitrary weights
Recently we have developed a new model for graph alignment

Casts alignment and labeling as a combination of general classification and resource optimization
Pipeline for automatic tree alignment

1. linguistic processing
2. feature extraction
3. relation classification
4. weighting
5. matching
6. merging
Linguistic processing

- Tokenization
- Syntactic parsing (Alpino)
- Only required in the sentence alignment module - not for corpus alignment
Feature extraction

- for all possible pairings of a node $n$ in the source tree to a node $m$ in the target tree, extract features

- a general system of feature extraction functions makes it easy to extract all sorts of shallow and deep features

  - string identity
  - substring
  - Levenshtein distance
  - same root
  - POS / category
  - parent node has same category
  - overlap in dependency triples
  - lexical semantic relations (e.g. synonym, hyponym, hyperonym) from Cornetto
  - matching digits to numerical expressions
  - ...
Relation classification

- for all possible pairings of a node n in the source tree to a node m in the target tree, use a generic machine learner to predict the alignment relation (possibly none)
- currently we are using Timbl (IB1)
Weighting

- assign a weight (akin to a confidence score) to each relation prediction
- currently we use the normalized entropy of the class labels in the set of nearest neighbours (but there are other options)
Matching

- given $n \times m$ possible alignments and associated weights, we want to find a matching (i.e. only one-to-one alignments) with maximum weight
- this is well-known problem in combinatorial optimization known as the Assignment Problem
- in graph-theoretical terms: find a maximum weighted bipartite graph matching
- can be solved in polynomial time using e.g. the Hungarian algorithm ($O(n^3)$)
- finally, filter alignments with weight below a certain threshold
Merging

- merge an empty parallel graph corpus (i.e. only graph alignments, no node alignments) with the node matching to generate an aligned parallel graph corpus
- only required from corpus alignment - not in the sentence alignment module
Final result

- tree alignment script for aligning parallel graph corpus files
- stand-alone tree alignment module/server for aligning a pair of sentences
- framework of library and scripts for building, modifying, testing aligners
- documentation
Pycornetto

- Pycornetto is a Python interface to the Cornetto database
- There are basically four ways to use Pycornetto:
  - interactively from the command line of the Python interpreter
  - from your own Python code
  - run the database server and connect to it with the supplied client
  - run the database server and connect to it through XML-RPC
- Option to add word counts and a number of corpus-based word similarity measures (Resnik, Jiang & Conrath, Lin)
- Online demo
Near future plan

- finish tree aligner
- automatic alignment of second half of corpus
- deliver DAESO corpus
- publish papers about corpus and aligner
- continue work on sentence fusion