Analysing Temporally Annotated Corpora with CAVaT

The University Of Sheffield.

Temporal Annotation

What to annotate?

- Events and time expressions (intervals)
- Temporal, aspectual and subordinate links between intervals
- Signals that indicate recurrence or temporal ordering

TimeML is a formal specification for annotating these kinds of entity

Surveying temporally annotated corpora

To learn about how time is expressed in English, we can examine a temporally annotated corpus – TimeBank

At a high level, we can gather TimeML tag and attribute frequency distributions

Basic tools to do this quickly - sed/awk/grep

These quickly become awkward and prone to human error

Surveying temporally annotated corpora

To see the distribution of PoS type in a TimeML corpus:

```
$ grep -h MAKEI ~/corpora/timebank_1_2/data/timeml/* | sed
's/>/>\n/g' | grep MAKEI | sed 's/ /\n/g' | grep pos= | sed
's/\/>//' | sort | uniq -c
```

```
 266 pos="ADJECTIVE"
2225 pos="NOUN"
 299 pos="OTHER"
  28 pos="PREPOSITION"
5122 pos="VERB"
```

Surveying temporally annotated corpora

Solution: Parse TimeML corpus and load it into a database

Query the database with SQL

- Much easier than complex command lines!

Often we want specially-formatted output (LaTeX tables)

There is a core set of queries that are very similar and are used very frequently

The University Of Sheffield.

The CAVaT prompt

`cavat>`

Readline-based interface

Flexible PyParsing grammar

Query specification is kept close to natural-language

Online help available for most commands

Multiple output formats:

- LaTeX
- CSV
- screen

The University Of Sheffield.

The CAVaT prompt

To see the distribution of PoS type in a TimeML corpus:

```
# CAVaT Corpus Analysis and Validation for TimeML
# Version:  0.2   Support:  leon@dcs.shef.ac.uk
cavat> show distribution of event pos
  Frequency    EVENT pos     Proportion
 ========================================
       5122    VERB             64.5%
       2225    NOUN             28.0%
        299    OTHER            3.77%
        266    ADJECTIVE        3.35%
         28    PREPOSITION      0.353%
       7940    Total
```

If we're writing a report, we can just enter:

```
cavat> show distribution of event pos as latex
```

The University Of Sheffield.

Problems in existing TimeML corpora

In earlier research, we found some issues with TimeBank:

- Building a TLINK relType classifier, we found events that were ordered as being after themselves (odd).
- TLINKs introduced via temporal closure sometimes conflicted with initial annotations

To help TimeML annotators spot these errors in future, CAVaT includes validation checks.

CAVaT checks are modular and can be authored with only knowledge of the database schema.

The University Of Sheffield.

Looped TLINK check

When both arguments of a TLINK reference the same event instance.

From wsj_0586.tml:

```
<TLINK lid="l192" relType="BEFORE"
eventInstanceID="ei1404"
relatedToEventInstance="ei1404"/>
```

For simultaneous or identity relations, the link is redundant;

If the relation type is anything else, it is erroneous.

Give a warning if a TLINK links two different instances of the same event

The University Of Sheffield.

Looped TLINK check

Example output:

```
cavat> check tlink_loop in all
# TLINK loop checker v1 loaded
# Checking wsj_0736.tml (id 4)
TLINK ID l48 loops directly (instanceID match), type
IDENTITY, event ei316 / ei316
# Checking APW19980227.0494.tml (id 38)
TLINK ID l61 loops directly (instanceID match), type AFTER,
event ei2338 / ei2338

# Checking ABC19980304.1830.1636.tml (id 165)
TLINK ID l23 may be a loop (eventID match), type INCLUDES,
event ei286 / ei288 - check document manually
```

The University Of Sheffield.

Consistency check

The temporal links, events and timexes in a document can be though of edges and nodes in a temporal graph.

Not all temporal graphs are consistent; for example,

- **A** `before` **B**
- **B** `before` **C**
- **C** `before` **A**

CAVaT includes an agenda-based temporal consistency check, which first converts intervals to pairs of points, and then attempts to perform a closure.

Conflicting information indicates an inconsistency.

Consistency check

```
cavat> check consistent in all
# Temporal graph consistency checker v1 loaded
# Checking wsj_0927.tml (id 3)
! Inconsistent closure - could not assert (ei2415_2 <
ei2414_1)
# Checking wsj_0778.tml (id 7)
! Inconsistent closure - could not assert (ei2090_2 <
ei1988_2)
# Checking wsj_0762.tml (id 10)
! Inconsistent closure - could not assert (ei2005_2 <
ei2003_2)
...
```

The University Of Sheffield.

Advanced uses

Orphan detection finds elements not linked to the rest of the document.

- Helps avoid incomplete annotation
- Events or timexes not linked at all
- Uninstantiated events
- Instances of non-existent events
- Signals not referenced by a TLINK or event instance

The University Of Sheffield.

Advanced uses

Subgraph identification:

- A document may have multiple independent temporal subgraphs
- Identify subgraph sizes and count
- Report on how well a document is linked overall with an entropy-based measure

The University Of Sheffield.

Advanced uses

```
cavat> check split_graph in wsj_0132.tml
# Split graph detection v1 loaded
# Checking wsj_0132.tml (id 1)
Subgraphs found: 3 - composed of 28 nodes and linked by 25
TLINKS.
Isolated subgraphs, that contain just one TLINK: 2
(making up 66.7% of all subgraphs / consuming 14.3% of all
nodes / described by 8.0% of all TLINKs);
Mean graph size 9.3 nodes; largest subgraph (size 24) has
85.7% of all nodes.
Entropy of subgraph sizes:  0.15279294609
    2 nodes: ( 2) ..
   24 nodes: ( 1) .
```

TLINK loops manifest as single-node subgraphs

The University Of Sheffield.

Future work

Extensions to query syntax

- “save” keyword to dump results to a file
- “signalled” keyword for elements that link to a signal

Extra checks

- S2T verification
- Ensure SLINKs only interact with certain event classes

Feature generation for classifier training

The University Of Sheffield.

Open project

Available via SVN at:

http://code.google.com/p/cavat/

Extensible check module architecture; an example class is included

Thank you! Any questions?