Structural Metadata at CUED: Progress Report

Marcus Tomalin, Sue Tranter, Phil Woodland, & the CUED STT Team (including Ji-Hwan Kim)

May 21st 2003

Cambridge University Engineering Department
Progress: Jan 2003 - May 2003

- Specific CUED Structural Metadata Research:
  - Main focus on Slash Unit (SU) detection/classification.
  - Prosodic Feature Model (PFM).
  - SU Language Model (SULM).
  - SU Decoder.

- General MACEARS Structural Metadata Tasks:
  - Involved in SimpleMDE Annotation spec discussions.
  - Involved in the SimpleMDE pilot annotation.
  - Involved in the tool testing process.
Where were we in Jan 2003?

The CUED RT-03 dryrun SU system used word-time information and token-spotting algorithms:

\[ N : \text{gap of } N \text{ seconds in transcriptions} \rightarrow \text{SU} \]
\[ \text{SENT: SENT\_START or SENT\_END tag in STT output} \rightarrow \text{SU} \]

\[ \text{QUES} = \{ \text{WHAT WHY WHERE WHEN HOW DO ARE IS HAVE DID HAS REALLY} \} \]
\[ \text{CO-CONJ} = \{ \text{AND BUT OR} \} \]
\[ \text{SUB-CONJ} = \{ \text{IF HOWEVER THEREFORE} \} \]
\[ \text{ART} = \{ \text{THE A AN} \} \]
\[ \text{QUANT} = \{ \text{ANY ALL MOST EVERY} \} \]
\[ \text{INCOMP} = \{ \text{CO-CONJ $SUB$-CONJ $ART$ QUANT} \} \]

RULE1: \( \text{su} = \text{question if ( su-initial word == QUES )} \)
RULE2: \( \text{su} = \text{incomplete if ( su-final word == INCOMP )} \)
RULE3: \( \text{su} = \text{backchannel if ( su == BC+ )} \)
RULE4: \( \text{su} = \text{statement if (su not already classified) } \)
Training and Test Data

Data Sets:

- Training data: subset of Treebank3 (TB3) corpus (c.90 hours).
- ‘Held out’ data: subset of TB3 corpus (c.1 hour).
- Test data: RT-03 dry-run test set.

Some problems with this:

- The training and test data sets are not annotated in exactly the same way - but we needed training data!
- Backchannels not labelled separately in the training data.
- Only the test data has reference ctm/mdtm files - so system tuning has to be performed upon the test data.
SU System Overview

Figure 3: SU Detection System
The Prosodic Feature Model

The Prosodic Features (PFs):

<table>
<thead>
<tr>
<th>Prosodic Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pause.Length</td>
<td>the pause length at the end of the word</td>
</tr>
<tr>
<td>Duration</td>
<td>the duration from the previous pause</td>
</tr>
<tr>
<td>Avg_F0_L</td>
<td>the mean of the good F0 values in left window</td>
</tr>
<tr>
<td>Avg_F0_R</td>
<td>the mean of the good F0 values in right window</td>
</tr>
<tr>
<td>Avg_F0_ratio</td>
<td>Avg_F0_L / Avg_F0_R</td>
</tr>
<tr>
<td>Cnt_F0_L</td>
<td>the number of good F0s in left window</td>
</tr>
<tr>
<td>Cnt_F0_R</td>
<td>the number of good F0s in right window</td>
</tr>
<tr>
<td>Eng_L</td>
<td>the RMS energy in left window</td>
</tr>
<tr>
<td>Eng_R</td>
<td>the RMS energy in right window</td>
</tr>
<tr>
<td>Eng_ratio</td>
<td>Eng_L / Eng_R</td>
</tr>
</tbody>
</table>

[Following Shriberg et al. 1998, Kim 2001]
The Prosodic Feature Model

4 SU types defined:

- **SU_S**: statement SU boundary
- **SU_Q**: question SU boundary
- **SU_I**: incomplete SU boundary
- **SU_N**: no SU-boundary

Steps in the PFM construction process:

- Convert training data into word sequences.
- Classify each word into one of the above SU sub-types.
- Obtain Forced Alignments for training data word sequences.
- Extract PF info using word start/end times.
- Construct CART decision tree using PFs and SU sub-type classification.
The Prosodic Feature Model

1456 Nodes: (728 non-terminal + 729 terminal)

Measures for determining the contribution of the PFs:

- **Feature Appearance:**
  the number of times a feature is used as a classifying feature.

- **Feature Usage:**
  the proportion of the number of times a feature is queried.
The Prosodic Feature Model

<table>
<thead>
<tr>
<th>Prosodic Feature</th>
<th>Feature Appearance</th>
<th>Feature Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pause_Length</td>
<td>180</td>
<td>0.615</td>
</tr>
<tr>
<td>Duration</td>
<td>115</td>
<td>0.094</td>
</tr>
<tr>
<td>Avg_F0_L</td>
<td>67</td>
<td>0.001</td>
</tr>
<tr>
<td>Avg_F0_R</td>
<td>62</td>
<td>0.014</td>
</tr>
<tr>
<td>Avg_F0_ratio</td>
<td>52</td>
<td>0.018</td>
</tr>
<tr>
<td>Cnt_F0_L</td>
<td>36</td>
<td>0.066</td>
</tr>
<tr>
<td>Cnt_F0_R</td>
<td>29</td>
<td>0.018</td>
</tr>
<tr>
<td>Eng_L</td>
<td>63</td>
<td>0.033</td>
</tr>
<tr>
<td>Eng_R</td>
<td>70</td>
<td>0.116</td>
</tr>
<tr>
<td>Eng_ratio</td>
<td>54</td>
<td>0.003</td>
</tr>
</tbody>
</table>

Table 1: Prosodic Feature Usage
The SU Language Model

Training Data Preparation:
Insert the required SU token after every word in the training data:
Example:

\[
\langle s \rangle \text{ OKAY SU_S ARE SU_N WE SU_N READY SU_Q I SU_N THINK SU_N WE SU_N SHOULD SU_N GIVE SU_I OKAY SU_S} \ldots \langle /s \rangle
\]

Number of words in training data: 348,231
Three kinds of SULM were constructed:

- N-gram SULM
- Class-based SULM
- Interpolated N-gram + class-based SULM
The SU Language Model

<table>
<thead>
<tr>
<th>SULM Type</th>
<th>Perplexity</th>
<th>Classes</th>
<th>Interpolation Weights</th>
</tr>
</thead>
<tbody>
<tr>
<td>bg</td>
<td>29.1</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>tg</td>
<td>21.2</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>40cl-bg</td>
<td>28.3</td>
<td>40</td>
<td>N/A</td>
</tr>
<tr>
<td>40cl-tg</td>
<td>31.4</td>
<td>40</td>
<td>N/A</td>
</tr>
<tr>
<td>bg + 40cl-bg</td>
<td>27.7</td>
<td>40</td>
<td>~0.3, ~0.7</td>
</tr>
<tr>
<td>tg + 40cl-tg</td>
<td>20.8</td>
<td>40</td>
<td>~0.9, ~0.1</td>
</tr>
<tr>
<td>bg + 40cl-tg</td>
<td>28.3</td>
<td>40</td>
<td>~0.7, ~0.3</td>
</tr>
</tbody>
</table>

Table 2: The SULMs

- Training data used to build SULMs.
- ‘Held out’ data used to obtain Perplexity (PP) values.
- The PPs are low (compared to typical STT perplexities) because the probability of the inter-word SU tokens is high.
The SU Decoder

The basic method used to combine the PFM and the SULM:

• Obtain STT output for test data.

• Obtain PFM scores (for the 4 SU sub-types) for each word in STT output.

• Create initial lattices using PFM scores and STT test data word sequences.

• Expand the initial lattices, using the SULM and standard lattice tools, to create a network.

• Select the best path (i.e., highest prob) through the expanded lattice.

• Output word and SU token sequence corresponding to the best path.

• Identify Backchannels in post-processing stage (token-spotting).
The SU Decoder

PFM scores are added to the arcs of the initial lattice:

Figure 3: Initial SU Decoder lattice
The SU Decoder

The Grammar Scale Factor (GSF) constant weights the PFM and SULM scores:

\[ \log \text{PFM\_score} + (\text{GSF} \times \log \text{SULM\_score}) \]

The GSF can be varied (NB: this is tuning on the test data!)

Figure 2: SU Error for Different Grammar Scale Factors

\[ SU \text{ Error for GSF Values} \]
## SU Results

<table>
<thead>
<tr>
<th>System</th>
<th>GSF</th>
<th>%Del</th>
<th>%Ins</th>
<th>%Sub</th>
<th>%Err</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUED Dryrun*</td>
<td>N/A</td>
<td>32.08</td>
<td>31.67</td>
<td>21.59</td>
<td>85.34</td>
</tr>
<tr>
<td>PFM</td>
<td>N/A</td>
<td>24.88</td>
<td>43.98</td>
<td>54.61</td>
<td>123.47</td>
</tr>
<tr>
<td>SULM_bg</td>
<td>N/A</td>
<td>81.30</td>
<td>6.32</td>
<td>3.56</td>
<td>91.19</td>
</tr>
<tr>
<td>SULM_40cl-tg</td>
<td>N/A</td>
<td>84.47</td>
<td>6.28</td>
<td>3.86</td>
<td>94.61</td>
</tr>
<tr>
<td>SULM_bg+40cl-tg</td>
<td>N/A</td>
<td>86.35</td>
<td>4.51</td>
<td>2.96</td>
<td>93.81</td>
</tr>
<tr>
<td>PFM+SULM_bg</td>
<td>0.8</td>
<td>38.94</td>
<td>16.41</td>
<td>15.20</td>
<td>70.54</td>
</tr>
<tr>
<td>PFM+SULM_40cl-tg</td>
<td>1.2</td>
<td>38.12</td>
<td>19.91</td>
<td>14.92</td>
<td>72.95</td>
</tr>
<tr>
<td>PFM+SULM_bg+40cl-tg</td>
<td>1.0</td>
<td>43.78</td>
<td>13.85</td>
<td>14.26</td>
<td>71.89</td>
</tr>
</tbody>
</table>

Table 3: SU Results†

* a debugged and tuned version of the dryrun system

† these results differ from those presented at the May workshop since they use a more recent version of the su-eval-v01.pl tool
Conclusions

• Standard Lattice-based Viterbi search techniques enable PFM and SULM scores to be combined easily.

• PFMs and SULMs model complementary information.

• Interpolated SULMs can be used to reduce SU %Err.

• Bigram SULMs give largest reductions in %Err when combined with the PFM (using the current training and test data!).

• The current CUED SU System achieves lower %Err values than the type of system used for the dryrun.
Future Plans

- Continue to participate in annotation/tools discussions.
- Develop the PFM (i.e., experiment with other kinds of features).
- Investigate different ways of calculating interpolation weights for SULMs.
- Explore different kinds SULMs (i.e., techniques for training with sparse data).
- Explore different lattice structures (i.e., ‘skips’ instead of SU\_N tokens).
- Consider impact of STT performance upon the SU detection task.
- Use syntactic parsing techniques in post-processing stage to reassign SUs in decoder output to different sub-types.
- Start to focus on the disfluency subset of Structural Metadata tasks.