Leveraging DUC

Terry COPECK, Diana INKPEN, Anna KAZANTSEVA,
Alistair KENNEDY, Darren KIPP, Vivi NASTASE, Stan SZPAKOWICZ
School of Information Technology and Engineering
University of Ottawa
800 King Edward Avenue
Ottawa, Ontario, Canada K1N 6N5
{terry,diana,ankazant,akennedy,dkipp,vnastase,szpak}@site.uottawa.ca

Abstract

Work at the University of Ottawa on text summarization in connection with the Document Understanding Conference 2006 advanced along two tracks. We continued to refine and expand the corpus of SCU-marked documents which we created last year from materials that conference participants received from the group at Columbia University. We also developed an internal manual summary evaluation scheme based on the same responsiveness and quality criteria that NIST's evaluators apply. This scheme and a ranking procedure employing the SCU-marked corpus allowed us to evaluate various heuristics that members of our team implemented. Our DUC submission ultimately incorporated elements of three specialized research projects. This is how in 2006 we took advantage of our continuing involvement in DUC.

1 Introduction

Our team participates in DUC each year to contribute to the communal summarization effort. We also benefit: we leverage various DUC data. In 2006, in particular, we doubled in size the corpus of topic document collections whose sentences are marked with the Pyramid Summary Content Unit (SCU) data provided by Columbia University and other DUC participants (Copeck and Szpakowicz 2005). A systematically growing collection of DUC-provided training data (adding each year's test data) serves as a rallying point in our work. Modules arising from several other research projects in our NLP group contributed to the development of this year's DUC-bound system.

2 Work on Pyramid Data

Last year we successfully linked most of the sentences appearing in the peer summaries (in Pyramid .pan files) back to their origin in a source document. This allowed us to annotate the topic document collections involved with SCU identity and weight data, providing an objective basis for rating the quality of generic summaries.

2.1 Recent Developments

This year we augmented the SCU-annotated corpus with the 2006 data, another 20 unique topics, which doubled it in size. While fewer peers participated in annotation than in 2005—22 versus 27—but almost half this difference in count can be accounted for by the inclusion in 2005 of two human-authored peer summaries for each topic treated. Thankfully, this year only one topic, D0631, was duplicated.

We found more evidence that systems edit source document sentences by removing syntactic elements irrelevant to the purpose of the summary. This year, therefore, we extended the SCU-marking program (Copeck and Szpakowicz 2005) to make a second attempt to match summary sentences which do not have a source document collection hit within the 25% edit distance span used in the Perl `amatch` approximate string match function (employed in the first attempt at matching). In a computationally-expensive operation, the best candidate sentence is now identified by testing increasingly shorter sequences of the tokens in the unmatched summary sentence against the source document collection down to a minimum length of six tokens. Any match found is validated by requiring that

	2006		2005	
Source Sentences	14410		18794	
Summary Sentences	4242	*100%*	5073	*100%*
linked to SCUs	2055	*48%*	2628	*52%*
linked to positive SCUs	2053	*48%*	2076	*41%*
linked to negative SCUs			539	*11%*
linked to source texts	4072	*96%*	4193	*83%*
not linked to source texts	260	*4%*	925	*18%*

Table 1: Counts and Percentages of Summary Sentence Linkages, 2005 and 2006

¾ of the tokens which have been excluded from the tested sequence on successive iterations in order to achieve a hit, do appear somewhere in the hit sentence. Inspection of the results shows this strategy to work well for summary sentences edited by elision. Recognizing the source of summary sentences edited by addition or substantially modified will require a different approach.

The results of linking SCU annotations to source documents were comparable to those achieved the previous year. This information appears in Table 1, which reports matters from the perspective of the summary: how many sentences were linked backward to SCUs, forward to source documents etc.

Document collections averaged 721 sentences this year, 77% of the 940 sentence average last year. This may be due to different content introduced by a change in source periodicals. The lower number of peer collaborators in Pyramid annotation is reflected in a drop in the number of summary sentences—fewer peers, fewer summaries. The number of these sentences which can be linked to Pyramid SCUs is comparable, 48% versus 41% last year. In 2005 another 11% of sentences were explicitly marked as *not* realizing any SCU, increasing to 52% the total marked with SCUs in some manner. Changes in annotation practice in 2006 eliminated this class of negative SCU examples. Another 2006 change was a reduction from seven to four in the number of model summaries on which Pyramids are based. This may explain why the average number of SCUs defined in a topic pyramid decreased from 119 to 80; the results presented above, however, show there was no effect on systems' selection of SCU-marked sentences in the source document collection.

Reworking our SCU matching program improved forward linking into the topic corpus. In 2006, 96% of summary sentences were located in a source document with good confidence, while we had 83% last year (multiple hits account for inexact totals). This success suggests that peer summarizing practices were similar to those employed in 2005.

2.2 SCU Theory and Practice

In our own assessment of evaluation outcomes, we put the most weight on the two measures by which human evaluators directly or indirectly judge how well a summary addresses the topic information need—NIST's Responsiveness measure and Columbia's Modified SCU Score (called modified_score in 2005).

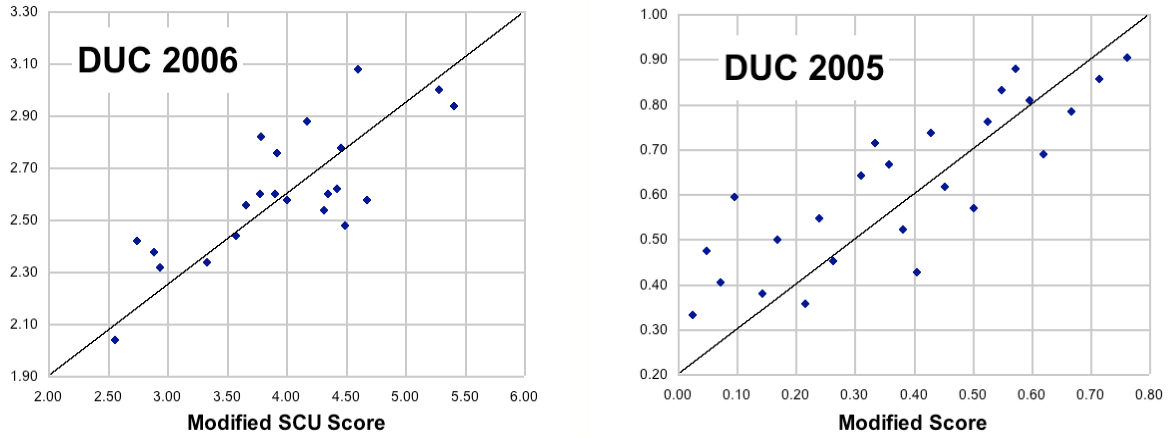

Figure 1: Responsiveness versus Modified SCU Score, 2006 and 2005

It would be hard to imagine a measure more direct than Responsiveness, where judges are explicitly (and simply) told what to take into account in assigning a 1-5 rank (NIST 2006a). Matters are more complex with the Pyramid measure. Here Summary Content Units (arranged in a pyramid) model the information contained in a set of summaries written to answer the topic information need. Their human authors have read the topic document collection—an activity which parallels that of the systems in the conference. Summaries produced by DUC contributors' systems are manually annotated with the SCUs they realize, and scores computed on the degree to which they incorporate SCUs (Passonneau 2006).

How well do the two measures agree? Quite well, it would appear. Excluding the two human control summaries from last year's data, the Pearson correlation coefficient values for 2005 and 2006 are 0.79 and 0.84 respectively. Scatterplots of the two years' data for these variables appear in Figure 1. The significance of this result to us is the validation it provides for the effort spent in propagating SCU counts and weights back from topic pyramids to sentences in the document collection. These values provide a static measure to aid in optimizing system performance; and we now know that they are of good quality—a reasonable approximation to manual assessment.

3 Work on the DUC System

Continued development of the SCU-marked corpus notwithstanding, most of our effort between the 2005 and 2006 conferences has been focused on the summarization task and on improving our system to summarize better. From one perspective, a highlight of the past year has been the adoption of an internal manual evaluation procedure employing NIST's Linguistic Quality and Responsiveness measures. Previous years' systems had been developed on a blind 'best efforts' basis; this year, the availability of a small group of human judges allowed us repeatedly to assess the impact of system changes ourselves without having to wait for the return of conference results.

Growth of the team interested in text summarization at the University of Ottawa also means that more individuals can contribute to the system design. Our 2006 submission benefited from modules imported from the research of three people, who conversely had an opportunity of a scoped, practical application of their work tangential to their long-term academic objectives.

3.1 Internal Evaluation

Our previous DUC summarization systems were developed as a singular effort of a few people. In 2005 the group interested in text summarization doubled in size. When we began planning our 2006 submission, one simple but important way to take advantage of that increased interest was for everyone to serve as evaluators of summaries, creating a feedback loop that could be used to direct our work towards improving our system.

After a little trial and error served to orient everyone and to teach us to make the materials small and regular in format, we used email to perform and report on 13 rounds of Responsiveness and Linguistic Quality assessment of summaries of 2005 topics produced using different configurations of our summarization system. Each round judged three summaries of different 2005 topics. Where data was available, SCU rankings were also computed for the summaries involved.

To increase the data available for evaluation of a given system configuration, summaries were generated and SCU rankings computed for all 21 DUC 2005 topics marked with SCUs, and this supplementary information was also provided to the team (who did not read the summaries involved) when they assessed their evaluations and determined in which direction to move the system configuration.

The benefit of manually assessing summarizer output is obvious and needs no justification. So long as we can do so, we will continue this practice. Note DUC's role in providing the measures used here (and their definitions, which our team used) and in annually recalibrating our assessments with fresh conference results.

3.2 Selecting Configurations

The rounds of assessment described in the previous subsection can only be undertaken to the degree that the system which produces summaries supports quick and

```
> Projects    N:mod:A new
  Projects    N:mod:A hydroelectric

> fin         C:whn:N project
  project     N:det:Det       what
  project     N:mod:A hydroelectric
  fin         C:i:V   plan
  plan        V:be:be be
  plan        V:obj:N project
  in          Prep:pcomp-n:N  progress
  fin         C:whn:N problem
  problem     N:det:Det       what
  fin         C:i:V   associate
  associate   V:be:be be
  associate   V:obj:N problem
  associate   V:mod:Prep      with
  with        Prep:pcomp-n:N  them
```

Table 2: Minipar Output for the D307B Information Request

easy reconfiguration. Ours uses a modular pipeline architecture with well-defined common data structures, so swapping modules in and out or adding additional nodes on the pipeline was not difficult once the common data structures were documented. Summary-generating turns typically took no more than a day or two during the experimental run-up to DUC 2006.

The configuration which ultimately produced our 2006 submission incorporated modules developed by three team members. Each module addressed a quite different facet of the summarization task. Debugging and improvement of the main system into which these modules plug is an ongoing operation, and also factors into the annual performance of our entire system.

The new module first encountered in the processing sequence expands on traditional keyphrase approaches to match graph structures. These are expressed as relational tuples derived from Minipar parses of source document sentences. The advantage of this technique is its capacity to match non-contiguous sequences in the text and to recognize equivalent syntactic structures to some degree. A second module added in 2005 and used again this year employs ROUGE to filter from an ordered list of candidate summary sentences those whose information content appears to be redundant. The output of a third new module is applied to replace third-person singular pronouns with their referents in the summary when these can be identified in a sentence's context in the source document. This is done to improve fluency. Each module is discussed in greater detail in a subsection below.

3.2.1 Graph-matching

The following section describes one technique for ranking individual sentences on their suitability for use in a summary meant to answer an information need, or query (Nastase and Szpakowicz 2006). We begin the process by applying the Minipar parser (Lin 1993) to the titles and contents of each topic information request, and to all documents in its collection. The parser output is then post-processed to identify all dependency pairs for open-class words: when the process encounters prepositions or clausal connectives, it traverses them to link any open-class words involved in a binary relation.

Working through an example may make this process clearer. Table 2 shows the Minipar output for the D307B query *What hydroelectric projects are planned or in progress and what problems are associated with them?* After post-processing, the sentence is represented by the two lists of words and relations given in Table 3. Note that parsing and subsequent processing is not error-free: in the example *with* has been traversed in the parse output to incorrectly link *associated* with *them*.

To accommodate synonymy and grain changes caused by generalization or specialization, the list of words is expanded with the *WordNet* synset elements and one-step hypernyms and hyponyms for all nouns

```
LIST OF WORDS:
associate
hydroelectric
in
plan
problem
progress
project
projects
them

LIST OF PAIRS:
relation(project,hydroelectric)
relation(projects,hydroelectric)
relation(associate,problem)
relation(plan,project)
relation(in,progress)
relation(associate,them)
```

Table 3: Open-Class Words and Dependency Pairs in the D307B Information Request

and verbs appearing in it. Experiments on the 2005 DUC data showed that limiting expansion to these two parts of speech gave better results than applying it to all open-class words. Graph-matching and keyword path search processes are then run on these data structures to assess the similarity of individual document sentences to the query.

In the graph view of a sentence adopted here, nodes are open-class words and edges are dependency relations. A match is found when nodes in each data structure are identical or are members of the *WordNet* expansion of query words. A sentence graph-match score is computed as

$$S = S_N + \text{WeightFactor} * S_E$$

where

S_N, the node match score, is the node (keyword) overlap between the two text units;

S_E, the edge match score, is the edge (dependency relation) overlap

$\text{WeightFactor} \in \{0,1,2,..,15,20,50,100\}$

Similarity is also assessed by looking for paths between any pair of query nodes, the actual words or their *WordNet* expansions, in the sentence graph. Adding this element into the equation produces the similarity formula which was used in DUC 2006:

$$S = S_N + \text{WeightFactor} * (S_E + S_P)$$

where

S_P, the path score, the number of query word pairs connected in the sentence graph

WeightFactor allows the respective contributions of nodes and edges to be tuned by trial and error to give the best summarization performance measured in terms of SCU counts or Responsiveness. For the submission run the factor was set to 15, giving significant emphasis to the two edge-related components. Work continues actively on refining these factors and in general on a graph-matching basis for summarization.

3.2.2 Filtering Out Redundant Sentence

Summaries may be repetitive, especially when they are based on a number of documents discussing the same matters. The problem is large enough to warrant explicit mention in the list of facets of Linguistic Quality: "There should be no unnecessary repetition in the summary" (NIST 2006b). To address the problem of redundant information in summaries of multiple documents, we applied the ROUGE system (Lin 2004) to measure similarity between sentences.

1: The U.S. National Transportation Safety Board (NTSB) said last week it had been unable to conclude what caused the crash of EgyptAir Flight 990, which was heading for Cairo from New York on October 31, 1999, when it suddenly plunged into the ocean, killing all 217 people on board.
27: The EgyptAir flight crashed Oct. 31 off the Massachusetts island of Nantucket, killing the 217 people on board.

Table 4: Two sentences deemed redundant by scoring 0.63636 from topic D0617h

A ROUGE score was computed for certain pairs of sentences for each of the 50 document sets. The redundancy process was run after the initial sentence scoring process described in the preceding subsection. While there is no practical impediment to running ROUGE on all possible pairs of sentences from each document set, each run has certain processing cost and the task is combinatorial—n choose 2. We therefore settled on comparing the highest-ranked 50 sentences from each document collection as previously determined by the sentence selection algorithm. This required a more tractable 1225 sentence comparisons per topic.

We considered this number of sentences sufficient because a 250-word summary—20 sentences at the most—was likely to be composed from the top 50 sentences no matter how much subsequent processing changed their relative position in the ranking. Speaking in terms of this module, for a sentence ranked worse than 50th to appear in the final summary, over 60% of the highest-ranked sentences would have to be found to be mutually repetitive. This is unlikely to occur.

ROUGE 1.5.5 was run using the Porter Stemmer (Porter 1983), with stop words removed and with the average R score from ROUGE-L (longest common subsequence) chosen as the basis on which to assess redundancy. While our evaluation of which ROUGE measure and settings produce the best result was not comprehensive, trials did continue until inspection showed acceptable results. Thus, although the chosen

settings may not be optimal, they do indeed identify repetitive sentences to a degree we find satisfactory.

The results of these 1225 pairwise tests were used to remove from consideration the lower-scored sentence in any pair whose ROUGE-L score greater than or equal to 0.5. As with the choice of system values and settings, this threshold was determined empirically. Exhaustive experimentation was not undertaken to establish that the 0.5 threshold is optimal, the value was simply found to work well upon an examination of the results. This threshold can readily be made higher or lower depending on how aggressively one wishes to remove potentially redundant sentences.

Table 4 shows an example of redundancy elimination involving two sentences from topic D0617h. The leading integer shows the sentence's ranking before redundancy processing; the upper ranked first after initial sentence selection while the lower ranked 27th. Words in common are underlined. The two sentences generated a ROUGE-L score of 0.63636 and the lower one was therefore dropped.

An additional benefit of using this particular module arises from its side effect of discouraging two sentences with long strings of the same words from appearing together in the summary. Since redundancy is seen to occur not only in sentences, but also in "the repeated use of a noun or noun phrase" (NIST 2006b) this should tend to improve a summary's Linguistic Quality rating. By increasing the variety of sentences in the summary, it may also make it more readable.

3.2.3 Resolving Pronoun References

Pronouns appearing in a summary created from extracted sentence all too often suggest the wrong referent if one reads the sentences out of context. We have therefore attempted to eliminate such pronouns, when experiments using the 2005 data showed they could be guaranteed not to make the resulting text less grammatical. The anaphora resolution module in our DUC 2006 system (Kazantseva 2006) was created to resolve one particular kind of referring expression, those that denote people. The module finds antecedents of 3rd person singular pronouns (*he*, *her* etc.) and singular definite noun phrases that refer to people (e.g. *that woman*). The module is implemented in Java as a plug-in for the GATE framework (Cunningham, Maynard, Bontcheva and Tablan 2002). The plug-in relies on the output of a syntactic parser, the Connexor Machinese Syntax Parser (Tapanainen and Järvinen 1997).

The system operates in two steps: first it identifies instances of co-referring entities and then it locates anchors—referents—for them. Initially, a document of interest is parsed using the Connexor parser and the parse structures loaded into GATE. The Gazetteer module in GATE annotates it for instances which mention persons and for those persons' gender. The next step involves identifying anaphoric expressions in texts. Pronominal anaphoric expressions are recognized using a hard-coded list of (3rd person pronouns). Identification of anaphoric noun phrases is less straightforward. To accomplish it, we implemented the rules proposed for this purpose in Poesio and Vieira (2000). Poesio and Vieira identify candidate instances in the text, test them for a variety of syntactic characteristics (such as being an appositive or a copula, having a proper head noun or a restrictive post-modifier), and then apply a series of heuristics to these criteria to determine which of the candidates are in fact anaphoric noun phrases.

Once anaphoric expressions have been identified, our system attempts to find for each such expression an anchor — the entity referred to by the expression in focus. To this end we implemented the rule-based syntactically-motivated algorithm RAP, described in Leass and Lappin (1994). RAP accumulates values for a number of salience parameters for noun phrases and then applies a decision-making procedure to select from among these the most likely antecedent for a given pronoun. The parameters of interest to the algorithm are: grammatical role, parallelism of grammatical roles, frequency of mention, proximity, and sentence recency.

This algorithm was intended only to resolve 3rd person pronouns. We deal, however, with a very limited subset of noun phrase anaphora involving singular animate expressions and our candidate anchors are annotated with gender information. That is why we found Leass and Lappin's RAP algorithm to be acceptably effective at finding antecedents for them.

4 Results

The work we presented in the previous section had some effect on the ranking of our results in 2006. The greatest improvement was in Linguistic Quality, where we moved up significantly in the ranking. Our Responsiveness score also improved somewhat, an outcome which may be a consequence of our effort in the preceding months. These two results place us in the middle of the pack on NIST's Overall score. Our results on the automated (ROUGE, BE) and semi-automated (SCU) assessments remain unchanged from 2005 and are quite poor. In consequence, our system is one of the outlier points on the 2006 Responsiveness/SCU scatterplot in Figure 3.

5 Future Work

We will continue to update the corpus of SCU-marked topics with new material as it becomes available, and to use it to guide future development of our summarization system. Future work on sentence redundancy includes the continued use of ROUGE coupled with rigorous experiment to determine what settings, measures and thresholds produce the best results. We will also explore other redundancy measures that may supplant or supplement ROUGE. As already noted, approaches to summarization using graph representation of sentences remain an active interest.

Acknowledgements

Partial support for this work comes from the Natural Sciences and Engineering Research Council of Canada.

References

Copeck, Terry and Stan Szpakowicz. 2005. Leveraging Pyramids. *Proc Workshop on Automatic Summarization* (DUC 2005), HLT/EMNLP-2005.

Cunningham, Hamish, Diana Maynard, Kalina Bontcheva and Valentin Tablan. 2002. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. *Proc 40th Anniversary Meeting of the Association for Computational Linguistics*. Philadelphia, July 2002.

Kazantseva, Anna. 2006. An Approach to Summarizing Short Stories. *Proc Student Research Workshop at EACL 2006,* 47-55.

Leass, Herbert and Shalom Lappin. 1994. An Algorithm for Pronominal Anaphora Resolution. *Computational Linguistics*, 20(4), 535-561.

Lin, Chin-Yew. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. *Proc Text Summarization Branches Out*, Post-Conference Workshop of ACL 2004, Barcelona, Spain.

Lin, Dekang. 1993. Principle-Based Parsing without Overgeneration. *Proc ACL-1993*, 112-120.

Nastase, Vivi and Stan Szpakowicz (2006) "A Study of Two Graph Algorithms in Topic-driven Summarization". *Proc TextGraphs 2006, workshop at NAACL 2006*, New York, to appear.

NIST. 2006a. Responsiveness Assessment Instructions. www-nlpir.nist.gov/projects/duc/duc2006/responsiveness.assessment.instructions

NIST. 2006b. Linguistic Quality Questions. www-nlpir.nist.gov/projects/duc/duc2006/quality-questions.txt

Passonneau, Rebecca. 2006. Pyramid Annotation Guide: DUC 2006. www1-cs.columbia.edu/~becky/DUC2006/2006-pyramid-guidelines.html

Poesio, Massimo and Renata Vieira. 2000. An Empirically Based System for Processing Definite Descriptions. *Computational Linguistics*, 26(4), 525-579.

Porter, Martin. 1983. An Algorithm for Suffix Stripping, *Program*, 14(3):130-137.

Tapanainen, Pasi and Timo Järvinen. 1997. A Non-Projective Dependency Parser. In *Proc 5th Conf on Applied Natural Language Processing*, 64-71.