TLR at DUC: Tree similarity

Frank Schilder and **Andrew McCulloh** and **Bridget Thomson McInnes**[*] and **Alex Zhou**

Thomson Legal & Regulatory
R&D
610 Opperman Drive
Eagan MN 55123, USA

[*] Department of Computer Science and Engineering
University of Minnesota
200 Union Street SE
Minneapolis MN 55455, USA

Abstract

We provide a solution to this year's task of a question-based multi-document summarization by employing tree similarity of the dependency parse trees for reformulated questions and candidate sentences.

1 Introduction

This paper describes our contribution to the 2005 Document Understanding Conference. We developed an approach to match questions to possible answer sentences from a document collection. More specifically, we adopted a recent proposal to fact-based question-answering, as described by Punyakanok et al. (2004). They show that computing the tree similarity of dependency parse trees between a question and candidate answer sentences outperforms a simple bag-of-words approach. It is our impression that tree similarity scores have not been used for the generation of longer text summaries.[1] We tried this approach for this year's DUC competition and received competitive results.

The remainder of this paper is organized as follows. After a brief introduction to the task, we will outline the system's modules in section 3. In section 4, we will analyze the results in more detail and propose a modification of current scoring methods in order to capture coherence and text structure. In section 5, we will conclude and discuss future extensions of our system.

2 Task description and goals

This year's task was a complex summarization and question answering task. Given a short list of questions, systems were required to construct a summary based on a set of twenty five to fifty documents. This summary was tailored to meet a complex information need expressed as a natural language question (e.g. *By how much is world population projected to grow or decline in the next century, and what are the principal factors influencing growth or decline worldwide and in specific countries?*). This task is significantly more complex than the typical information extraction problem of simply extracting a name, date, or other facts. To answer a natural language question, a well-composed and coherent passage needs to be generated. The task is also more user-oriented than former multi-document summarization tasks, because the summary had to be tailored to the information need. Moreover, a specific versus general distinction was given for each topic. This distinction models a simple user profile.[2]

The DUC road map committee for 2005-2007 had originally intended to come up with a more purpose-oriented summarization task requiring the fusion of information from a range of different source types. But another issue gained prominence in the discussions over the last year. The evaluation program committee thought that it would be better to address the issues of human variation in summary evaluation, as discussed specifically at the EACL 2004 workshop on summarization in Barcelona. Based on this discussion, it was decided to have a simpler multi-document (but not multi-media) summarization task that remained somewhat user-oriented.

There were three main goals for this year's DUC:

1. Inclusion of user and task context information in summaries generated by automated systems and human summarizers.

2. Evaluation of content in terms of more basic units of meaning.

3. Improving the understanding of normal human variability in a summarization task and how it may affect evaluation of summarization systems.

[1] Tree similarity scores, however, have been applied to a more complex task of textual entailment (Kouylekov and Magnini, 2005). For this task, one has to decide whether a short text entails a given statement (Dagan et al., 2005).

[2] The task was suggested by ”An Empirical Study of Information Synthesis Tasks” written by Enrique Amigo, Julio Gonzalo, Victor Peinado, Anselmo Penas, Felisa Verdejo.

Our contribution provides some insights on the first point by modeling the task context via tree similarity scores for the queries and candidate sentences that could satisfy a given information need.

In addition, we also include a proposal on how the automatic evaluation metrics can be improved. We noticed that neither the ROGUE nor the Pyramid method provide a good measure for the overall coherence and text organization of the automatically generated summaries. Our modification on how these two scores are computed will capture this crucial feature of a well-written summary.

3 Our approach

Our approach is based on computing similarity scores between the questions and candidate sentences. In particular, our measure computes a distance between the dependency parse trees of two sentences. This approach has already been tried for fact-based question answering systems, but has not been applied to complex questions that require a cohesive answer.

Our summarization algorithm performs the following three major processing steps:

- **Linguistic smoothing (LS).** In order to avoid dangling pronouns and incoherent rhetorical structures we developed two modules that carry out substitutions and deletions on the text. We begin by applying a pronoun substitution to the sentences in the documents. This component incorporates the output from an existing pronoun resolution tool (i.e. LingPipe[3]). Next, the rhetorical smoothing tool removes some phrases from the sentences. It utilizes a list of discourse markers and contains a simple discourse parser.

- **Question reformulation (QR).** Some preprocessing of the questions is necessary so that candidate sentences form the document collections can be compared to the query to compute a distance score. The second processing step translates every question into an affirmative sentence (e.g. *What is the World Bank?* → *The World Bank is *ANSWER*.*). In the remainder of the paper, we will refer to these sentences as the **ANSWER**-statements.

- **Tree extraction (TE).** Using a dependency parser (MiniPar, (Lin, 1998)) we parsed the reformulated questions and the sentences in the topic collection. A tree similarity score was computed between the **ANSWER**-statements and the sentences from the collections. The most similar sentences, with respect to the given question, were extracted from the set of candidate sentences in the collection.

The resulting summary was extracted from the document collection by identifying the highest scoring sentences with respect to the **ANSWER**-statements. The number of sentences was limited by the 250 word limit on the summaries. To improve linguistic quality we avoided truncating sentences, discarding the final sentence that would cause the summary to exceed the 250 word limit. This decision diminished the recall significantly, but produced high precision values. Unfortunately, the ROUGE evaluations are recall-based, based on the assumption that all summaries comply to a fixed word length.

The following subsections describe the three major processing steps of our algorithm in greater detail.

3.1 Linguistic smoothing

When sentences are extracted from a document and put into a summary, surrounding context is lost. Without ties to the document's other sentences these broken connections can seriously impact the coherence of the summary text. There are several types of context links, our system focused on the following two: (a) pronouns and (b) discourse markers.

3.1.1 Pronoun substitution

The goal here is to eliminate dangling or unresolved pronouns. Since the extracted sentence is often put into the summary next to a sentence extracted from an entirely different document, antecedent determination for pronouns would, in the best case, be confusing and at worst impossible.

We solved this problem by using a pronoun resolution system called LingPipe. This software resolves the antecedents for pronouns by labeling entities and linking the pronouns to these labels. If the antecedent and pronoun can be found in the same sentence, such a substitution may lead to wrong inferences. In the following example sentence, *he* would be co-referential with *Peter*. However, substituting *he* with *Peter* would lead to the inference that another Peter is meant.

(1) Peter$_i$ said that he$_i$/Peter$_j$ liked the movie.

Consequently, we needed to post-process the LingPipe output by writing rules on when to substitute and when not. In addition, there were times when no antecedent was resolved. LingPipe did not seem to work very well for antecedents that were more than a couple of sentences away from the sentence containing the pronoun. We also added rules to improve the overall resolution accuracy.

3.1.2 Coherence maintenance

Some discourse markers signal a rhetorical relation that holds between a sentence and the preceding discourse:

[3] http://alias-i.com/lingpipe/

(2) **However**, Mr Watanabe says he fully expects (. . .)

If such a sentence were included in the summary, the rhetorical connection to the preceding discourse would be broken and coherency reduced. Consequently, we deleted discourse markers that point to the preceding discourse, while keeping markers that signal a sentence-internal relation, such as *although*.

(3) **Although** Budik had originally been identified as Smith's wife, she clarified the couple's relationship Thursday, saying they were not married but had lived together for two years.

We used Marcu's list of discourse markers (Marcu, 1997) and classified them according to these two types. Then we wrote a simple discourse parser to extract the discourse markers that can cause dangling rhetorical relations.

3.2 Question reformulation

The Question Reformulation (QR) tool transforms a set of questions into an affirmative or *ANSWER*-statement for each question. For example, the *ANSWER*-statement for the input question *What hydroelectric projects are planned?* is as follows:

(4) *ANSWER* are hydroelectric projects that are planned.

The **ANSWER** tag is then used to indicate a place for the actual answer to be incorporated directly into the statement. This reformulated sentence is compared to the sentences from the document collection and similar ones are extracted as summary candidates.

The QR tool contains four modules: a sentence splitter, a part-of-speech (POS) tagger, a shallow parser, and the statement generator. The sentence splitter and POS tagger process the data so that it can be parsed by the shallow parser. First, the sentence splitter extracts the questions and then the POS tagger tags each word in the sentence. The system uses the BRILL POS tagger (Brill, 1992).

The third module is the shallow parser. The parser used for this system is the CASS parser developed by Abney (1990). CASS consists of a series of cascading finite-state transducers. In addition to the partial parse of the sentence, the parser also provides subject and object information.

The last module, the statement generator, extracts four components from the CASS parser output: the question word, the main verb, the subject, and the object. It then feeds the sentence into a cascading set of rules that, based on these four components, creates a template statement that the question can be transformed into.

Consider the following example query, *What problems are associated with them?*. Given the information from the CASS parser the statement generator sequentially applies the following rules: the first rule set identifies the question word. In this case, the sentence contains the question word *what*. The second rule set identifies that the verb is (*are*.) Finally, the subject and object are identified. In our example, the subject is *problems* and an object does not exist. The sentence is then transformed using the following answer template:

(5) the <subject> that <verb> <sentence> are *ANSWER*

This template generates the following sentence for our example:

(6) the problems that are associated with them are *ANSWER*

3.3 Tree extraction

To match the **ANSWER**-statement to a candidate sentence we used a tree similarity mechanism. Tree matching algorithms measure the similarity between two trees by comparing subtrees and computing a similarity measure over them. This similarity measure between candidate sentences and the query phrases is analogous to a string edit distance. The document sentence with the fewest differences, hence the smallest distance, to the query sentence is considered the best candidate for the summary. We chose a tree edit distance focused on the dependency tree of the sentences after (Punyakanok et al., 2004).

The general tree edit distance algorithm, like the analogous string edit distance uses dynamic programming to compute the minimum number of inserts, deletions, and translations required to transform one tree into another. Like the string edit distance each of these operations is assigned some cost. By altering the cost function different facets of the similarity can be emphasized. Unlike string edit distance the operations act on nodes and modify the tree by changing the parent/child relationships of given trees. A string edit distance translation implies changing the label on a node in the tree. Deletion implies removing the node from the tree and attaching the deleted nodes children to its parent. Finally, insertion involves adding a new child to a node and possibly making a consecutive subset of the original parent's children, children of the added node. For examples see Figure 1.

The basic tree edit distance algorithm has been extended by (Shasha and Zhang, 1989) to include *don't care* nodes, which allow for approximate matches. Any differences between such a *don't care* node and nodes in a second tree are not counted. We treat nodes containing the **ANSWER** indicator as these *don't care* nodes. Then we can use the tree edit distance to look for candidates where part of the sentence matches the query tree while the part

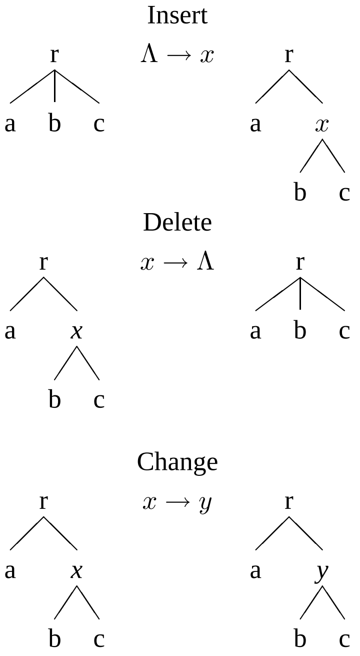

Figure 1: The three editing options

matching the *don't care* node may contain desired information.

We computed the differences in the edit distance as follows. We wanted to penalize those candidate sentences which added information to the tree outside of the *don't care* region and penalize those trees where the distance required deleting information from the query tree. In addition we modified the scores to contain more or less of a penalty depending on several other factors. If two nodes contained the same word the penalty was zero. The largest penalty was reserved for a node change when completely different words were compared. Here the penalty was high as the two trees had very different information at the subtrees being compared. The entire list of penalties can be seen in Table 1.

The answer token (**ANSWER**) was handled separately by the tree distance algorithm. A node with this string would indicate to the program that it needed to take a special path of execution and did not require that we determine a penalty when comparing that node to a subtree.

To implement the program we took the Minipar API (Lin, 1998)) and wrote a wrapper around it for Python using SWIG.[4] We then implemented the tree distance algorithm in Python. The collection of sentences, both from the queries and the documents had all trailing punctuation stripped to facilitate comparison. The query sentences associated with each topic were parsed by the Minipar parser and then compared to the dependency trees of sentences extracted from the corresponding document collection. A score for each query was then associated with each sentence from the collection. The sentences with the least distance were considered to have the best scores and were passed on to the next stage of the pipeline.

action	kind	score
insert	regular word	5
insert	stop word	200
delete	regular word	200
delete	stop word	5
translate	identical words	0
translate	same root form	1
translate	otherwise	200

Table 1: Distance penalties

3.4 Results

NIST carried out the evaluation by automatic means (i.e. ROUGE, (Lin, 2004)) as well as by human annotation (including measures of responsiveness and linguistic quality). The ROUGE score was introduced to DUC last year. It measures the n-gram word-overlap between the automatically generated summary and N human-written model summaries. Of the many variations of possible ROUGE scores the macro-averaged ROUGE-2 and the ROUGE-SU4 scores were the two that were officially reported and can be found for all systems in Figure 2 and 3 (our system ID: 20).

The human-annotated results are based on two factors. First, annotators rated how well the summary satisfies the information need expressed by the questions. This score was called responsiveness and rated according to a scale from 1 to 5 (5 being most responsive).

Second, the summaries were rated for overall linguistic quality. This rating was broken down into five quality criteria:

1. Grammaticality
2. Non-redundancy
3. Referential clarity
4. Focus
5. Structure and Coherence

In addition, Columbia University carried out an evaluation based on the pyramid evaluation method, as described in (Nenkova and Passonneau, 2004). This method addressed the problem of human variation in summarization by hand-annotating human-written summaries. For this method, a pyramid of so-called summary content units (SCU) is compiled for the human-written summaries. An SCU is realized by a so-called contributor in a

[4] http://www.swig.org

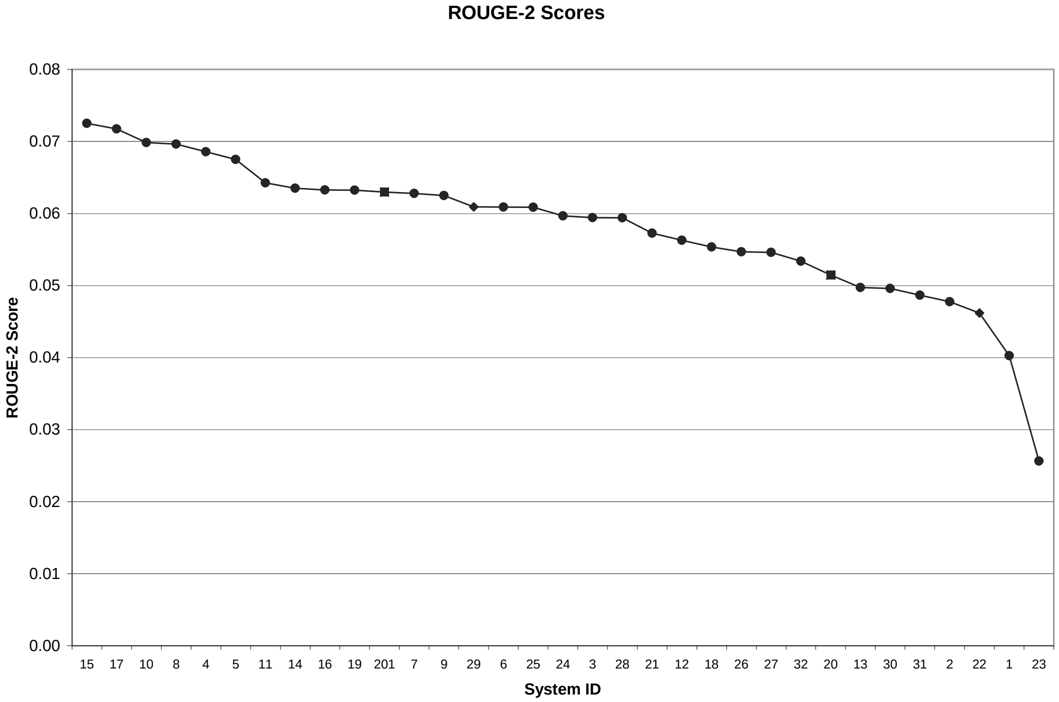

Figure 2: ROUGE-2 scores

summary which can be a short sentence, a clause or only a phrase. Each SCU expresses a topic mentioned by the summary; SCUs that can be found in all human-written summaries are higher ranked than topics that can only be found in one or two human-written summaries.

In order to score the automatically generated summaries, annotators need to map fragments from the automatically generated summary to the list of SCUs previously compiled from the human-annotated summaries. This method captures contributors that refer to the same SCU even though they may be differently expressed (e.g. *John bought a car* and *John's purchase of a vehicle*).

For these evaluation measures, our system performed consistently better than the baseline (ID: 1), but so did most of the other 30 systems. For the automatically-generated ROUGE score we received relatively low ranks (ROUGE-2: 25th; ROUGE-SU4: 30th). However, after looking at the results in more detail we noticed that our system did not use the 250 word limit for the summaries. Our attempt to produce a summary of only complete sentences led often to less than 250 words which hurts recall. Precision, on the other hand, was very high for our system (2nd rank).

In a post-hoc experiment, we adjusted our system to use the full 250 word limit (system ID for rerun: 201). Our scores for recall improved significantly and we reached much better ranks (i.e. ROUGE-2: 11th; ROUGE-SU4: 13th).

For the human-annotated scores from NIST and Columbia University we were obviously not able to resubmit our modified results with the 250 word limit. The scores for responsiveness and the pyramid evaluation[5] show our system at rank 24 (out of 32 systems) and rank 23 (out of 25 systems), respectively. On the other hand, the efforts we put into the linguistic smoothing module showed a higher score for linguistic quality (rank 13).

We also compared the results of the different evaluation methods and plotted the automatically generated scores from ROUGE against the human-based scores in Table 2. Based on the Pearson and Spearman correlations we computed, we notice ROUGE-2/ROUGE-SU4 are highly correlated to responsiveness, but not to linguistic quality.

It is interesting to note that summaries that are not very highly rated in linguistic quality (i.e. redundancy, referential clarity and other criteria) can still score highly in responsiveness and ROUGE. Since the current evaluation did not include an extrinsic evaluation that tested the usefulness of the generated summary to a user, linguis-

[5] The results for the pyramid evaluation varied a lot for the given 20 topics this evaluation method was applied.

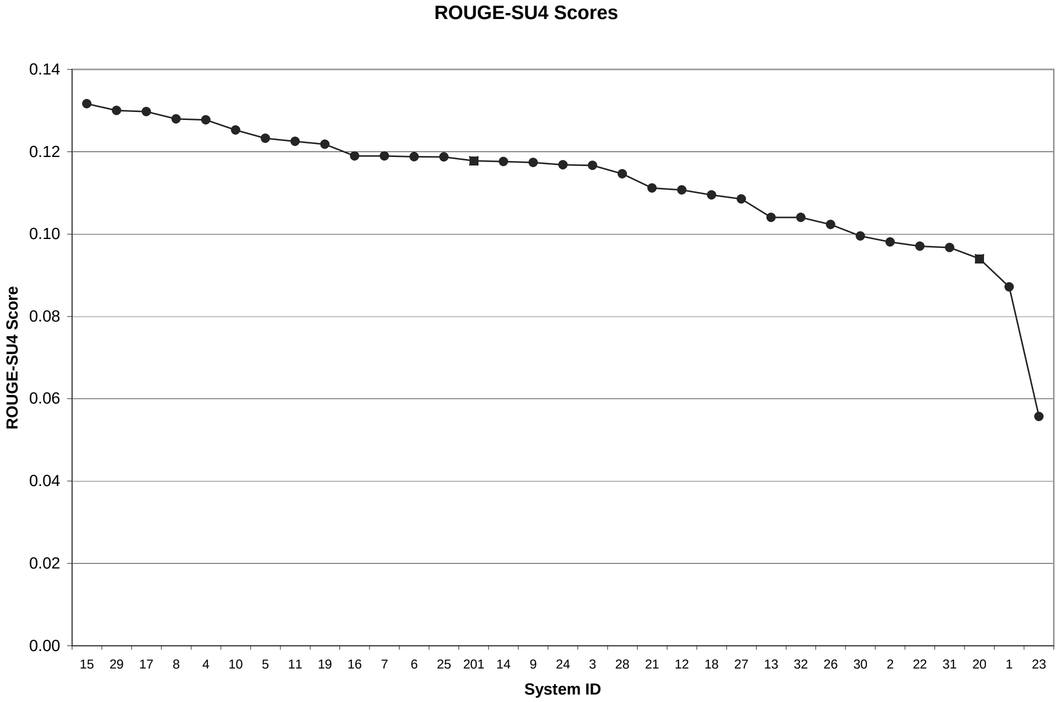

Figure 3: ROUGE-SU4 scores

	Pearson	Spearman
ROUGE-2/ScalRespons	**0.93**	**0.90**
ROUGE-SU4/ScalRespons	**0.92**	**0.87**
ROUGE-2/LingQual	0.03	0.19
ROUGE-SU4/LingQual	0.04	0.13

Table 2: Pearson and Spearman coefficient for ROUGE-2/ROUGE-SU4 and Responsiveness and linguistic quality

tic quality can perhaps be ignored. However, if a more purpose-oriented summarization task had to be evaluated (e.g. situation reports on natural disaster relief status) linguistic quality may become more prominent and actually crucial for the evaluation. It may be possible to measure user's reaction times based on the summary provided. A summary that presents the information in a coherent and is well structured way would probably score higher than one that contains all important information but is of low linguistic quality and consequently difficult to process for a user.

In addition, taking linguistic quality into account may also be another distinguishing factor for scoring all participating systems since many of their scores are not significantly different. If we factor in linguistic quality into the score some systems may show consistently better performance than other systems on all levels (i.e. responsiveness, coverage of SCUs, word overlap and linguistic quality).

4 Evaluation analysis

4.1 System performance

The originally reported results for our system seem disappointing at first sight, but after re-running our system with the full 250 word limit, we actually reached competitive scores. Clearly, there is still room for improvement and one particular area we would like to focus on is the question analysis. The current task seems to indicate an interesting merge of two research areas: summarization and question answering. Consequently, findings from QA systems can be beneficial for this summarization task. Moreover, question analysis has to go beyond what normally is carried out for fact-based QA systems. A more detailed analysis of the question is required. First, the type of question (e.g. fact-based, opinion-based, narrative etc.) has to be determined. Second, generalizations have to be derived from the questions and instances in the collections have to be found. The system has to derive from the phrase *worldwide and in specific countries* that at first statements concerning the world and then facts about particular countries have to be found in the text collection. In particular, the sequence of what had been asked in the question should be reflected in how the answer is constructed. This observation leads us to the question on how the text structure could be better captured by an evaluation score such as the pyramid method or ROUGE.

4.2 Text structure

In order to answer this question we need to do the two following things: First, we need to determine whether information that is considered of high value for a summary is somehow reflected by the text structure of the model summaries. Second, we would like to propose a modification of the pyramid method and ROUGE that takes into account the average position of a SCU or n-gram, respectively.

Consider, for example, the reference summary D for topic 435 (World population):

(7) United Nations population projections in 1993 were for the world population to double from the current 5.5 billion to 11 billion by 2050.[...] Western Europeans countries are experiencing a “baby bust”; with extremely low birth rates in all countries except Ireland and Poland.

Most of the reference summaries mention the UN projection at the beginning of the summary, because it directly answers the topic question: *By how much is world population projected to grow or decline in the next century?* By providing modified weights based on the contributor’s average position for all model summaries, we could emphasize the central ideas at the right position in the text and penalize contributors that are placed somewhere else in the text.

Consequently, our assumption is that SCUs with a high weight tend to occur at the beginning of the model summaries, whereas contributors for SCUs with a low weight can mainly be found at the end of the summary. In order to verify this assumption, we took the pyramid *.edt.pan data files and computed where the contributors start within the model summaries and mapped this offset to a normalized score ranging from 0-100. The boxplot in figure 4 shows the mean and the 25% and 75% percentiles for each SCU type.

Our results show that contributors for SCUs with weight 7 are more often found at the beginning of a summary. Almost 75% of all contributors for a SCU of weight 7 are found in the first fifty percent of a summary. The mean position for a contributor is at about 25% of the summary text. However, the other SCU’s contributors are more or less below the 50% mark indicating that these SCUs are uniformly distributed and that there is no strong relation between the weight of the SCU and its position in the summary.

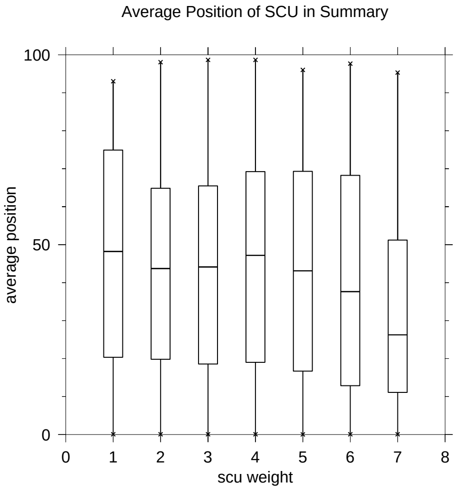

Figure 4: Average position for SCUs of weight *n*

Nevertheless, we would like to propose a modification of the Pyramid (or the ROUGE) score that takes into account the position of the SCU contributor within the text. In addition to the mean position for a given SCU, we also need to compute the standard derivation for each SCU to measure the degree of agreement among human summarizers.

A weight should be computed capturing this information. If we assume, for example, that 95% of all contributors for a SCU appear very close to the beginning, a matching contributor for this SCU from an automatically generated summary that does not fall within this range should receive a lower score compared to a contributor that can be found at the beginning of the text. If the contributors can be found at various positions in the model summary, on the other hand, the standard deviation s should be high and a contributor from an automatically generated summary is likely to fall within this range.

We propose the following formula for computing such a weight:

$$w = \begin{cases} 1 & : \frac{(SCU_{avg} - Pos)}{s} < 1 \\ 1/\frac{(SCU_{avg} - Pos)}{s} & : otherwise \end{cases}$$

This weight could be used for the Pyramid evaluation as well as for the ROUGE scores.

5 Conclusion and future work

We investigated the applicability of a tree matching algorithm for question-based summarization. ROUGE recall scores as well as hand-annotated responsiveness and pyramid evaluation scores were relatively low, but the ROUGE precision score was very high. After carrying out post-hoc experiments using the full 250 word limit for a summary, we obtained overall competitive results for the ROUGE-2 and ROUGE-SU4 scores.

Our approach combined with smoothing techniques led to high linguistic quality of the answers which may be important if acceptability and usefulness would have been tested for this task. An extrinsic evaluation method is needed for this aspect of the summarization task. However, this was not the focus of this year's competition (cf. (Dorr et al., 2005)).

In the future, we are going to look at two areas where we can improve our system. The first is question analysis. We believe that a more detailed question analysis will lead to better results, as results from question answering research have shown. In addition, linguistic smoothing techniques should also be applied to the questions and not only to the sentences from the document collection. This has not been an issue for fact-based question answering systems, because only one question is considered at a time.

The second area of anticipated improvement to our system is in the matching of the tree similarity component. For this competition we chose to use a standard metric for computing the tree edit distance. We are currently investigating the possibility of using a broader synonym set for the node comparison in the tree similarity algorithm. For example, WordNet (Fellbaum, 1998) could be used to give smaller penalties in tree distance depending on weather the lemmas of the terms in the nodes are synonyms or hyponyms, etc. Alternatively, a verb frame database could be used to focus the candidate selection to those sentences which have similar verbs to the question.

Acknowledgment

We would like to thank the editors Madonna Backstrom, Holly Koski, Christina McCann and Jean Wassenaar for their evaluation work on the sample data and their efforts in the pyramid evaluation and Khalid Al-Kofahi for his helpful comments.

References

S. Abney. 1990. Rapid incremental parsing with repair. In *Proceedings of the 6th New OED Conference*, Waterloo, Ontario.

Eric Brill. 1992. A simple rule-based part-of-speech tagger. In *Proceedings of ANLP-92, 3rd Conference on Applied Natural Langu age Processing*, pages 152–155, Trento, IT.

Ido Dagan, Oren Glickman, and Bernardo Magnini, editors. 2005. *The PASCAL Recognising Textual Entailment Challenge*.

Bonnie Dorr, Christof Monz, Stacy President, Richard Schwartz, and David Zajic. 2005. A methodology for extrinsic evaluation of text summarization: Does ROUGE correlate? In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 1–8, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Christiane Fellbaum. 1998. *WordNet – An electronic lexical database*. MIT Press, Cambridge, Massachusetts and London, England.

Milen Kouylekov and Bernardo Magnini. 2005. Recognizing textual entailment with tree edit distance algorithms. In Oren Glickman Ido Dagan and Bernardo Magnini, editors, *Proceedings of the PASCAL Recognising Textual Entailment Challenge*, April.

Dekang Lin. 1998. Dependency-based evaluation of minipar. In *Proceedings of the Workshop on the Evaluation of Parsing Systems, First International Conference on Language Resources and Evaluation*, Granada, Spain, May.

Chin-Yew Lin. 2004. ROUGE: a package for automatic evaluation of summaries. In *Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004)*, Barcelona, Spain.

Daniel Marcu. 1997. *The Rhetorical Parsing, Summarization, and Generation of Natural Language Texts*. Ph.D. thesis, Department of Computer Science, University of Toronto. Also published as Technical Report CSRG-371, Computer Systems Research Group, University of Toronto.

Ani Nenkova and Rebecca Passonneau. 2004. Evaluating content selection in summarization: The pyramid method. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 145–152, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.

V. Punyakanok, D. Roth, and W. Yih. 2004. Mapping dependencies trees: An application to question answering. In *Proceedings of AI&Math 2004*.

D. Shasha and K. Zhang. 1989. Fast parallel algorithms for the unit cost editing distance between trees. In *SPAA '89: Proceedings of the first annual ACM symposium on Parallel algorithms and architectures*, pages 117–126, New York, NY, USA. ACM Press.