In Proceedings of the 2007 Document Understanding Conference
(DUC-2007) at NLT/NAACL 2007, April 2007, Rochester, New York.

Multiple Alternative Sentence Compressions for Automatic Text Summarization

Nitin Madnani, David Zajic, Bonnie Dorr, Necip Fazil Ayan & Jimmy Lin
Institute for Advanced Computer Studies
University of Maryland
College Park, MD, 20742
{nmadnani,dmzajic,bonnie,nfa,jimmylin}@umiacs.umd.edu

Abstract

We perform multi-document summarization by generating compressed versions of source sentences as summary candidates and using weighted features of these candidates to construct summaries. We combine a parse-and-trim approach with a novel technique for producing multiple alternative compressions for source sentences. In addition, we use a novel method for tuning the feature weights that maximizes the change in the ROUGE-2 score (Δ_{ROUGE}) between the already existing summary state and the new state that results from the addition of the candidate under consideration. We also describe experiments using a new paraphrase-based feature for redundancy checking. Finally, we present the results of our DUC2007 submissions and some ideas for future work.

1 Introduction

This paper presents MASC (Multiple Alternative Sentence Compressions), a framework for using sentence compression in automatic summarization. MASC systems use a sentence compression module to generate multiple compressions of source sentences in combination with a candidate selector to construct a summary from the compressed candidates. The selector uses a combination of static and dynamic features to select candidates that will maximize relevance while minimizing redundancy within the summary.

In previous work (Zajic et al., 2007), the weights assigned to the features were manually optimized to maximize the ROUGE-2 recall score on a held out dataset. In the MASC framework, we generate n-best lists of candidates at each iteration and employ an automatic optimization technique that tunes the weights to directly maximize the *change* in the ROUGE-2 recall score (Δ_{ROUGE}) that would result from adding a candidate from the n-best list to the current summary state. As with manual optimization, we use a held out dataset for which we have access to model summaries.

The update task presents an interesting challenge and an ideal platform to test another novel component of MASC—an update-oriented redundancy checking technique. We adapt our existing redundancy feature to suit the update task requirements.

The next section relates our approach to other existing summarization systems. Section 3 gives an overview of the MASC framework. Section 4 discusses the optimization technique in detail and Section 5 explains the adaptation of the existing interpolated redundancy feature for the update task. Section 6 presents some post-hoc experiments on sentence filtering and paraphrase-based redundancy checking. Finally, in Section 7, we present an analysis of our main and update task submissions and ideas for future work.

2 Background

Extractive multi-document summarization systems typically rank candidate sentences according to a set of factors. Redundancy within the summary is minimized by iteratively re-ranking the candidates as they are selected for inclusion in the summary. For example, MEAD (Radev et al., 2004; Erkan and Radev, 2004) ranks sentences using a linear combination of features. The summary is constructed from the highest scoring sentences, then all sentences are rescored with a redundancy penalty, and a new summary is constructed based on the new ranking. This process is repeated until the summary stabilizes. Maximal Marginal Relevance (MMR) (Carbonell and Goldstein, 1998; Goldstein et al., 2000) balances relevance and anti-redundancy by selecting one sentence at a time for inclusion in the summary and re-scoring for redundancy after each selection. Our system takes the latter approach to summary construction, but differs in that the candidate pool is enlarged by making multiple sentence compressions derived from the source sentences.

There are several automatic summarization systems that make use of sentence compression (Jing, 2000; Daumé and Marcu, 2005; Knight and Marcu, 2002; Banko et al., 2000; Turner and Charniak, 2005; Conroy et al., 2006; Melli et al., 2006; Hassel and Sjöbergh, 2006). All such approaches perform sentence compression through removal of non-essential sentential components (e.g., relative clauses or redundant constituents), either as preprocessing before selection or post-processing after selection. Our approach differs in that multiple trimmed versions of source sentences are generated and the selection process determines which compressed candidate, if any, of a sentence to use. The potential of multiple alternative compressions has also been explored by (Vanderwende et al., 2006).

3 Overview of MASC

Multiple Alternative Sentence Compressions (MASC) (Zajic, 2007) is a framework for automatic summarization consisting of three stages: filtering, compression, and candidate selection. This framework has been applied to single and multi-document summarization tasks.

In the first stage (filtering), sentences of high relevance and centrality are selected for further processing. For multi-document summarization, we assume that sentences early on in the documents are more relevant and important than the sentences further down and so, unless otherwise specified, we filter out all sentences except the first five from each document in the cluster.

In the second stage (sentence compression), multiple alternative compressed versions of the source sentences are generated, including a version with no compression, i.e., the original sentence. We have used two different sentence compression approaches with MASC: HMM Hedge (Zajic et al., 2002; Zajic, 2007) and Trimmer (Dorr et al., 2003; Zajic et al., 2006b; Zajic, 2007). HMM Hedge uses a noisy channel model with language models of newspaper stories and newspaper headlines to generate the most probable compressions of a source sentence. Trimmer uses linguistically-motivated trimming rules to remove constituents from a parse tree. Both approaches associate compression-specific feature values with the candidate compressions that can be used in candidate selection. HMM Hedge uses the probability of the compression along with decoding properties, such as the number of clumps of contiguous words from the source sentence. Trimmer uses the number and parse tree depth of various rule applications. The work presented here uses Trimmer as the sentence compression module of MASC.

Trimmer generates multiple compressions by treating the output of each Trimmer rule application as a distinct compression. The output of a Trimmer rule is a single parse tree and an associated surface string. As rules are applied in a pre-determined order, the surface string shrinks until a minimum length is reached, and trimming stops. A limitation associated with this approach is that each rule application is an irrevocable decision. Recent work has introduced Trimmer rules that produce multiple outputs. These Multi-Candidate Rules (MCRs) increase the pool of candidates by having Trimmer processing continue along each MCR output. For example, a MCR for conjunctions generates three outputs: one in which the conjunction and the left child are trimmed, one in which the conjunction and the right child are trimmed, and one in which neither are trimmed. The three outputs of this particular conjunction MCR on the sentence *The program promotes education and fosters community* are: (1) *The program promotes education*, (2) *The program fosters community*, and (3) the original sentence itself. Other MCRs deal with the selection of the root node for the compression and removal of preambles. For a more detailed description of MCRs, see (Zajic, 2007). The Trimmer system used for our submission employed a number of MCRs.

The final stage in MASC is the selection of candidates from the pool created by filtering and compression. We use a linear combination of static and dynamic candidate features to select the highest scoring candidate for inclusion in the summary. Static features include position of the original sentence in the document, length, compression-specific features, and relevance scores. These are calculated prior to the candidate selection process and do not change. Dynamic features include redundancy with respect to the current summary state, and the number of candidates already in the summary from a candidate's source document. The dynamic features have to be recalculated after every candidate selection. In addition, when a candidate is selected, all other candidates derived from the same source sentence are removed from the candidate pool. Candidates are selected for inclusion until either the summary reaches the prescribed word limit or the pool is exhausted. The next section discusses how the weights for the candidate features are determined.

4 Automatic Weight Optimization

In our DUC2006 submission, the weights for the candidate features were optimized manually to maximize the ROUGE-2 recall score (Lin and Hovy, 2003) on the final system output. Note that the system output is actually the output of a series of candidate selections at each iteration of the summary construction algorithm. Therefore, this method of optimization tunes the weights of the candidate features only indirectly. In our current submission, we use a more direct way to optimize the feature weights by automatically tuning the feature weights based on the actual candidates selected at each iteration.

Figure 1 shows how the algorithm works. At each iteration, we generate k-best candidates instead of just the single best. For each of these k candidates, we compute the difference in the ROUGE scores of the current summary state and a hypothetical summary state that would result by adding this candidate to the summary. We refer to this metric as Δ_{ROUGE}. Once Δ_{ROUGE} has been calculated for all k candidates, the summary state is updated by adding the highest scoring candidate and the candidate pool is culled and updated as explained previously. We continue in this fashion until either the word limit is reached or we run out of candidates. The process is then repeated for all other document clusters.

This process generates a large number of candidates—each with its associated features, the Δ_{ROUGE} value, and the number of words in the summary state to which the candidate was (hypothetically) added. Each of these candidates serves as a hypothesis for the optimizer. Note that at each iteration, the summary used in the calculation of Δ_{ROUGE} is the result of adding *only* the 1-best candidate from the previous iteration, as shown in the figure. The k-best candidates are used only to create additional hypotheses for optimization.

The next step is to run an optimizer that takes the following inputs:

1. This list of hypotheses.
2. Initial values for the feature weights.
3. The perturbation range for each feature weight.

The optimizer uses Powell's method (Powell, 1965) to maximize the Δ_{ROUGE} value and outputs the feature weights that can be considered optimized to pick the best candidate to add to the summary state at any point.

For the purposes of comparison, we ran our system on the DUC2007 training data with the old manual optimization technique. For both techniques, DUC2006 data was used as the held out dataset on which the weights were tuned. Table 1 shows the ROUGE scores for both the optimization approaches. The improvements are all significant at $p < 0.05$.

ROUGE	Manual	Automatic
1	0.36394	0.40387
2	0.08131	0.10412
SU4	0.12625	0.15447

Table 1: Comparing ROUGE scores for the automatic optimization technique against previously used manual optimization.

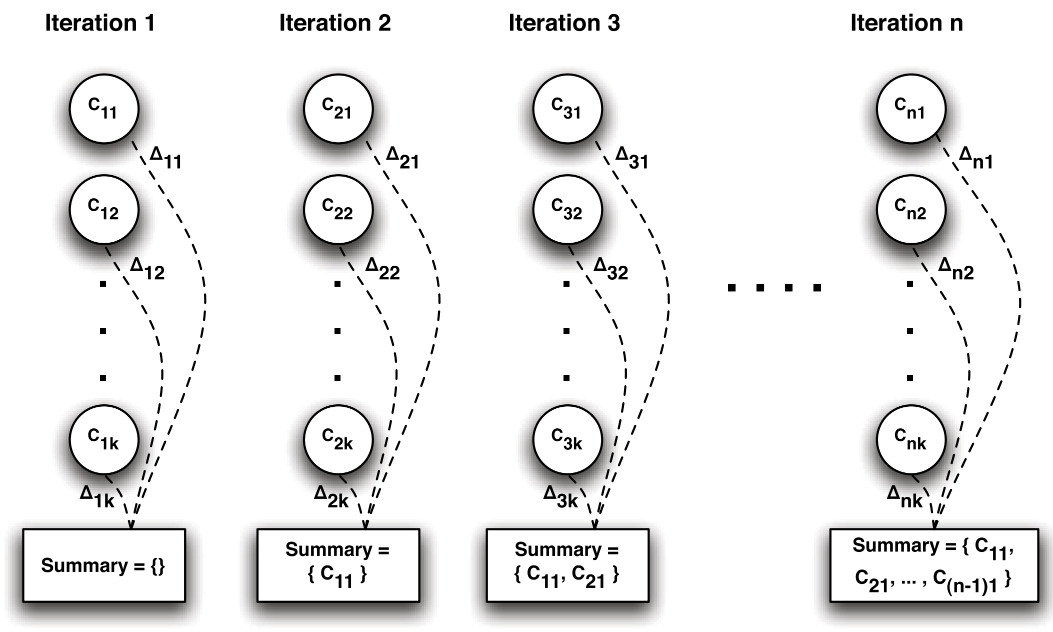

Figure 1: Computing Δ_{ROUGE} values for each of the k-best candidates at each iteration.

5 Redundancy

One of the candidate features used in the MASC framework is the redundancy feature which measures how similar a candidate is to the current summary. If a candidate contains words that occur much more frequently in the summary than in the general language, the candidate is considered to be redundant with respect to the summary. This feature is based on the assumption that the summary is produced by a word distribution which is separate from the word distribution underlying the general language. We use an interpolated probability formulation to measure whether a candidate word w is more likely to have been generated by the summary word distribution than by the word distribution representing the general language:

$$P(w) = \lambda P(w|S) + (1 - \lambda)P(w|L) \qquad (1)$$

where S is text representing the summary and L is text representing the general language.[1] As a general estimate of the portion of words in a text that are specific to the text's topic, λ was set to 0.3. The conditional probabilities are estimated using maximum likelihood:

$$P(w|S) = \frac{\text{Count of } w \text{ in } S}{\text{number of words in } S}$$

$$P(w|L) = \frac{\text{Count of } w \text{ in } L}{\text{number of words in } L}$$

Assuming the words in a candidate to be independently and identically distributed, the probability of the entire candidate C, i.e., the value of the redundancy feature, is given by:

$$P(C) = \prod_{i=1}^{N} P(w_i)$$

$$= \prod_{i=1}^{N} \lambda P(w_i|S) + (1 - \lambda)P(w_i|L)$$

We use log probabilities for ease of computation:

$$= \sum_{i=1}^{N} \log(\lambda P(w_i|S) + (1 - \lambda)P(w_i|L))$$

Note that redundancy is a *dynamic* feature because the word distribution underlying the summary changes each time a new candidate is added to it.

[1]The documents in the cluster being summarized are used to estimate the general language model.

5.1 Adaptation for Update Task

The interpolated redundancy feature can quantitatively indicate whether the current candidate is more like the sentences in other documents (non-redundant) or more like sentences in the current summary state (redundant). We adapted this feature for the update task to indicate whether a candidate was more like text from novel documents (update information) or from previously-read documents (not update information). This adaption was straightforward—for each of the three given document clusters, we added the documents that are assumed to have been previously read by the user to S from equation 1. Since S represents content already included in the summary, any candidate with content from the already-read documents is automatically considered redundant by our system.

5.2 Using paraphrases

The redundancy feature essentially treats the summary text S as a bag-of-words to compute the maximum likelihood estimate for $P(w|s)$ used in equation 1. However, this feature would not adequately capture redundancy in cases where the candidate word is a synonym or a paraphrase of a word already in the summary. Indeed, such words would be classified as non-redundant due to the specific nature of a bag-of-words match. We attempt to improve this feature by adding a second bag-of-words—one that contains paraphrases of the words in S. During the redundancy feature computation, this second bag is consulted only when a candidate word does not match any of the words in S.

The paraphrases are extracted statistically from a $850,000$-sentence Chinese-English parallel corpus following Bannard et al. (2005). For the purposes of this experiment, we choose only the one word paraphrases from the larger list of paraphrases that the technique is able to generate.

This technique has not yet yielded significant improvements in ROUGE scores. We investigate this further and present a more detailed analysis in Section 6.2.

6 Post-hoc Experiments

We also conducted some post-hoc experiments in order to test the usefulness of some ideas after the DUC 2007 model summaries were officially released.

6.1 Sentence Filtering

Previous work (Zajic et al., 2006a) has shown that selecting only the first few sentences of a document for the purposes of syntactic trimming usually yields summaries of higher quality. The MASC framework currently only considers candidates derived from the first five sentences from any source document. We conducted a series of experiments in order to determine if there is any further performance gain to be had by making stricter restrictions on the position of the sentence from which the candidate was derived.

We first added a single additional parameter N_s to the filtering stage to allow through only the first N_s sentences of each document for compression and selection. It is reasonable to assume that if sentence filtering can have an effect on the quality of the produced summary, it can also affect the weights that were produced by the automatic optimization technique and subsequently used for candidate selection. To measure this effect, if any, we introduced another parameter N_o, which restricts the candidates used by the optimizer to only those derived from the first N_o sentences of each document in the held out dataset (DUC2006 test data and reference summaries).

We varied both N_s and N_o from 1 to 5 to measure the interaction between the two parameters and calculated the ROUGE-2 scores for the resulting summaries for the DUC 2007 main task. Table 2 shows the results of these experiments. The (i,j)-th entry in the table corresponds to the experiment where (a) only candidates derived from the first i sentences in each document in the held out set were used to optimize the feature weights and (b) only candidates derived from the first j sentences in each document in the DUC2007 data were chosen for inclusion in the summary.

We find that almost none of the differences between any two experiments is statistically significant. However, in the few cases where the differences are significant, it is always the case that using a larger number of sentences yields better summaries. It is entirely possible, however, that a more theoretically grounded sentence-filtering approach (Conroy et al., 2006), where the filtering is based not just on the position of the sentence in the

	$\mathbf{N_s = 1}$	$\mathbf{N_s = 2}$	$\mathbf{N_s = 3}$	$\mathbf{N_s = 4}$	$\mathbf{N_s = 5}$
$\mathbf{N_o = 1}$	0.09904	0.09164	0.08803	0.08818	0.08807
$\mathbf{N_o = 2}$	0.10327	0.09758	0.09812	0.09863	0.09842
$\mathbf{N_o = 3}$	0.10675	0.10509	0.10515	0.10596	0.10640
$\mathbf{N_o = 4}$	0.10473	0.09963	0.10016	0.10078	0.09999
$\mathbf{N_o = 5}$	0.10476	0.10634	0.10634	0.10634	0.10675

Table 2: ROUGE-2 scores for various sentence filtering experiments on DUC2007 data.

document but also on the quality of the compressed candidate that can be generated from that sentence, might be more effective at selecting the right subset of sentences.

6.2 Using WordNet for Paraphrases

Our previous experiment to enhance redundancy capturing using statistically extracted single-word paraphrases did not yield any significant improvements. The paraphrasing technique we used has high coverage but relatively low precision. In order to test whether it is the quality of the paraphrases that limits their usefulness, we conducted a similar experiment in which we used WordNet to generate the synonym for each word in the summary bag-of-words.

To generate the synonym for any word, we first locate the most frequent synset for this word in each of the four WordNet databases (N, V, ADJ, ADV). We then collect all single word synonyms from all these synsets and choose one at random. Synonyms for all words in the summary text are generated and collected as a second bag-of-words that is used in the same fashion as described in Section 5.2. Although, we do have access to the part-of-speech tag for each word in the summary text and we could have used it to consult only the "correct" WordNet database, it turns out not to matter at all as explained below.

We discovered that neither of the WordNet-based experiments yield any improvements. We further investigated the cause of this behavior, which seemed to be unrelated to the quality of the paraphrases used and found that the anomalous cases for which we had decided to create a second of bag-of-words actually occur quite rarely. Most of the time, the original summary bag-of-words is sufficient to capture the redundancy present in a candidate. We believe this to be a result of the tightness of the document clusters provided to us.

ROUGE	**Avg. Recall**	**Avg. Precision**	**Avg. F**
1	0.41802 (9)	0.40691 (20)	0.41224 (13)
2	0.10795 (8)	0.10516 (11)	0.10650 (8)
3	0.03972 (7)	0.03869 (9)	0.03919 (8)
4	0.01963 (8)	0.01913 (8)	0.01937 (8)
L	0.38289 (10)	0.37270 (19)	0.37758 (13)
W-1.2	0.11102 (11)	0.20055 (18)	0.14284 (12)
SU4	0.15990 (9)	0.15563 (14)	0.15767 (11)

Table 3: ROUGE scores for MASC (System 7) on the main task with ranks out of 30 systems.

7 Results and Future Work

For DUC 2007, we participated in both the main summarization task and the pilot update-summarization task. In this section, we first present the results of our system on both the tasks and then we discuss some ideas to improve our system for future submissions.

7.1 Main Task Results

The DUC2007 main task modeled real-world question answering and required the participants to generate a fluent, well-organized 250-word summary for 45 document clusters, each containing 25 documents relevant to a given topic query. The weights for the candidates were automatically optimized (using the technique described in Section 4) on the DUC2006 test data, using the DUC2006 reference summaries. In the DUC2007 evaluations, the UMD-MASC system was System 7. Table 3 shows the ROUGE scores for MASC on the DUC2007 data with ranks out of 30 submitted systems. MASC generally ranked higher for recall than for precision, suggesting that MASC is currently better at finding relevant content than it is at weeding out irrelevant content.

The DUC2007 evaluation also included human judgments of linguistic quality which are shown in

Question	Avg. Score	Rank
Grammaticality	3.22	22
Non-Redundancy	3.33	27
Referential Clarity	3.98	2
Focus	3.42	14
Structure & Coherence	2.27	18

Table 4: Linguistic scores for MASC (System 7) on the main task with ranks out of 30 systems.

Table 4, along with the ranks.

We have improved our grammaticality and referential clarity scores since last year. This might suggest that the candidate selector is simply not picking the compressed versions of the sentences for inclusion. We plan to conduct a more thorough analysis of this behavior before the next submission. The human assessors also assigned a content responsiveness score to each summary. MASC achieved an average responsiveness score of 3.089 and was ranked 4^{th} out of the 30 submitted systems. This further confirms our hypothesis that MASC is much better at finding content relevant to a given query.

7.2 Update Task Results

The DUC2007 update summary pilot task was to create short 100-word multi-document summaries under the assumption that the reader has already read some number of previous documents. There were 10 topics in the test data, with 25 documents to each topic. For each topic, the documents were sorted in chronological order and then partitioned into three sets $A - C$. The participants were then required to generate (a) a summary for cluster A, (b) an update summary for cluster B assuming documents in A have already been read, and (c) an update summary for cluster C assuming documents in A and B have already been read. The system we used for the update task was identical to the one used for the main task, except for the adaptation of the redundancy feature as described in Section 5.1.

In the DUC2007 evaluations, the UMD-MASC system was System 36 for the update task. Table 5 shows a subset of the ROUGE scores and ranks for MASC.

Table 6 shows the average content responsiveness scores for MASC. These scores indicate that although the adapted redundancy feature may not be very effective for a large set of already-read documents, it does its job extremely well for smaller sets—as evident by the high rank MASC achieved.

ROUGE	Avg. Recall	Avg. Precision	Avg. F
1	0.35352 (7)	0.34266 (9)	0.34754 (7)
2	0.08762 (7)	0.08462 (10)	0.08595 (9)
SU4	0.12602 (8)	0.12182 (10)	0.12369 (8)

Table 5: ROUGE scores for MASC (System 36) on the update task with ranks out of 22 systems.

	Score	Rank
Cluster A	3.1	2
Cluster B	2.7	5
Cluster C	2.6	6
Overall	2.8	2

Table 6: Content responsiveness scores for MASC (System 36) on the update task with ranks out of 22 systems.

7.3 Future Work

There are a large number of things that we plan to try in future DUC submissions. We believe that the redundancy feature can be better formulated in a number of ways:

1. We plan to experiment with splitting this feature into two separate features—one that captures redundancy and another that uses an n-gram language model to capture non-redundancy. Once split, the weights for the two resulting features can be tuned automatically rather than using a manually estimated interpolation coefficient λ.

2. Since this is the only feature in the framework to be computed on a logarithmic scale, it is possible that even if there was a significant difference in the computed value, it would be drowned out by less significant differences in the values of the other features that are not computed on a logarithmic scale.

We also plan to improve the syntactic trimming process by making it fully order-independent. Both of our initial paraphrase-based redundancy experiments had drawbacks: (1) The statistically extracted paraphrases have high coverage but low precision, and (2) the WordNet-derived paraphrases do not allow for domain specificity. In a perfect experiment, we would want to use paraphrases that have both high-coverage and are tunable to the particular domain. We plan to continue to experiment with the automatically acquired paraphrases to create additional variants of the redundancy feature. In addition, we hope to be able to scale up to using phrase-level paraphrases rather than limiting ourselves to just the single-word ones.

Acknowledgments

This work has been supported, in part, under the GALE program of the Defense Advanced Research Projects Agency, Contract No. HR0011-06-2-0001, the TIDES program of the Defense Advanced Research Projects Agency, BBNT Contract No. 9500006806, and the University of Maryland Joint Institute for Knowledge Discovery. Any opinions, findings, conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of DARPA.

References

Michele Banko, Vibhu Mittal, and Michael Witbrock. 2000. Headline generation based on statistical translation. In *Proceedings of ACL*, pages 318–325, Hong Kong.

Colin Bannard and Chris Callison-Burch. 2005. Paraphrasing with bilingual parallel corpora. In *Proceedings of ACL*, pages 597–604.

Jaime Carbonell and Jade Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of SIGIR*, pages 335–336, Melbourne, Australia.

John M. Conroy, Judith D. Schlesinger, Dianne P. O'Leary, and J. Goldstein. 2006. Back to basics: CLASSY 2006. In *Proceedings of the NAACL-2006 Document Understanding Conference Workshop*, New York, New York.

Hal Daumé and Daniel Marcu. 2005. Bayesian multi-document summarization at MSE. In *Proceedings of ACL 2005 Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization*.

Bonnie J. Dorr, David M. Zajic, and Richard Schwartz. 2003. Hedge Trimmer: A parse-and-trim approach to headline generation. In *Proceedings of the HLT-NAACL 2003 Text Summarization Workshop and Document Understanding Conference (DUC 2003)*, pages 1–8, Edmonton, Alberta.

Güneş Erkan and Dragomir R. Radev. 2004. The university of michigan at duc2004. In *Proceedings of Document Understanding Conference Workshop*, pages 120–127.

Jade Goldstein, Vibhu Mittal, Jaime Carbonell, and Mark Kantrowitz. 2000. Multi-document summarization by sentence extraction. In *Proceedings of the ANLP/NAACL Workshop on Automatic Summarization*, pages 40–48.

Martin Hassel and Jonas Sjöbergh. 2006. Towards holistic summarization – selecting summaries, not sentences. In *Proceedings of LREC*, Genoa, Italy.

Hongyan Jing. 2000. Sentence reduction for automatic text summarization. In *Proceedings of ANLP*.

Kevin Knight and Daniel Marcu. 2002. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, 139(1):91–107.

Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of HLT/NAACL*, pages 71–78, Edmonton, Alberta.

Gabor Melli, Zhongmin Shi, Yang Wang, Yudong Liu, Annop Sarkar, and Fred Popowich. 2006. Description of SQUASH, the SFU question answering summary handler for the DUC-2006 summarization task. In *Proceedings of DUC*.

Michael J.D. Powell. 1965. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The Computer Journal*, 7:155–162.

Dragomir R. Radev, Hongyan Jing, Malgorzata Styś, and Daniel Tam. 2004. Centroid-based summarization of multiple documents. In *Information Processing and Management*, volume 40, pages 919–938.

Jenine Turner and Eugene Charniak. 2005. Supervised and unsupervised learning for sentence compression. In *Proceedings of ACL 2005*, pages 290–297, Ann Arbor, Michigan.

Lucy Vanderwende, Hisami Suzuki, and Chris Brockett. 2006. Microsoft Research at DUC2006: Task-focused summarization with sentence simplification and lexical expansion. In *Proceedings of DUC*, pages 70–77.

David M. Zajic, Bonnie J. Dorr, and Richard Schwartz. 2002. Automatic headline generation for newspaper stories. In *Proceedings of the ACL-2002 Workshop on Text Summarization (DUC 2002)*, pages 78–85.

D. M. Zajic, J. Conroy, B. J. Dorr, J. Lin, D. P. O'Leary, and J. Schlesinger. 2006a. Sentence trimming and selection: Mixing and matching. In *Proceedings of DUC*.

D. M. Zajic, B. J. Dorr, J. Lin, and R. Schwartz. 2006b. Sentence compression as a component of a multi-document summarization system. In *Proceedings of DUC*.

D. M. Zajic, B. J. Dorr, J. Lin, and R. Schwartz. 2007. Multi-candidate reduction: Sentence compression as a tool for document summarization tasks. *Information Processing and Management Special Issue on Summarization*, 43.

David M. Zajic. 2007. *Multiple Alternative Sentence Compressions (MASC) as a Tool for Automatic Summarization Tasks*. Ph.D. thesis, University of Maryland, College Park.