Multi-Candidate Reduction: Sentence Compression as a Tool for Document Summarization Tasks*

David Zajic[1], Bonnie J. Dorr[1], Jimmy Lin[1], Richard Schwartz[2]

[1]University of Maryland
College Park, Maryland, USA
dmzajic@cs.umd.edu, bonnie@umiacs.umd.edu, jimmylin@umd.edu

[2]BBN Technologies
9861 Broken Land Parkway
Columbia, MD 21046
schwartz@bbn.com

Abstract

This article examines the application of two single-document sentence compression techniques to the problem of multi-document summarization—a "parse-and-trim" approach and a statistical noisy-channel approach. We introduce the Multi-Candidate Reduction (MCR) framework for multi-document summarization, in which many compressed candidates are generated for each source sentence. These candidates are then selected for inclusion in the final summary based on a combination of static and dynamic features. Evaluations demonstrate that sentence compression is a valuable component of a larger multi-document summarization framework.

Keywords: headline generation, summarization, parse-and-trim, Hidden Markov Model

PACS: Artificial intelligence, 07.05.Mh; Computer science and technology, 89.20.Ff; Spoken languages, processing of, 43.71.Sy

1 Introduction

This article presents an application of two different single-document sentence compression methods to the problem of multi-document summarization. The first, a "parse-and-trim" approach, has been implemented in a system called Trimmer and its extended version called Topiary. The second, an HMM-based approach, has been implemented in a system called HMM Hedge.

These systems share the basic premise that a textual summary can be constructed by selecting a subset of words, in order, from the original text, with morphological variation allowed for some word classes. Trimmer selects subsequences of words using a linguistically-motivated algorithm to trim syntactic constituents from sentences until a desired length has been reached. In the context of

*Please cite as: David Zajic, Bonnie Dorr, Jimmy Lin, and Richard Schwartz. Multi-Candidate Reduction: Sentence Compression as a Tool for Document Summarization Tasks. *Information Processing and Management*, 43(6):1549-1570, 2007. [DOI: 10.1016/j.ipm.2007.01.016] This is the pre-print version of a published article. Citations to and quotations from this work should reference that publication. If you cite this work, please check that the published form contains precisely the material to which you intend to refer. (Received 18 July 2006; revised 3 January 2007; accepted 8 January 2007) This document prepared August 31, 2007, and may have minor differences from the published version.

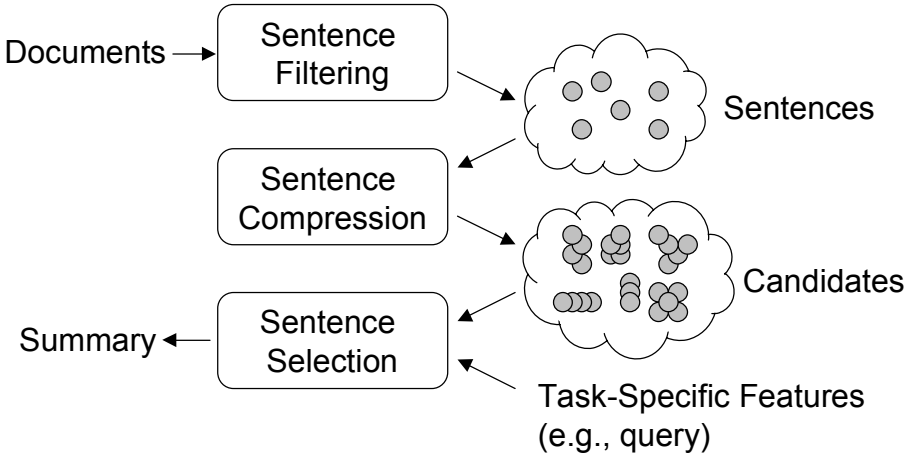

Figure 1: The Multi-Candidate Reduction (MCR) framework.

very short single-document summarization, or headline generation, this approach is applied to the lead sentence (the first non-trivial sentence) in order to produce a short headline tens of characters long. HMM Hedge (Hidden Markov Model HEaDline GEnerator), on the other hand, is a statistical headline generation system that finds the subsequence of words most likely to be a headline for a given story. This approach is similar to techniques used in statistical machine translation in that the observed story is treated as the garbled version of an unseen headline transmitted through a noisy channel. In their original configurations, both Trimmer and HMM Hedge create sentence compressions that mimic the style of Headlinese, a form of compressed English associated with newspaper headlines (Mårdh, 1980).

The goal of our work is to transition from a single-document summarization task with sentence-level character-based length constraint to a multi-document summarization task with summary-level word-based length constraint. In headline generation, summaries should fill as much of the available space as possible, without going over the limit. Headlines that fall below the length limit miss the opportunity to include additional information. Headlines that exceed the length limit must be truncated, resulting in ungrammatical sentences and loss of information. When the length constraint is applied to a collection of sentences instead of a single sentence, concerns about individual sentence length become less important. A short sentence does not result in wasted space and it is acceptable to include a lengthy sentence as long as it includes important, non-redundant information.

To transition from single-document to multi-document summarization, we examined different combinations of possible sentence compressions to construct the best summary. A key innovation of this work—tested for the first time in 2005 at the Document Understanding Conference (DUC-2005) (Zajic et al., 2005b) and Multilingual Summarization Evaluation (MSE-2005) (Zajic et al., 2005a)—is the use of multiple compressed candidates for sentences in the source documents. Sentence compression is used for different purposes in single- and multi-document summarization tasks. In the first case, the goal is to fit long sentences into the available space while preserving important information. In multi-document summarization, sentence compression is used to generate multiple candidates that capture relevant, non-redundant information. Sentence selection algorithms can then be applied to determine which compressed candidates provide the best combination of topic coverage and brevity.

We introduce a framework for extractive multi-document summarization called Multi-Candidate Reduction (MCR), whose architecture is shown in Figure 1. Sentence compression techniques are employed in the intermediate stage of the processing pipeline to generate multiple compressed variants of source sentences. A sentence selector then builds the final summary by choosing among these candidates, based on features propagated from the sentence compression method, features of the candidates

themselves, and features of the present summary state. Under this framework, our previously-developed systems, Trimmer and HMM Hedge, are employed to generate the compressed candidates. The MCR architecture also includes a filtering module that chooses the source sentences from which compressed candidates are generated. This work does not examine the filtering process in detail, as we only employ very simple approaches, e.g., retain first n sentences from each document.

Sentence compression supports extractive multi-document summarization by reducing the length of summary candidates while preserving their relevant content, thus allowing space for the inclusion of additional material. However, given the interaction of relevance and redundancy in a multi-sentence summary, it is unlikely that a single algorithm or scoring metric can provide the "best" compression of a sentence. This is the motivation for MCR, which provides several alternative candidates for each source sentence. Subsequent processes can then select among many alternatives when constructing the final summary.

This article is organized as follows: The next section relates our approach to other existing summarization systems. In Section 3, we describe Trimmer, a variant of Trimmer called Topiary, and HMM Hedge. These systems implement two fundamentally different approaches to sentence compression. Section 4 describes how these tools can be applied to a multi-document summarization task. Section 5 describes evaluations of our framework. Our systems have also been applied to several different types of texts; some of these applications will be briefly covered in Section 6. Finally, we propose future work in the area before concluding.

2 Related Work

We have developed a "parse-and-trim" approach to sentence compression, implemented in a system called Trimmer (Dorr et al., 2003b). The system generates a headline for a news story by compressing the *lead* (or main) topic sentence according to a linguistically-motivated algorithm that operates on syntactic structures. Syntactic approaches to compression have been used in single-document summarization systems such as Cut-and-Paste (Jing and McKeown, 2000) and also in multi-document summarization systems such as SC (Blair-Goldensohn et al., 2004) and CLASSY (Conroy et al., 2005; Conroy et al., 2006). The SC system pre-processes input to remove appositives and relative clauses. CLASSY uses an HMM sentence selection approach combined with a conservative sentence compression method based on shallow parsing to detect lexical cues to trigger phrase eliminations. The approach taken by Trimmer is most similar to that of Jing and McKeown (2000), in which algorithms for removing syntactic constituents from sentences are informed by analysis of human summarizers. Trimmer differs from these systems in that multiple compressed candidates of each sentence are generated. The potential of multiple alternative compressions has also been explored by Vanderwende et al. (2006).

Summaries may also contain lists of words or short phrases that denote important topics or concepts in the document. In particular, extractive topic summaries consist of keywords or key phrases that occur in the document. We have built Topiary, an extension to Trimmer, that constructs headlines combining compressed sentences with topic terms. This approach is similar to the work of Euler (2002), except that Euler uses topic lists to guide sentence selection and compression toward a query-specific summary, whereas Topiary uses topics to augment the concept coverage of a generic summary. Our system was demonstrated to be the top-performing single-document summarization system in the DUC-2004 evaluation (Zajic et al., 2004).

In contrast to Topiary, which combines sentence compression with topic terms, others have constructed summaries directly from topic terms. For example, Bergler et al. (2003) choose noun phrases that represent the most important entities as determined by noun phrase coreference chains. Wang et al. (2005) propose a baseline system that constructs headlines from topic descriptors identified using term frequency counts; this system was reported to outperform LexTrim, their Topiary-style system. Zhou and Hovy (2003) construct fluent summaries from a topic list by finding phrase clusters early in

the text that contain important topic words found throughout the text.

The task of assigning topic terms to documents is related to text categorization, in which documents are assigned to pre-defined categories. The categories can be labeled with topic terms, so that the decision to put a document in a category is equivalent to assigning that category's label to the document. Assigning topic terms to documents by categorization permits the assignment of terms that do not occur in the document. Lewis (1999) describes a probabilistic feature-based method for assigning Reuters topics to news stories. Along these lines, OnTopic™ (Schwartz et al., 1997) uses an HMM to assign topics to a document.

In addition to Trimmer, we have implemented HMM Hedge (Hidden Markov Model HEaDline GEnerator), a statistical headline generation system that finds the most likely headline for a given story. This approach is similar to techniques used in statistical machine translation in that the observed story is treated as the garbled version of an unseen headline transmitted through a noisy channel. The noisy-channel approach has been used for a wide range of Natural Language Processing (NLP) applications including speech recognition (Bahl et al., 1983); machine translation (Brown et al., 1990); spelling correction (Mays et al., 1990); language identification (Dunning, 1994); and part-of-speech tagging (Cutting et al., 1992). Of those who have taken a noisy-channel approach to sentence compression for text summarization Banko et al. (2000), Knight and Marcu (2000), Knight and Marcu (2002), and Turner and Charniak (2005) have used the technique to find the most probable short string that generated the observed full sentence.

Like other summarization systems based on the noisy-channel model, HMM Hedge treats the observed data (the story) as the result of unobserved data (headlines) that have been distorted by transmission through a noisy channel. The effect of the noisy channel is to add story words between the headline words. Our approach differs from Knight and Marcu (2000), Banko et al. (2000), and Turner and Charniak (2005), in that HMM Hedge does not require a corpus of paired stories and summaries. HMM Hedge uses distinct language models of news stories and headlines, but does not require explicit pairings of stories and summaries.

To transition our single-sentence summarization techniques to the problem of multi-document summarization, we must consider how to select candidates for inclusion in the final summary. A common approach is to rank candidate sentences according to a set of features and iteratively build the summary, appropriately re-ranking the candidates at each step to avoid redundancy. MEAD (Radev et al., 2004) scores source sentences according to a linear combination of features including centroid, position, and first-sentence overlap. These scores are then refined to consider cross-sentence dependencies, such as redundancy, chronological order, and source preferences. MCR differs in that multiple variants of the same source sentences are available as candidates for inclusion in the final summary.

Minimization of redundancy is an important element of a multi-document summarization system. Carbonell and Goldstein (1998) propose a technique called Maximal Marginal Relevance (MMR) for ranking documents returned by an information retrieval system so that the front of the ranked list will contain diversity as well as high relevance. Goldstein et al. (2000) extend MMR to multi-document summarization. MCR borrows the ranking approach of MMR, but uses a different set of features. Like MEAD, these approaches use feature weights that are optimized to maximize an automatic metric on training data.

Several researchers have shown the importance of summarization in domains other than written news (Muresan et al., 2001; Clarke and Lapata, 2006). Within the MCR framework, we discuss the portability of Trimmer and HMM Hedge to a variety of different texts: written news, broadcast news transcriptions, email threads, and text in foreign language.

3 Single-Sentence Compression

Our general approach to the generation of a summary from a single document is to produce a *headline* by selecting words in order from the text of the story. Consider the following excerpt from a news story and corresponding headline:

(1) (i) After months of debate following the Sept. 11 terrorist hijackings, the Transportation Department has decided that airline **pilots** will **not** be **allowed to have guns in** the **cockpits**.

(ii) Pilots not allowed to have guns in cockpits.

The bold words in (1i) form a fluent and accurate headline, as shown in (1ii).

This basic approach has been realized in two ways. The first, Trimmer, uses a linguistically-motivated algorithm to remove grammatical constituents from the lead sentence until a length threshold is met. Topiary is a variant of Trimmer that combines fluent text from a compressed sentence with topic terms to produce headlines. The second, HMM Hedge, employs a noisy-channel model to find the most likely headline for a given story. The remainder of this section will present Trimmer, Topiary, and HMM Hedge in more detail.

3.1 Trimmer

Our first approach to sentence compression involves iteratively removing grammatical constituents from the parse tree of a sentence using linguistically-motivated rules until a length threshold has been met. When applied to the lead sentence, or first non-trivial sentence of a story, our algorithm generates a very short summary, or headline. This idea is implemented in our Trimmer system, which can leverage the output of any constituency parser that uses the Penn Treebank conventions. At present we use Charniak's parser (Charniak, 2000).

The insights that form the basis and justification for the Trimmer rules come from our previous study, which compared the relative prevalence of certain constructions in human-written summaries and lead sentences in stories. This study used 218 human-written summaries of 73 documents from the TIPSTER corpus (Harman and Liberman, 1993) dated January 1, 1989. The 218 summaries and the lead sentences of the 73 stories were parsed using the BBN SIFT parser (Miller et al., 2000). The parser produced 957 noun phrases (NP nodes in the parse trees) and 315 clauses (S nodes in the parse trees) for the 218 summaries. For the 73 lead sentences, the parser produced 817 noun phrases and 316 clauses.

At each level (sentence, clause, and noun phrase), different types of linguistic phenomena were counted.

- At the sentence level, the numbers of preposed adjuncts, conjoined clauses, and conjoined verb phrases were counted. Children of the root S node that occur to the left of the first NP are considered to be preposed adjuncts. The bracketed phase in "[According to police] the crime rate has gone down" is a prototypical example of a preposed adjunct.
- At the clause level, temporal expressions, trailing SBAR nodes, and trailing PP nodes were counted. Trailing constituents are those not designated as an argument of a verb phrase.
- At both the sentence and clause levels, conjoined S nodes and conjoined VP nodes were counted.
- At the NP level, determiners and relative clauses were counted.

The counts and prevalence of the phenomena in the human-generated headlines and lead sentences are shown in Table 1. The results of this analysis illuminated the opportunities for trimming constituents and guided the development of our Trimmer rules, detailed below.

Level	Phenomenon	Summary		Lead Sentence	
Sentence	preposed adjuncts	0/218	0%	2/73	2.7%
	conjoined S	1/218	0.5%	3/73	4%
	conjoined VP	7/218	3%	20/73	27%
Clause	temporal expression	5/315	1.5%	77/316	24%
	trailing PP	165/315	52%	184/316	58%
	trailing SBAR	24/315	8%	49/316	16%
Noun Phrase	relative clause	3/957	0.3%	29/817	3.5%
	determiner	31/957	3%	205/817	25%

Table 1: Counts and prevalence of phenomena found in summaries and lead sentences.

3.1.1 Trimmer Algorithm

Trimmer applies syntactic compression rules to a parse tree according the following algorithm:

1. Remove temporal expressions
2. Select Root S node
3. Remove preposed adjuncts
4. Remove some determiners
5. Remove conjunctions
6. Remove modal verbs
7. Remove complementizer *that*
8. Apply the XP over XP rule
9. Remove PPs that do not contain named entities
10. Remove all PPs under SBARs
11. Remove SBARSs
12. Backtrack to state before Step 9
13. Remove SBARs
14. Remove PPs that do not contain named entities
15. Remove all PPs

Steps 1 and 4 of the algorithm remove low-content units from the parse tree.

Temporal expressions—although certainly not content-free—are not usually vital for summarizing the content of an article. Since the goal is to provide an informative headline, the identification and elimination of temporal expressions (Step 1) allow other more important details to remain in the length-constrained headline. The use of BBN's IdentiFinder™ (Bikel et al., 1999) for removal of temporal expressions is described in Section 3.1.2.

The determiner rule (Step 4) removes leaf nodes that are assigned the part-of-speech tag DT and have the surface form *the*, *a*, or *an*. The intuition for this rule is that the information carried by

articles is expendable in summaries, even though this makes the summaries ungrammatical for general English. Omitting articles is one of the most salient features of newspaper headlines. Sentences (2) and (3), taken from the New York Times website on September 27, 2006, illustrate this phenomenon. The italicized articles did not occur in the actual newspaper headlines.

(2) *The* Gotti Case Ends With *a* Mistrial for *the* Third Time in a Year

(3) *A* Texas Case Involving Marital Counseling Is *the* Latest to Test *the* Line Between Church and State

Step 2 identifies nodes in the parse tree of a sentence that could serve as the root of a compression for the sentence. Such nodes will be referred to as Root S nodes. A node in a tree is a Root S node if it is labeled S in the parse tree and has children labeled NP and VP, in that order. The human-generated headlines we studied always conform to this rule. It has been adopted as a constraint in the Trimmer algorithm that the lowest leftmost Root S node is taken to be the root node of the headline. An example of this rule application is shown in (4). The boldfaced material in the parse is retained and the italicized material is eliminated.

(4) (i) Input: Rebels agreed to talks with government officials, international observers said Tuesday.

(ii) Parse: *[S* **[S [NP Rebels][VP agreed to talks with government officials]]***, international observers said Tuesday.]*

(iii) Output: Rebels agreed to talks with government officials.

When the parser produces a usable parse tree, this rule selects a valid starting point for compression. However, the parser sometimes produces incorrect output, as in the cases below (from DUC-2003):

(5) (i) Parse: **[S[SBAR What started as a local controversy][VP has evolved into an international scandal.]]**

(ii) Parse: **[NP[NP Bangladesh][CC and][NP[NP India][VP signed a water sharing accord.]]]**

In (5i), an S exists, but it does not conform to the requirements of the Root S rule because it does not have as children an NP followed by a VP. The problem is resolved by selecting the lowest leftmost S, ignoring the constraints on the children. In (5ii), no S is present in the parse. This problem is resolved by selecting the root of the entire parse tree as the root of the headline. These parsing errors occur infrequently—only 6% of the sentences in the DUC-2003 evaluation data exhibit these problems, based on parses generated by the BBN SIFT parser.

The motivation for removing preposed adjuncts (Step 3) is that all of the human-generated headlines omit what we refer to as the *preamble* of the sentence. Preposed adjuncts are constituents that precede the first NP (the subject) under the Root S chosen in Step 2; the preamble of a sentence consists of its preposed adjuncts. The impact of preposed adjunct removal can be seen in example (6).

(6) (i) Input: According to a now finalized blueprint described by U.S. officials and other sources, the Bush administration plans to take complete, unilateral control of a post-Saddam Hussein Iraq.

(ii) Parse: **[S***[PP According to a now finalized blueprint described by U.S. officials and other sources], [Det the]* **Bush administration plans to take complete, unilateral control of***[Det a]* **post-Saddam Hussein Iraq.]**

(iii) Output: Bush administration plans to take complete unilateral control of post-Saddam Hussein Iraq.

The remaining steps of the algorithm remove linguistically peripheral material through successive deletions of constituents until the sentence is shorter than a length threshold. Each stage of the algorithm corresponds to the application of one of the rules. Trimmer first finds the pool of nodes in the parse to which a rule can be applied. The rule is then iteratively applied to the deepest, rightmost remaining node in the pool until the length threshold is reached or the pool is exhausted. After a rule has been applied at all possible nodes in the parse tree, the algorithm moves to the next step.

In the case of a conjunction with two children (Step 5), one of the children will be removed. If the conjunction is *and*, the second child is removed. If the conjunction is *but*, the first child is removed. This rule is illustrated by the following examples, where the italicized text is trimmed.

(7) When Sotheby's sold a Harry S Truman signature that turned out to be a reproduction, the prestigious auction house apologized *and bought it back*.

(8) President Clinton *expressed sympathy after a car-bomb explosion in a Jerusalem market wounded 24 people but* said the attack should not derail the recent land-for-security deal between Israel and the Palestinians.

The modal verb rule (Step 6) applies to verb phrases in which the head is a modal verb and the head of the child verb phrase is a form of *have* or *be*. In such cases, the modal and auxiliary verbs are removed. Sentences (9) and (10) show examples of this rule application. Note that although in Sentence (10) the omission of trimmed material changes the meaning, given a tight space constraint, the loss of the modality is preferable to the loss of other content information.

(9) People's palms and fingerprints *may be* used to diagnose schizophrenia.

(10) Agents *may have* fired potentially flammable tear gas cannisters.

The complementizer rule (Step 7) removes the word *that* when it occurs as a complementizer. Sentence (11) shows an example in which two complementizers can be removed.

(11) Hoffman stressed *that* study is only preliminary and can't prove *that* treatment useful.

The XP-over-XP rule (Step 8) is a linguistic generalization that allows a single rule to cover two different phenomena. XP in the name of the rule is a variable that can take two values: NP and VP. In constructions of the form [XP [XP ...] ...], the other children of the higher XP are removed. Note that the child XP must be the first child of the parent XP. When XP = NP the rule removes relative clauses (as in Sentence (12)) and appositives (as in Sentence (13)).

(12) Schizophrenia patients *whose medication couldn't stop the imaginary voices in their heads* gained some relief.

(13) A team led by Dr. Linda Brzustowicz*, assistant professor of neuroscience at Rutgers University's Center for Molecular and Behavioral Neuroscience in Newark,* studied DNA of dozens of members of 22 families.

The rules that remove prepositional phrases and subordinate clauses (Steps 9 through 15) are sometimes prone to removing important content. Thus, these rules are applied last, only when there are no other types of rules to apply. Moreover, these rules are applied with a backoff option to avoid

over-trimming the parse tree. First, the PP rule is applied (Steps 9 and 10),[1] followed by the SBAR rule (Step 11). If the desired sentence length has not been reached, the system reverts to the parse tree as it was before any prepositional phrases were removed (Step 12) and applies the SBAR rule (Step 13). If the desired length still has not been reached, the PP rule is applied again (Steps 14 and 15).

The intuition behind this ordering is that, when removing constituents from a parse tree, it is preferable to remove smaller fragments before larger ones and prepositional phrases tend to be smaller than subordinate clauses. Thus, Trimmer first attempts to achieve the desired length by removing smaller constituents (PPs), but if this cannot be accomplished, the system restores the smaller constituents, removes a larger constituent, and then resumes the deletion of the smaller constituents. To reduce the risk of removing prepositional phrases that contain important information, BBN's IdentiFinder is used to distinguish PPs containing temporal expressions and named entities, as described next.

3.1.2 Use of BBN's IdentiFinder in Trimmer

BBN's IdentiFinder is used both for the elimination of temporal expressions and for conservative deletion of PPs containing named entities. The elimination of temporal expressions (Step 1) is a two-step process: (a) use IdentiFinder to mark temporal expressions; and (b) remove [PP ... [NP [X] ...] ...] and [NP [X]] where X is tagged as part of a temporal expression. The following examples illustrate the application of temporal expression removal rule:

(14) (i) Input: The State Department on Friday lifted the ban it had imposed on foreign fliers.

(ii) Parse: **[S [NP** *[Det The]* **State Department** *[PP [IN on] [NP [NNP Friday]]]* **[VP lifted** *[Det the]* **ban it had imposed on foreign fliers.]]**

(iii) Output: State Department lifted ban it had imposed on foreign fliers.

(15) (i) Input: An international relief agency announced Wednesday that it is withdrawing from North Korea.

(ii) Parse: **[S [NP** *[Det An]* **international relief agency][VP announced** *[NP [NNP Wednesday]]* **that it is withdrawing from North Korea.]]**

(iii) Output: International relief agency announced that it is withdrawing from North Korea.

IdentiFinder is also used to ensure that prepositional phrases containing named entities are not removed during the first round of PP removal (Step 9). However, prepositional phrases containing named entities that are descendants of SBARs are removed before the parent SBAR is removed, since we should remove a smaller constituent before removing a larger constituent that subsumes it. Sentence (16) shows an example of a SBAR subsuming two PPs, one of which contains a named entity.

(16) The commercial fishing restrictions in Washington will not be lifted [SBAR unless the salmon population increases [PP to a sustainable number] [PP in the Columbia River]].

If the PP rule were not sensitive to named entities, the PP *in the Columbia River* would be the first prepositional phrase to be removed, because it is the lowest rightmost PP in the parse. However, this PP provides an important piece of information: the location of the salmon population. The rule in Step 9 will skip the last prepositional phrase and remove the penultimate PP *to a sustainable number*.

This concludes an overview of the Trimmer rules and our syntactic sentence compression algorithm. Given a length limit, the system will produce a single compressed version of the target sentence. Multiple compressions can be generated by setting the length limit to be very small and storing the state of the sentence after each rule application as a compressed variant. In section 4, we will describe how multiple compressed candidates generated by Trimmer are used as a component of a multi-document summarization system.

[1] The reason for breaking PP removal into two stages is discussed in Section 3.1.2.

3.2 Topiary

We have used the Trimmer approach to compression in another variant of single-sentence summarization called Topiary. This system combines Trimmer with a topic discovery approach (described next) to produce a fluent summary along with additional context.

The Trimmer algorithm is constrained to build a headline from a single sentence. However, it is often the case that no single sentence contains all the important information in a story. Relevant information can be spread over multiple sentences, linked by anaphora or ellipsis. In addition, the choice of lead sentence may not be ideal and our trimming rules are imperfect.

On the other hand, approaches that construct headlines from lists of topic terms (Lewis, 1999; Schwartz et al., 1997) also have limitations. For example, Unsupervised Topic Discovery (UTD)—described below—rarely generates any topic terms that are verbs. Thus, topic lists are good at indicating the general subject but rarely give any direct indication of what events took place. Intuitively, we need both fluent text to tell what happened and topic terms to provide context.

3.2.1 Topic Term Generation: UTD and OnTopic

OnTopic (Schwartz et al., 1997) uses an HMM to assign topics to a document; topic models are derived from an annotated corpus. However, it is often difficult to acquire such data, especially for a new genre or language. UTD (Sista et al., 2002) was developed to overcome this limitation: it takes as input a large unannotated corpus and automatically creates a set of topic models with meaningful names.

The UTD algorithm has several stages. First, it analyzes the corpus to find multi-word sequences that can be treated as single tokens. It does this using two methods. One method is a minimum description length criterion that detects phrases that occur frequently relative to the individual words. The second method uses BBN's IdentiFinder to detect multi-word names. These names are added to the text as additional tokens. They are also likely to be chosen as potential topic names. In the second stage of UTD, we find those terms (both single-word and multi-word) with high *tf.idf*. Only those topic names that occur as high-content terms in at least four different documents are kept. The third stage trains topic models corresponding to these topic terms. The modified Expectation Maximization procedure of BBN's OnTopic system is used to determine which words in the documents often signify these topic names. This produces topic models. Fourth, these topic models are used to find the most likely topics for each document, which is equivalent to assigning the name of the topic model to the document as a topic term. This often assigns topics to documents where the topic name does not occur in the document text.

We found, in various experiments (Sista et al., 2002), that the topic names derived by this procedure were usually meaningful and that the topic assignment was about as good as when the topics were derived from a corpus that was annotated by people. We have also used this procedure on different languages and shown the same behavior. Since UTD is unsupervised, it can run equally well on a new language, as long as the documents can be divided into strings that approximate words.

The topic list in (17) was generated by UTD and OnTopic for a story about the FBI investigation of the 1998 bombing of the U.S. embassy in Nairobi.

(17) BIN LADEN, EMBASSY, BOMBING, POLICE OFFICIALS, PRISON, HOUSE, FIRE, KABILA

Topiary uses UTD to generate topic terms for the collection of documents to be summarized and uses OnTopic to assign the topic terms to the documents. The next section will describe how topic terms and sentence compressions are combined to form Topiary summaries.

3.2.2 Topiary Algorithm

As each Trimmer rule is applied to a sentence, the resulting state of the sentence is stored as a compressed variant of the source sentence. Topiary selects from the variants the longest one such that there is room to prepend the highest scoring non-redundant topic term. Suppose the highest scoring topic term is "terrorism" and the length threshold is 75 characters. To make room for the topic "terrorism", the length threshold is lowered by 10 characters: 9 characters for the topic and 1 character as a separator. Thus, Topiary chooses the longest trimmed variant under 65 characters that does not contain the word "terrorism", If there is no such candidate, i.e., all the trimmed variants contain the word terrorism, Topiary would consider the second highest scoring topic word, "bomb". Topiary would select the longest trimmed variant under 70 characters that does not contain the word "bomb". After Topiary has selected a trimmed variant and prepended a topic to it, it checks to see how much unused space remains under the threshold. Additional topic words are added between the first topic word and the compressed sentence until all space is exhausted.

This process results in a headline that contains one or more main topics about the story and a short sentence that says what happened concerning them. The combination is often more concise than a fully fluent sentence and compensates for the fact that the information content from the topic and the compressed sentence do not occur together in any single sentence from the source text.

As examples, sentences (18) and (19) are the outputs of Trimmer and Topiary, respectively, for the same story in which UTD selected the topic terms in (17).

(18) FBI agents this week began questioning relatives of the victims

(19) BIN LADEN, EMBASSY, BOMBING: FBI agents this week began questioning relatives

By combining topics and parse-and-trim compression, Topiary achieved the highest score on the single-document summarization task (i.e., headline generation task) in DUC-2004 (Zajic et al., 2004).

3.3 HMM Hedge

Our second approach to sentence compression, implemented in HMM Hedge, treats the observed data (the story) as the result of unobserved data (headlines) that have been distorted by transmission through a noisy channel. The effect of the noisy channel is to add story words between the headline words. The model is biased by parameters to make the resulting headlines more like Headlinese, the observed language of newspaper headlines created by copy editors.

Formally, we consider a story S to be a sequence of N words. We want to find a headline H, a subsequence of words from S, that maximizes the likelihood that H generated the story S, or:

$$argmax_H P(H|S)$$

It is difficult to directly estimate $P(H|S)$, but this probability can be expressed in terms of other probabilities that are easier to compute, using Bayes' rule:

$$P(H|S) = \frac{P(S|H)P(H)}{P(S)}$$

Since the goal is to maximize this expression over H, and $P(S)$ is constant with respect to H, the denominator of the above expression can be omitted. Thus we wish to find:

$$argmax_H P(S|H)P(H)$$

Let H be a headline consisting of words $h_1, h_2, ..., h_n$. Let the special symbols *start* and *end* represent the beginning and end of a headline, respectively. $P(H)$ can be estimated using a bigram model of Headlinese:

$$P(H) = P(h_1|start)P(h_2|h_1)...P(end|h_n)$$

The bigram probabilities of the words in the headline language were computed from a corpus of English headlines taken from 242,918 AP newswire stories in the TIPSTER corpus. The headlines contain 2,848,194 words from a vocabulary of 88,627 distinct words.

Given a story S and a headline H, the action of the noisy channel is to form S by adding non-headline words to H. Let G be the non-headline words added by the channel to the headline: $g_1, g_2, ..., g_m$. For the moment, we assume that the headline words are transmitted through the channel with probability 1. We estimate $P(S|H)$, the probability that the channel added non-headline words G to headline H to form story S. This is accomplished using a unigram model of newspaper stories that we will refer to as the general language, in contrast to the headline language. Let $P_{gl}(g)$ be the probability of non-headline word g in the general language and $P_{ch}(h) = 1$ be the probability that headline word h is transmitted through the channel as story word h.

$$\begin{aligned} P(S|H) &= P_{gl}(g_1)P_{gl}(g_2)...P_{gl}(g_m)P_{ch}(h_1)P_{ch}(h_2)...P_{ch}(h_n) \\ &= P_{gl}(g_1)P_{gl}(g_2)...P_{gl}(g_m) \end{aligned}$$

The unigram probabilities of the words in the general language were computed from 242,918 English AP news stories in the TIPSTER corpus. The stories contain 135,150,288 words from a vocabulary of 428,633 distinct words.

The process by which the noisy channel generates a story from a headline can be represented by a Hidden Markov Model (HMM) (Baum, 1972). An HMM is a weighted finite-state automaton in which each state probabilistically emits a string. The simplest HMM for generating headlines consists of two states: an H state that emits words that occur in the headline and a G state that emits all the other words in the story.

Since we use a bigram model of headlines, each state that emits headline words must "remember" the previously emitted headline word. If we did not constrain headline words to actually occur in the story, we would need an H state for each word in the headline vocabulary. However, because headline words are chosen from the story words, it is sufficient to have an H state for each story word. For any story, the HMM consists of a start state S, an end state E, an H state for each word in the story, a corresponding G state for each H state, and a state G_{start} that emits words that occur before the first headline word in the story. An H state can emit only the word it represents. The corresponding G state remembers which word was emitted by its H state and can emit any word in the story language. A headline corresponds to a path through the HMM from S to E that emits all the words in the story in the correct order. In practice the HMM is constructed with states for only the first N words of the story, where N is a constant (60), or N is the number of words in the first sentence.[2]

In example (1i), given earlier, the H states will emit the words in bold (pilots, not, allowed, to, have, guns, in, cockpits) and the G states will emit all the other words. The HMM will transition between the H and G states as needed to generate the words of the story. In the current example, the model will have states $Start, G_{start}, End$, and 28 H states with 28 corresponding G states.[3] The headline given in example (1ii) corresponds to the following sequence of states: $Start$, G_{start} 17 times, H_{pilots}, G_{pilots}, H_{not}, G_{not}, $H_{allowed}$, H_{to}, H_{have}, H_{guns}, H_{in}, G_{in}, $H_{cockpits}$, End. This path is not the only one that could generate the story in (1i). Other possibilities are:

(20) (i) Transportation Department decided airline pilots not to have guns.

(ii) Months of the terrorist has to have cockpits.

[2]Limiting consideration of headline words to the early part of the story is justified in Dorr et al. (2003a) where it was shown that more than half of the headline words are chosen from the first sentence of the story. Other methods for selecting the window of story words are possible and will be explored in future research.

[3]The subscript of a G state indicates the H state it is associated with, not the story word it emits. In the example, G_{pilots} emits story word *will*, G_{not} emits story word *be*, and G_{in} emits story word *the*.

Although (20i) and (20ii) are possible headlines for (1i), the conditional probability of (20ii) given (1i) will be lower than the conditional probability of (20i) given (1i).

The Viterbi algorithm (Viterbi, 1967) is used to select the most likely headline for a given story. We use length constraints to find the most likely headlines consisting of W words, where W ranges from 5 to 15. Multiple backpointers are used so that we can find the n most likely headlines at each length.

HMM Hedge is enhanced by three additional decoding parameters to help the system choose outputs that best mimic actual headlines: a position bias, a clump bias, and a gap bias. The incorporation of these biases changes the score produced by the decoder from a probability to a relative desirability score. The three parameters were motivated by analysis of system output and their values were set by trial and error. A logical extension to this work would be to learn the best setting of these biases, e.g., through Expectation Maximization.

The position bias favors headlines that include words near the front of the story. This reflects our observations of human-constructed headlines, in which headline words tend to appear near the front of the story. The initial position bias p is a positive number less than one. The story word in the nth position is assigned a position bias of $log(p^n)$. When an H state emits a story word, the position bias is added to the desirability score. Thus, words near the front of the story carry less of a position bias than words farther along. Note that this generalization often does not hold in the case of human interest and sports stories, which may start with a hook to get the reader's attention, rather than a topic sentence.

We also observed that human-constructed headlines tend to contain contiguous blocks of story words. Example (1ii), given earlier, illustrates this with the string "allowed to have guns". The string bias is used to favor "clumpiness". i.e., the tendency to generate headlines composed of clumps of contiguous story words. The log of the clump bias is added to the desirability score with each transition from an H state to its associated G state. With high clump biases, the system will favor headlines consisting of fewer but larger clumps of contiguous story words.

The gap bias is used to disfavor headline "gappiness", i.e., large gaps of non-headline words in the story between clumps of headline words. Although humans are capable of constructing fluent headlines by selecting widely spaced words, we observed that HMM Hedge was more likely to combine unrelated material by doing this. At each transition from a G state to an H state, corresponding to the end of a sequence of non-headline words in the story, a gap bias is applied that increases with the size of the gap between the current headline and the last headline word to be emitted. This can also be seen as a penalty for spending too much time in one G state. With high gap biases, the system will favor headlines with few large gaps.

One characteristic difference between newspaper headline text and newspaper story text is that headlines tend to be in present tense while story sentences tend to be in the past tense. Past tense verbs occur more rarely in the headline language than in the general language. HMM Hedge mimics this aspect of Headlinese by allowing morphological variation between headline verbs and the corresponding story verbs. Morphological variation for verbs is achieved by creating an H state for each available variant of a story verb. These H states still emit the story verb but they are labeled with the variant. HMM Hedge can generate a headline in which *proposes* is the unobserved headline word that emits the observed story word *proposed*, even though *proposes* does not occur in the story.

(21) (i) A group has proposed awarding $1 million in every general election to one randomly chosen voter.

(ii) Group proposes awarding $1 million to randomly chosen voter.

Finally, we constrain each headline to contain at least one verb. That is to say, we ignore headlines that do not contain at least one verb, no matter how desirable the decoding is.

Although we have described an application of HMM Hedge to blocks of story words without reference to sentence boundaries, it is also possible to use HMM as a single sentence compressor by limiting the block to the words in a single sentence.

Also, as we will see shortly, multiple alternative compressions of a sentence may be generated with HMM Hedge. The Viterbi algorithm is capable of discovering n-best compressions of a window of story words and can be constrained to consider only paths that include a specific number of H states, corresponding to compressions that contain a specific number of words. We use HMM Hedge to generate 55 compressions for each sentence by computing the five best headlines at each length, from 5 to 15 words. In Section 4 we will describe how HMM Hedge is used as a component of a multi-document summarization system.

4 Multi-Document Summarization

The sentence compression tools we developed for single-document summarization have been incorporated into our Multi-Candidate Reduction framework for multi-document summarization. MCR produces a textual summary from a collection of relevant documents in three steps. First, sentences are selected from the source documents for compression. The most important information occurs near the front of the stories, so we select the first five sentences of each document for compression. Second, multiple compressed versions of each sentence are produced using Trimmer or HMM Hedge to create a pool of candidates for inclusion in the summary. Finally, a sentence selector constructs the summary by iteratively choosing from the pool of candidates based on a linear combination of features until the summary reaches a desired length.

At present, weights for the features are determined by manually optimizing on a set of training data to maximize the ROUGE-2 recall score (Lin and Hovy, 2003), using ROUGE version 1.5.5. A typical summarization task might call for the system to generate a 250-word summary from a couple of dozen news stories. These summaries may be query-focused, in the sense that the summaries should be responsive to a particular information need, or they may be generic, in that a broad overview of the documents is desired.

Our sentence selector adopts certain aspects of Maximal Marginal Relevance (MMR) (Carbonell and Goldstein, 1998), an approach that attempts to balance relevance and anti-redundancy. In MCR's selection module, the highest scoring sentence from the pool of eligible candidates is chosen for inclusion in the summary. Features that contribute to a candidate's score can be divided into two types: dynamic and static. When a candidate is chosen, all other compressed variants of that sentence are eliminated. After a candidate is added to the summary, the dynamic features are re-computed, and the candidates are re-ranked. Candidates are added to the summary until the desired length is achieved. The ordering of candidates in the summary is the same as the order in which they were selected for inclusion. The final sentence of the summary is truncated if it causes the summary to exceed the length limit.

4.1 Static Features

Static features are calculated before sentence selection begins and do not change during the process of summary construction:

- Position. The zero-based position of the sentence in the document.
- Sentence Relevance. The relevance score of the sentence to the query.
- Document Relevance. The relevance score of the document to the query.
- Sentence Centrality. The centrality score of the sentence to the topic cluster.

- Document Centrality. The centrality score of the document to the topic cluster.
- Scores from the Compression Modules:
  - Trim rule application counts. For Trimmer-based MCR, the number of Trimmer rule instances applied to produce the candidate.
  - Negative Log Desirability. For HMM-based MCR, the relative desirability score of the candidate.

We use the Uniform Retrieval Architecture (URA), University of Maryland's software infrastructure for information retrieval tasks, to compute relevance and centrality scores for each compressed candidate. There are four such scores: the relevance score between a compressed sentence and the query, the relevance score between the document containing the compressed sentence and the query, the centrality score between a compressed sentence and the topic cluster, and the centrality score between the document containing the compressed sentence and the topic cluster. We define the topic cluster to be the entire collection of documents relevant to this particular summarization task. Centrality is a concept that quantifies how similar a piece of text is to all other texts that discuss the same general topic. We assume that sentences having higher term overlap with the query and sources more "central" to the topic cluster are preferred for inclusion in the final summary.

The relevance score between a compressed sentence and the query is an *idf*-weighted count of overlapping terms (number of terms shared by the two text segments). Inverse document frequency (*idf*), a commonly-used measure in the information retrieval literature, roughly captures term salience. The *idf* of a term t is defined by $log(N/c_t)$, where N is the total number of documents in a particular corpus and c_t is the number of documents containing term t; these statistics were calculated from one year's worth of LA Times articles. Weighting term overlap by inverse document frequency captures the intuition that matching certain terms is more important than matching others.

Lucene, a freely-available off-the-shelf information retrieval system, is used to compute the three other scores. The relevance score between the document containing the compressed sentence and the query is computed using Lucene's built-in similarity function. The centrality score between the compressed sentence and the topic cluster is the mean of the similarity between the sentence and each document comprising the cluster (once again, as computed by Lucene's built-in similarity function). The document-cluster centrality score is also computed in much the same way, by taking the mean of the similarity of the particular document with every other document in the cluster. In order to obtain an accurate distribution of term frequencies to facilitate the similarity calculation, we indexed all relevant documents (i.e., the topic cluster) along with a comparable corpus (one year of the LA Times)—this additional text essentially serves as a background model for non-relevant documents.

Some features are derived from the sentence compression modules used to generate candidates. For Trimmer, the rule application count feature of a candidate is the number of rules that were applied to a source sentence to produce the candidate. The rules are not presumed to be equally effective, so the rule application counts are broken down by rule type. For HMM Hedge, we use the relative desirability score calculated by the decoder, expressed as a negative log.

The features discussed in this section are assigned to the candidates before summary generation begins and remain fixed throughout the process of summary sentence selection. The next section discusses how candidate features are assigned new values as summary geneneration proceeds.

4.2 Dynamic Features

Dynamic features change during the process of sentence selection to reflect changes in the state of the summary as sentences are added.[4] The dynamic features are:

[4]At present the dynamic features are properties of the candidates, calculated with respect to the current summary state. There are no features directly relating to the amount of space left in the summary, so there is no mechanism that

- Redundancy. A measure of how similar the sentence is to the current summary.
- Sentence-from-doc. The number of sentences already selected from the sentence's document.

The intuition behind our redundancy measure is that candidates containing words that occur much more frequently in the current state of the summary than they do in general English are redundant to the summary. We imagine that sentences in the summary are generated by the underlying word distribution of the summary rather than the distribution of words in the general language. If a sentence appears to have been generated by the summary rather than by the general language, we take it to be redundant to the summary. Suppose we have a summary about earthquakes. The presence in a candidate of words like *earthquake*, *seismic*, and *Richter Scale*, which have a high likelihood in the summary, will make us think that the candidate is redundant to the summary.

To estimate the extent to which a candidate is more likely to have been generated by a summary than by the general language, we consider the probabilities of the words in the candidate. We estimate that the probability that a word w occurs in a candidate generated by the summary is

$$P(w) = \lambda P(w|D) + (1 - \lambda)P(w|C)$$

where D is the summary, C is the general language corpus[5], λ is a parameter estimating the probability that the word was generated by the summary and $(1-\lambda)$ is the probability that the word was generated by the general language. We have set $\lambda = 0.3$, as a general estimate of the portion of words in a text that are specific to the text's topic. We estimate the probabilities by counting the words[6] in the current summary and the general language corpus:

$$P(w|D) = \frac{count\ of\ w\ in\ D}{size\ of\ D}$$

$$P(w|C) = \frac{count\ of\ w\ in\ C}{size\ of\ C}$$

We take the probability of a sentence to be the product of the probabilities of its words, so we calculate the probability that a sentence was generated by the summary, i.e. our redundancy metric, as:

$$Redundancy(S) = \prod_{s \in S} \lambda P(s|D) + (1 - \lambda)P(s|C)$$

For ease of computation, we actually use log probabilities:

$$\sum_{s \in S} \log(\lambda P(s|D) + (1 - \lambda)P(s|C))$$

Redundancy is a dynamic feature because the word distribution of the current summary changes with every iteration of the sentence selector.

4.3 Examples of System Output

We applied our MCR framework to test data from the DUC-2006 evaluation (Dang and Harman, 2006). Given a topic description and a set of 25 documents related to the topic (drawn from AP newswire, the New York Times, and the Xinhua News Agency English Service), the system's task was to create

would affect the distribution of compressed candidates over the iterations of the sentence selector. This issue will be addressed as future work in Section 7.

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

[6]Actually, preprocessing for redundancy includes stopword removal and applying the Porter Stemmer (Porter, 1980).

Title: Native American Reservation System—pros and cons
Narrative Description: Discuss conditions on American Indian reservations or among Native American communities. Include the benefits and drawbacks of the reservation system. Include legal privileges and problems.

Figure 2: Topic D0601A from the DUC-2006 multi-document summarization task.

a 250-word summary that addressed the information need expressed in the topic. One of the topic descriptions is shown in Figure 2. The 25 documents in the document set have an average size of 1170 words, so a 250-word summary represents a compression ratio of 0.86%.

Figures 3, 4 and 5 show examples of MCR output using Trimmer compression, HMM Hedge compression, or no compression. For readability, we use ∘ as a sentence delimiter; this is not part of the actual system output. The sentences compressed by Trimmer mimic Headlinese by omitting determiners and auxiliary verbs. For example, the first sentence in Figure 3 is a compression of the following source sentence:

> Seeking to get a more accurate count of the country's American Indian population, the Census Bureau is turning to tribal leaders and residents on reservations to help overcome long-standing feelings of wariness or anger toward the federal government.

Three determiners and a form of *be* have been removed from the source sentence in the compression that appears in the summary. The removal of this material makes the sentence appear more like a headline.

In comparison with Trimmer compressions, HMM compressions are generally less readable and more likely to be misleading. Consider the final sentence in Figure 4.

(22) main purpose of reservation to pay American Indians by poverty proposals

This is a compression of the following source sentence:

(23) But the main purpose of the visit—the first to a reservation by a president since Franklin Roosevelt—was simply to pay attention to American Indians, who are so raked by grinding poverty that Clinton's own advisers suggested he come up with special proposals geared specifically to the Indians' plight.

Because HMM Hedge uses a bigram model of Headlines, it is unable to capture sentence-level grammaticality. The same limitation makes it difficult to prevent misleading or incorrect compressions.

For example, the third sentence from the end of Figure 4 seems to say that a court legalized gambling on Indian reservations:

(24) Supreme Court allows legalized gambling Indian reservations

However, it is a compression of the following source sentence:

(25) Only Monday, the California Supreme Court overturned a ballot measure that would have allowed expansion of legalized gambling on Indian reservations.

Nevertheless, we can see from the examples that sentence compression allows a summary to include more material from other sources. This increases the topic coverage of system output.

Seeking to get more accurate count of country's American Indian population, Census Bureau turning to tribal leaders and residents on reservations to help overcome long-standing feelings. ∘ American Indian reservations would get infusion. ∘ Smith and thousands seeking help for substance abuse at American Indian Community House, largest of handful of Native American cultural institutions. ∘ Clinton going to Pine Ridge Reservation for visit with Oglala Sioux nation and to participate in conference on Native American homeownership and economic development. ∘ Said Glen Revere, nutritionist with Indian Health Services on 2.8 million-acre Wind River Reservation, about 100 miles east of Jackson, Wyo. "Then we came up with idea for this community garden, and it been bigger than we ever expected." ∘ Road leading into Shinnecock Indian reservation is not welcoming one But main purpose of visit – first to reservation by president since Franklin Roosevelt – was simply to pay attention to American Indians, who raked by grinding poverty Clinton's own advisers suggested he come up with special proposals geared specifically to Indians' plight. ∘ "This highlights what going on out there, since beginning of reservation system," said Sidney Harring, professor at City University of New York School of Law and expert on Indian crime and criminal law. ∘ American Indians are victims. ∘ President Clinton turned attention to arguably poorest, most forgotten ∘ U.S. citizens: American Indians. ∘ When American Indians began embracing gambling, Hualapai tribe moved quickly to open casino. ∘ members of Northern Arapaho Tribe on Wind River Reservation started seven-acre community garden with donated land, seeds and

Figure 3: MCR Summary for DUC-2006 Topic D0601A, using Trimmer for sentence compression.

David Rocchio deputy legal counsel to Vermont Gov. Howard Dean who has been involved in discussions on Indian gambling through the National Governors Association said that the concern that governors have is not with the benefit casinos bring to tribes ∘ Native Americans living on reservations that maintain 50 percent or more unemployment are exempt from the national five year family limit on welfare benefits ∘ Smith and thousands like her are seeking help for their substance abuse at the American Indian Community House the largest of a handful of Native American cultural institutions in the New York area ∘ Juvenile crime is one strand in the web of social problems facing urban and reservation Indian communities the report said ∘ Soldierwolf's family represents the problems that plague many of the 1.3 million American Indians who live on reservations of whom 49 percent are unemployed ∘ Powless said the Onondaga people want to work with the community outside the reservation to improve the economy of the region perhaps creating tourism destinations that might include Indian culture or setting up a free trade zone at unused manufacturing sites ∘ As Indian communities across the nation struggle with short funds and a long list of problems they are watching the Navajo Nation's legal battle with the federal government ∘ recognize Indians not only Native Americans as Americans ∘ go on reservation system Harring Indian ∘ Supreme Court allows legalized gambling Indian reservations ∘ American Indian reservations tribal colleges rise faster than ∘ main purpose of reservation to pay American Indians by poverty proposals

Figure 4: MCR Summary for DUC-2006 Topic D0601A, using HMM Hedge for sentence compression

Seeking to get a more accurate count of the country's American Indian population, the Census Bureau is turning to tribal leaders and residents on reservations to help overcome long-standing feelings of wariness or anger toward the federal government. ∘ American Indian reservations would get an infusion of $1.2 billion in federal money for education, health care and law enforcement under President Clinton's proposed 2001 budget ∘ Smith and thousands like her are seeking help for their substance abuse at the American Indian Community House, the largest of a handful of Native American cultural institutions in the New York area. ∘ Clinton was going to the Pine Ridge Reservation for a visit with the Oglala Sioux nation and to participate in a conference on Native American homeownership and economic development. ∘ said Glen Revere, a nutritionist with the Indian Health Services on the 2.8 million-acre Wind River Reservation, about 100 miles east of Jackson, Wyo. "Then we came up with the idea for this community garden, and it's been bigger than we ever expected in so many ways." ∘ The road leading into the Shinnecock Indian reservation is not a welcoming one ∘ But the main purpose of the visit – the first to a reservation by a president since Franklin Roosevelt – was simply to pay attention to American Indians, who are so raked by grinding poverty that Clinton's own advisers suggested he come up with special proposals geared specifically to the Indians' plight. ∘ "This highlights what has been going on out there for 130 years,

Figure 5: MCR Summary for DUC-2006 Topic D0601A, with no sentence compression

	HMM Sentence	HMM 60 Block	Trimmer	Topiary
R1 Recall	**0.23552** (0.23014-0.24082)	**0.21381** (0.20912-0.21827)	**0.21014** (0.20436-0.21594)	**0.25143** (0.24632-0.25663)
R1 Precision	**0.21896** (0.21386-0.22384)	**0.18882** (0.18444-0.19301)	**0.20183** (0.19627-0.20722)	**0.23038** (0.22567-0.23522)
R1 F	**0.22496** (0.21983-0.22978)	**0.19966** (0.19505-0.20391)	**0.20179** (0.19612-0.20718)	**0.23848** (0.23373-0.24328)
R2 Recall	**0.06838** (0.06546-0.07155)	**0.06133** (0.05848-0.06414)	**0.06337** (0.06030-0.06677)	**0.06637** (0.06345-0.06958)
R2 Precision	**0.06287** (0.06017-0.06576)	**0.05351** (0.05097-0.05588)	**0.06230** (0.05887-0.06617)	**0.06024** (0.05747-0.06326)
R2 F	**0.06488** (0.06209-0.06785)	**0.05686** (0.05420-0.05942)	**0.06079** (0.05788-0.06401)	**0.06252** (0.05976-0.06561)

Table 2: ROUGE scores and 95% confidence intervals for 624 documents from DUC-2003 test set.

5 System Evaluations

We tested four single-document summarization systems on the DUC-2003 Task 1 test set:

- HMM Hedge using the first sentence of each document (HMM Sentence)
- HMM Hedge using the first 60 words of each document (HMM 60 block)
- Trimmer
- Topiary

Task 1 from DUC-2003 was to construct generic 75-byte summaries for 624 documents drawn from AP Newswire and the New York Times. The average size of the documents was 3,997 bytes, so a 75-byte summary represents a compression ratio of 1.9%.

An automatic summarization evaluation tool, ROUGE (Lin and Hovy, 2003), was used to evaluate the results. The system parameters were optimized by hand to maximize the ROUGE-1 recall on a comparable training corpus, 500 AP Newswire and New York Times articles from the DUC-2004 single-document short summary test data.

The ROUGE results are shown in Table 2. Results show that HMM Hedge 60 scored significantly lower than most other systems and that Topiary scored higher than all other systems for all R1 measures. In addition, HMM Hedge Sentence scored significantly higher than Trimmer for the R1 measures.

We also evaluated Trimmer and HMM Hedge as components in our Multi-Candidate Reduction framework, along with a baseline that uses the same sentence selector but does not use sentence compression. All three systems considered the first five sentences of each document and used the sentence selection algorithm presented in Section 4. The feature weights were manually optimized to maximize ROUGE-2 recall on a comparable training corpus, 1,593 Financial Times and Los Angeles Times articles grouped into 50 topics from the DUC-2005 query-focused multi-document summarization

	Trimmer	HMM Hedge	No Compression
R1 Recall	**0.29391** (0.28560-0.30247)	**0.27311** (0.26554-0.28008)	**0.27576** (0.26772-0.28430)
R2 Recall	**0.06718** (0.06332-0.07111)	**0.06251** (0.05873-0.06620)	**0.06126** (0.05767-0.06519)

Table 3: ROUGE scores and 95% confidence intervals for 50 DUC-2006 test topics, comparing three MCR variants.

	ROUGE-2	ROUGE-SU4	BE-HM
MCR Score	0.0805	0.1360	0.0413
Higher	1	1	0
Not Different	23	24	27
Range	0.0678-0.0899	0.1238-0.1475	0.0318-0.0508
Lower	11	10	8

Table 4: Official DUC-2006 Automatic Metrics for our MCR submission (System 32).

test data. The systems were used to generate query-focused, 250-word summaries using the DUC-2006 test data, described in Section 4.3.

The systems were evaluated using ROUGE, configured to omit stopwords from the calculation.[7] Results are shown in Table 3. MCR using Trimmer compressions scored significantly higher than MCR using HMM Hedge compressions and the baseline for ROUGE-1, but there was not a significant difference among the three systems for ROUGE-2.

Finally, the University of Maryland and BBN submitted a version of MCR to the official DUC-2006 evaluation. This version used Trimmer as the source of sentence compressions. Results show that use of sentence compression hurt the system on human evaluation of grammaticality. This is not surprising, since Trimmer aims to produce compressions that are grammatical in Headlinese, rather than standard English. Our MCR run scored significantly lower than 23 systems on NIST's human evaluation of grammaticality. However, the system did not score significantly lower than any other system on NIST's human evaluation of content responsiveness. A second NIST evaluation of content responsiveness asked evaluators to take readability into consideration. In this evaluation, MCR scored significantly lower than only two systems. The evaluators recognized that Trimmer compressions are not grammatical in standard English; yet, the content coverage was not significantly different from the best automatic systems and only two systems were found to be significantly more readable.

NIST computed three "official" automatic evaluation metrics for DUC-2006: ROUGE-2, ROUGE-SU4 and BE-HM. Table 4 shows the official scores of the submitted MCR system for these three metrics, along with numbers of systems that scored significantly higher, significantly lower, or were not significantly different from our MCR run. Also shown is the range of scores for the systems that were not significantly different from MCR. These results show that the performance of our MCR run was comparable to most other systems submitted to DUC-2006.

[7]This is a change in the ROUGE configuration from the official DUC-2006 evaluation. We note that the removal of non-essential stopwords (typical of Headlinese) is an important component of Trimmer-based sentence compression. For internal system comparisons, we configure ROUGE in a way that will allow us to detect system differences relevant to our research focus. For reporting of official ROUGE results on submitted systems we use the community's accepted ROUGE configurations.

The evaluation in Table 3 suggests that Trimmer sentence compression is preferable to HMM Hedge sentence compression for generation of English summaries of collections of document in English. However, HMM Hedge may prove to have value with noisier data, as we discuss in the next section. Nevertheless, sentence compression appears to be a valuable component of our framework for multi-document summarization, thus validating the ideas behind Multi-Candidate Reduction.

6 Applications to Different Types of Texts

We have applied the MCR framework to summarizing different types of texts. In this section we briefly touch on genre-specific issues that are the subject of ongoing work. Trimmer, Topiary, and HMM Hedge were designed for summarization of written news. In this genre, the lead sentence is almost always the first non-trivial sentence of the document. More sophisticated methods for finding lead sentences did not outperform the baseline of simply selecting the first sentence for AP wire "hard" news stories. However, some types of articles, such as sports stories, opinion pieces, and movie reviews often do not have informative lead sentences and will require additional work in finding the best sentence for compression.

MCR has also been applied to summarizing transcripts of broadcast news—another input form where lead sentences are often not informative. The conventions of broadcast news introduce categories of story-initial light content sentences, such as "I'm Dan Rather" or "We have an update on the story we've been following". These present challenges for the filtering stage of our MCR framework.

Such texts are additionally complicated by a range of problems not encountered in written news: noise introduced by automatic speech recognizers or other faulty transcription, issues associated with sentence boundary detection and story boundary detection. If word error rate is high, parser failures can prevent Trimmer from producing useful output. In this context, HMM Hedge becomes more attractive, since our language models are more resilient to noisy input.

We have performed an initial evaluation of Trimmer, Topiary, and a baseline consisting of the first 75 characters of a document, on the task of creating 75-character headlines for broadcast news transcriptions (Zajic, 2007). The corpus for this task consisted of 560 broadcast news stories from ABC, CNN, NBC, Public Radio International, and Voice of America. We used ROUGE-1 recall to evaluate the summaries and found that both systems scored higher than the baseline and that Topiary scored higher than Trimmer. However there were no significant differences among the systems.

Another application of our framework is the summarization of email threads—collections of emails that share a common topic or were written as responses to each other. This task can essentially be treated as a multi-document summarization problem, albeit email thread structure introduces some constraints with respect to the ordering of summary sentences. Noisy data is inherent in this problem and pre-processing to remove quoted text, attachments, and headers is crucial. We have found that metadata, such as the name of the sender of each included extract help make email summaries easier to read.

We performed an initial evaluation of HMM Hedge and Trimmer as the source of sentence compressions for an email thread summarization system based on the MCR framework (Zajic, 2007). The corpus for this task consisted of 10 manually constructed email threads from the Enron Corpus (Klimt and Yang, 2004). We used ROUGE-1 and ROUGE-2 recall with jackknifing to compare the automatic systems and the human summarizers. We did not observe a significant difference between the two systems, but we found that the task of summarizing email threads was extremely difficult for the humans (one summarizer scored significantly worse than the automatic systems). This application of MCR to email thread summarization is an initial effort. The difficulty of the task for the humans suggests that the community needs to develop a clearer understanding of what makes a good email thread summary and to explore practical uses for them.

Finally, Trimmer and HMM Hedge have been applied to Hindi-English cross-language summarization. In this case, Trimmer was applied to the output of machine translation. We adapted HMM Hedge to cross-lingual summarization by using the mechanism developed for morphological variation to represent translation probabilities from Hindi story words to English headline words. For more details, see Dorr et al. (2003a).

7 Future Work

Future work on text summarization under the Multi-Candidate Reduction framework will focus on the three main components of the architecture: sentence filtering, sentence compression, and candidate selection.

For single document summarization, the simple technique of selecting the first non-trivial sentence of a document for compression remains the best approach. However, for human interest stories or sports articles, this approach is less effective. In broadcast news transcripts, the first sentence often does not contain important information. Currently, filtering for multi-document summarization also relies on the assumption that important information tends to appear near the front of documents—the first five sentences of each document are retained to generate compressed candidates. An interesting area of future work is to explore other approaches to filtering, such as using query relevance and document centrality, to move beyond the baseline of selecting the first n sentences. For HMM Hedge, these methods can be used to determine the optimal blocks of text on which to apply the decoder.

Currently, Trimmer produces multiple compressions by applying rules in a fixed order; the state of the compressed sentence after each rule application becomes a candidate. A richer pool of candidates can be produced by modifying Trimmer rules to operate in order-independent combinations, rather than a fixed sequence. We believe that the sentence selector can produce better summaries if it has larger pools of candidates to choose from. Naturally, different sentence compressions are not the only techniques for enriching the candidate pool—other possibilities include merging sentences and resolving anaphora. Topiary will also be enhanced by using multiple combinations of compressions and topic terms in the context of headline generation.

We also plan to enrich the candidate selector by taking into account more features of the current summary state. Possibilities include sentence selector iteration count and remaining summary space, as well as feature weights that change during the progress of summary generation. These extensions will allow us to study the distribution of compressed and uncompressed sentences across sentence selector iterations. System output can potentially be improved by finer-grained control of this distribution. These features might also help avoid the current problem in which the final sentence is truncated due to length restrictions (e.g., by selecting a final sentence of more appropriate length).

Proper setting of parameters is another important area for future work. Systematic optimization of parameter values in HMM Hedge and the sentence selector could lead to significant improvements in output quality. A logical extension to this work would be to learn the best parameter settings, e.g., through Expectation Maximization.

At present, MCR focuses exclusively on summary content selection and does not take sentence ordering into consideration when constructing the summary. Naturally, high-quality summaries should read fluently in addition to having relevant content. Recent work in this area that can be applied to MCR includes includes Conroy et al. (2006), Barzilay et al. (2002), Okazaki et al. (2004), Lapata (2003), and Dorr and Gaasterland (this special issue 2007). Within the MCR architecture, fluency considerations can be balanced with other important factors such as relevance and anti-redundancy through appropriate feature weighting.

8 Conclusion

This work presents Multi-Candidate Reduction, a general architecture for multi-document summarization. The framework integrates successful single-document compression techniques that we have previously developed. MCR is motivated by the insight that multiple candidate compressions of source sentences should be made available to subsequent processing modules, which may have access to more information for summary construction. This is implemented in a dynamic feature-based sentence selector that iteratively builds a summary from compressed variants. Evaluations show that sentence compression plays an important role in multi-document summarization and that our MCR framework is both flexible and extensible.

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. The first author would like to thank Naomi for proofreading, support, and encouragement. The second author would like to thank Steve, Carissa, and Ryan for their energy enablement. The third author would like to thank Esther and Kiri for their kind support.

References

L. Bahl, F. Jelinek, and R. Mercer. 1983. A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(2):179–190.

M. Banko, V. Mittal, and M. Witbrock. 2000. Headline generation based on statistical translation. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL 2000)*, pages 318–325, Hong Kong.

R. Barzilay, N. Elhadad, and K. McKeown. 2002. Inferring strategies for sentence ordering in multidocument news summarization. *Journal of Artificial Intelligence Research*, 17:35–55.

L. Baum. 1972. An inequality and associated maximization technique in statistical estimation of probabilistic functions of a Markov process. *Inequalities*, 3:1–8.

S. Bergler, R. Witte, M. Khalife, Z. Li, and F. Rudzicz. 2003. Using knowledge-poor coreference resolution for text summarization. In *Proceedings of the HLT-NAACL 2003 Text Summarization Workshop and Document Understanding Conference (DUC 2003)*, pages 85–92, Edmonton, Alberta.

D. Bikel, R. Schwartz, and R. Weischedel. 1999. An algorithm that learns what's in a name. *Machine Learning*, 34(1/3):211–231.

S. Blair-Goldensohn, D. Evans, V. Hatzivassiloglou, K. McKeown, A. Nenkova, R. Passonneau, B. Schiffman, A. Schlaikjer, A. Siddharthan, and S. Siegelman. 2004. Columbia University at DUC 2004. In *Proceedings of the 2004 Document Understanding Conference (DUC 2004) at HLT/NAACL 2004*, pages 23–30, Boston, Massachusetts.

P. Brown, J. Cocke, S. Pietra, V. Pietra, F. Jelinek, J. Lafferty, R. Mercer, and P. Roossin. 1990. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85.

J. Carbonell and J. Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1998)*, pages 335–336, Melbourne, Australia.

Eugene Charniak. 2000. A Maximum-Entropy-Inspired Parser. In *Proceedings of the First Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL 2000)*, pages 132–139, Seattle, Washington.

J. Clarke and M. Lapata. 2006. Models for sentence compression: A comparison across domains, training requirements and evaluation measures. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING/ACL 2006)*, pages 377–384, Sydney, Australia.

J. Conroy, J. Schlesinger, and J. Goldstein. 2005. CLASSY query-based multi-document summarization. In *Proceedings of the 2005 Document Understanding Conference (DUC-2005) at NLT/EMNLP 2005*, Vancouver, Canada.

J. Conroy, J. Schlesinger, D. O'Leary, and J. Goldstein. 2006. Back to basics: CLASSY 2006. In *Proceedings of the 2006 Document Understanding Conference (DUC 2006) at HLT/NAACL 2006*, New York, New York.

D. Cutting, J. Pedersen, and P. Sibun. 1992. A practical part-of-speech tagger. In *Proceedings of the Third Conference on Applied Natural Language Processing*, Trento, Italy.

Hoa Dang and Donna Harman. 2006. *Proceedings of the 2006 Document Understanding Conference (DUC 2006) at HLT/NAACL 2006*.

B. Dorr and T. Gaasterland. this special issue, 2007. Exploiting aspectual features and connecting words for summarization-inspired temporal-relation extraction. *Information Processing and Management*.

B. Dorr, D. Zajic, and R. Schwartz. 2003a. Cross-language headline generation for Hindi. *ACM Transactions on Asian Language Information Processing (TALIP)*, 2(3):270–289.

B. Dorr, D. Zajic, and R. Schwartz. 2003b. 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.

T. Dunning. 1994. Statistical identification of language. Technical Report MCCS 94-273, New Mexico State University.

T. Euler. 2002. Tailoring text using topic words: Selection and compression. In *Proceedings of 13th International Workshop on Database and Expert Systems Applications (DEXA 2002)*, pages 215–222, Aix-en-Provence, France.

J. Goldstein, V. Mittal, J. Carbonell, and M. Kantrowitz. 2000. Multi-document summarization by sentence extraction. In *Proceedings of ANLP/NAACL 2000 Workshop on Automatic Summarization*, pages 40–48.

D. Harman and M. Liberman. 1993. TIPSTER Complete. Linguistic Data Consortium (LDC), Philadelphia.

H. Jing and K. McKeown. 2000. Cut and paste based text summarization. In *Proceedings of the First Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL 2000)*, pages 178–185, Seattle, Washington.

B. Klimt and Y. Yang. 2004. Introducing the Enron Corpus. In *Proceedings of the First Conference on Email and Anti-Spam (CEAS)*, Mountain View, California.

K. Knight and D. Marcu. 2000. Statistics-based summarization—step one: Sentence compression. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-2000)*, Austin, Texas.

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

M. Lapata. 2003. Probabilistic text structuring: Experiments with sentence ordering. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL 2004)*, pages 545–552, Barcelona, Spain.

David Dolan Lewis. 1999. An evaluation of phrasal and clustered representations on a text categorization task. In *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1992)*, pages 37–50, Copenhagen, Denmark.

C.-Y. Lin and E. Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Human Language Technology Conference and the North American Chapter of the Association for Computational Linguistics Annual Meeting (HLT/NAACL 2003)*, pages 71–78, Edmonton, Alberta.

I. Mårdh. 1980. *Headlinese: On the Grammar of English Front Page Headlines*. Malmo.

E. Mays, F. Damerau, and R. Mercer. 1990. Context-based spelling correction. In *Proceedings of IBM Natural Language ITL*, pages 517–522, Paris, France.

S. Miller, L. Ramshaw, H. Fox, and R. Weischedel. 2000. A novel use of statistical parsing to extract information from text. In *Proceedings of the First Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL 2000)*, pages 226–233, Seattle, Washington.

S. Muresan, E. Tzoukermann, and J. Klavans. 2001. Combining linguistic and machine learning techniques for email. In *Proceedings of the ACL/EACL 2001 Workshop on Computational Natural Language Learning (ConLL)*, pages 290–297, Toulouse, France.

N. Okazaki, Y. Matsuo, and M. Ishizuka. 2004. Improving chronological sentence ordering by precedence relation. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004)*, pages 750–756, Geneva, Switzerland.

M. Porter. 1980. An algorithm for suffix stripping. *Program*, 14(3):130–137.

D. Radev, T. Allison, S. Blair-Goldensohn, J. Blitzer, A. Çelebi, S. Dimitrov, E. Drabek, A. Hakim, W. Lam, D. Liu, J. Otterbacher, H. Qi, H. Saggion, S. Teufel, M. Topper, A. Winkel, and Z. Zhang. 2004. MEAD—a platform for multidocument multilingual text summarization. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC 2004)*, Lisbon, Portugal.

R. Schwartz, T. Imai, F. Jubala, L. Nguyen, and J. Makhoul. 1997. A maximum likelihood model for topic classification of broadcast news. In *Proceedings of the Fifth European Speech Communication Association Conference on Speech Communication and Technology (Eurospeech-97)*, Rhodes, Greece.

S. Sista, R. Schwartz, T. Leek, and J. Makhoul. 2002. An algorithm for unsupervised topic discovery from broadcast news stories. In *Proceedings of the 2002 Human Language Technology Conference (HLT)*, pages 99–103, San Diego, California.

J. Turner and E. Charniak. 2005. Supervised and unsupervised learning for sentence compression. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pages 290–297, Ann Arbor, Michigan.

L. Vanderwende, H. Suzuki, and C. Brockett. 2006. Microsoft Research at DUC2006: Task-focused summarization with sentence simplification and lexical expansion. In *Proceedings of the 2006 Document Understanding Conference (DUC 2006) at HLT/NAACL 2006*, New York, New York.

A. Viterbi. 1967. Error bounds for convolution codes and an asymptotically optimal decoding algorithm. *IEEE Transactions on Information Theory*, 13:260–269.

R. Wang, N. Stokes, W. Doran, E. Newman, J. Carthy, and J. Dunnion. 2005. Comparing Topiary-style approaches to headline generation. In *Lecture Notes in Computer Science: Advances in Information Retrieval: 27th European Conference on IR Research (ECIR 2005)*, volume 3408, Santiago de Compostela, Spain. Springer Berlin / Heidelberg.

D. Zajic, B. Dorr, and R. Schwartz. 2004. BBN/UMD at DUC-2004: Topiary. In *Proceedings of the 2004 Document Understanding Conference (DUC 2004) at NLT/NAACL 2004*, pages 112–119, Boston, Massachusetts.

D. Zajic, B. Dorr, J. Lin, and R. Schwartz. 2005a. UMD/BBN at MSE2005. In *Proceedings of the MSE2005 Track of the ACL 2005 Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization*, Ann Arbor, Michigan.

D. Zajic, B. Dorr, R. Schwartz, C. Monz, and J. Lin. 2005b. A sentence-trimming approach to multi-document summarization. In *Proceedings of the 2005 Document Understanding Conference (DUC-2005) at NLT/EMNLP 2005*, pages 151–158, Vancouver, Canada.

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

L. Zhou and E. Hovy. 2003. Headline summarization at ISI. In *Proceedings of the HLT-NAACL 2003 Text Summarization Workshop and Document Understanding Conference (DUC 2003)*, pages 174–178, Edmonton, Alberta.