Proceedings of HLT-NAACL 2003
Main Papers , pp. 205-211
Edmonton, May-June 2003

A Web-Trained Extraction Summarization System

Liang Zhou and Eduard Hovy
USC Information Sciences Institute
4676 Admiralty Way
Marina del Rey, CA 90292-6695
{liangz, hovy}@isi.edu

Abstract

A serious bottleneck in the development of trainable text summarization systems is the shortage of training data. Constructing such data is a very tedious task, especially because there are in general many different correct ways to summarize a text. Fortunately we can utilize the Internet as a source of suitable training data. In this paper, we present a summarization system that uses the web as the source of training data. The procedure involves structuring the articles downloaded from various websites, building adequate corpora of (summary, text) and (extract, text) pairs, training on positive and negative data, and automatically learning to perform the task of extraction-based summarization at a level comparable to the best DUC systems.

1 Introduction

The task of an extraction-based text summarizer is to select from a text the most important sentences that are in size a small percentage of the original text yet still as informative as the full text (Kupiec et al., 1995). Typically, trainable summarization systems characterize each sentence according to a set of predefined features and then learn from training material which feature combinations are indicative of good extract sentences. In order to learn the characteristics of indicative summarizing sentences, a large enough collection of (summary, text) pairs must be provided to the system.

Research in automated text summarization is constantly troubled by the difficulty of finding or constructing large collections of (extract, text) pairs. Usually, (abstract, text) pairs are available and can be easily obtained (though not in sufficient quantity to support fully automated learning for large domains). But abstract sentences are not identical to summary sentences and hence make direct comparison difficult. Therefore, some algorithms have been introduced to generate (extract, text) pairs expanded from (abstract, text) inputs (Marcu, 1999).

The explosion of the World Wide Web has made accessible billions of documents and newspaper articles. If one could automatically find short forms of longer documents, one could build large training sets over time. However, one cannot today retrieve short and long texts on the same topic directly.

News published on the Internet is an exception. Although it is not ideally organized, the topic orientation and temporal nature of news makes it possible to impose an organization and thereby obtain a training corpus on the same topic. We hypothesize that weekly articles are sophisticated summaries of daily ones, and monthly articles are summaries of weekly ones, as shown in Figure 1. Under this hypothesis, how accurate an extract summarizer can one train? In this paper we first describe the corpus reorganization, then in Section 3 the training data formulation and the system, the system evaluation in Section 4, and finally future work in Section 5.

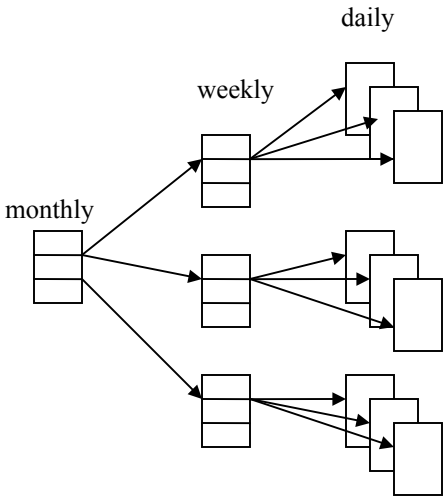

Figure 1. Corpus structure.

2 Corpus Construction

2.1 Download Initial Collection

The Yahoo Full Coverage Collection (YFCC) was downloaded from http://fullcoverage.yahoo.com during December 2001. The full coverage texts were downloaded based on a snapshot of the links contained in Yahoo Full Coverage at that time. A spider crawled the top eight categories: U.S., World, Business, Technology, Science, Health, Entertainment, and Sports. All news links in each category were saved in an index page that contained the headline and its full text URL. A page fetcher then downloaded all the pages listed in the snapshot index file.

Under the eight categories, there are 463 subcategories, 216590 news articles.

2.2 Preprocessing

All the articles in the YFCC are preprocessed as following. Each article is in the original raw html form with actual contents buried in layers of irrelevant tags and markings. Identifying the text body is a challenging process (Finn et al., 2001). The system identifies the main body of the article using a set of retrieval templates, and then further eliminates useless information embedded in the main body by considering each opening and closing tag set. For example, if the tag name indicates the contents between the opening and closing tags are images or just meta-info, the contents is discarded.

The clean texts are then processed by a sentence breaker, Lovin's stemmer, a part-of-speech tagger, and converted into standard XML form.

2.3 Chronological Reorganization

The news articles posted under Yahoo Full Coverage are from 125 different web publishers. Except for some well-known sites, the publishing frequencies for the rest of the sites are not known. But Yahoo tends to use those publishers over and over again, leaving for each publisher a trail of publishing habit. Our system records the publishing date for each article from each publisher chronologically, and then calculates the publishing frequency for each publisher. Over all the articles from a publisher, the system computes the minimum publishing gap (MPG) between two articles. If the MPG is less than 3 days or the MPG is unknown in the case of publishers seen only once in the YFCC, then this publisher is labeled as a daily publisher. If the MPG is greater than 3 days but less than 15, it is labeled as a weekly publisher. Publishers with all other MPG values are labeled as monthly publishers.

For each article in the collection, the system relabels it as a daily, weekly, or monthly publication. Each domain under each category in the collection is then restructured into a hierarchy by year, months within the year, weeks of each month, and finally days of each week. The visualization of an example of the hierarchical structure of the domain Africa under category World is shown in Figure 2.

3 System

Recognizing (*summary, text*) pairs automatically from the web repository is the key to overcoming the constant shortage of summarization training data. After taking a closer examination of the reorganized YFCC, one notices that for each day, there are a number of

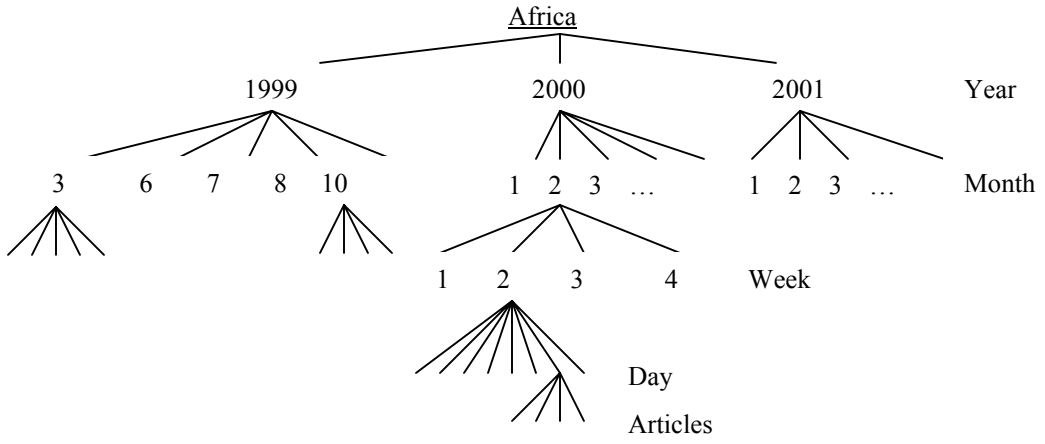

Figure 2: The hierarchical structure for domain Africa.

articles published that update the progress of a particular news topic. Daily articles are published by identified daily publishers. Then at the end of each week, there are several weekly articles published by weekly publishers on the same topic. At the end of each month, again there are articles on the same topic posted by publishers labeled as monthly publishers. There is a common thematic connection between the daily articles and the weekly articles, and between the weekly articles and the monthly articles. The daily articles on a particular event are more detailed, and are written step-by-step as it was happening. The weekly articles review the daily articles and recite important snippets from the daily news. The monthly articles are written in a more condensed fashion quoting from the weeklies.

Instead of asking human judges to identify informative sentences in documents, and since beautifully written "summaries" are already available, we need to align the sentences from the daily articles with each weekly article sentence, and align weekly article sentences with each monthly article sentence, in order to collect the (*summary, text*) pairs and eventually generate (*extract, text*) pairs. The pairs are constructed at both sentence and document levels.

3.1 Alignment

In our system, three methods for sentence-level and document-level alignment are investigated:

- **extraction-based**: Marcu (1999) introduces an algorithm that produces corresponding extracts given (*abstract, text*) tuples with maximal semantic similarity. We duplicated this algorithm but replaced inputs with (*summary, text*), parents and their respective children in the hierarchical domain tree. Thus for example, the summary is a monthly article when the text is a weekly article or a weekly article when the text is a daily one. We start with the cosine-similarity metric stated in (Marcu 1999) and keep deleting sentences that are not related to the summary document until any more deletion would result in a drop in similarity with the summary. The resulting set of sentences is the extract concerning the topic discussed in the summary. It forms the pair (*extract, text*). If there is more than one summary for a particular text (nonsummary article), the resulting extracts will vary if the summary articles are written on the same event, but are focused on different perspectives. Thus, a summary article may be aligned with several extracts and extracts generated from a single text may align with many summaries. The relationship amongst summaries, extracts, and texts forms a network topology.

  To generate sentence level alignment, we replaced the input with (*summary sentence, text*) pairs. Starting with a nonsummary text, the sentences that are irrelevant to the summary sentence are deleted repeatedly, resulting in the preservation of sentences similar in meaning to the summary sentence. For each sentence in the summary, it is aligned with a number of nonsummary sentences to form (*summary sentence, nonsummary sentences*) pairs. This alignment is done for each sentence of the summary articles. Finally for each nonsummary we group together all the aligned sentences to form the pair (*extract, text*).
- **similarity-based**: inspired by sentence alignment for multilingual parallel corpora in Machine Translation (Church, 1993; Fung and Church, 1994; Melamed, 1999), we view the alignment between sentences from summaries and sentences from nonsummaries as the alignment of monolingual parallel texts at the sentence level. In every domain of the YFCC, each article is represented as a vector in a vector space where each dimension is a distinct non-stop word appearing in this domain. Measuring the cosine-similarity between two articles, we can decide whether they are close semantically. This method has been widely used in Information Retrieval (Salton, 1975). To extend this idea, we measure the cosine-similarity between two sentences, one from a summary (weekly or monthly article) and the other one from a nonsummary (daily or weekly article). If the similarity score between the two crosses a predetermined threshold, the two sentences are aligned to form the pair (*summary sentence, text sentence*). The relationship between sentences is many-to-many. With any particular nonsummary article, sentences that are aligned with summary sentences form the extract and the pair (*extract, text*).
- **summary-based**: concerned with the noise that may accompany similarity calculations from extraction-based and similarity-based alignments, we align an entire summary article with all its nonsummary articles published in the same time period, as determined from the previously described chronological reorganization. The alignment results are pairs of the format (*summary, texts*). One summary can only be aligned with a certain group of nonsummaries. Each nonsummary can be aligned with many summaries. No sentence level alignment is done with this method.

3.2 Training Data

The main goal of a leaning-based extraction summarization system is to learn the ability to judge whether a particular sentence in a text appear in the extract or not. Therefore, two sets of training data are needed, one indicative enough for the system to select a sentence to be in the extract (labeled as positive data), the other indicative enough for the system to keep the sentence from being added to the extract (labeled as negative data). For each of the alignment methods, we produce summary training data and nonsummary training data for each domain in the YFCC.

From extraction-based and similarity-based alignment methods, for each nonsummary article, there are two sets of sentences, the set of sentences that compose the extract with the respect to some summary article or align with summary sentences, and the rest of the sentences that are not related to the summary or aligned. The two sets of sentences over all articles in the domain form the positive and negative training data sets.

Using summary-based alignment, all the summary articles are in the positive training set, and all the nonsummary material is in the negative set. Full texts are used.

3.3 Bigram Estimates Extract Desirability

We treat each domain independently. Using a bigram model, we estimate the desirability of a sentence appearing in the extract *P(S)* from the summary training data as:

$$P(S) = P(w_1 \mid start)\, P(w_2 \mid w_1) \ldots P(w_n \mid w_{n-1})$$

We estimate the desirability of a sentence not appearing in the extract *P'(S)* from the nonsummary training data as:

$$P'(S) = P'(w_1 \mid start)\, P'(w_2 \mid w_1) \ldots P'(w_n \mid w_{n-1})$$

For each domain in the YFCC, a summary bigram table and a nonsummary bigram table are created.

3.4 Extraction Process

Zajic et al. (2002) used a Hidden Markov Model as part of their headline generation system. In our system, we started with a similar idea of a lattice for summary extraction. In Figure 3, E states emit sentences that are going to be in the extract, and N states emit all other sentences. Given an input sentence, if *P(S)* is greater than *P'(S)*, it means that the sentence has a higher desirability of being an extraction sentence; otherwise, the sentence will not be included in the resulting extract.

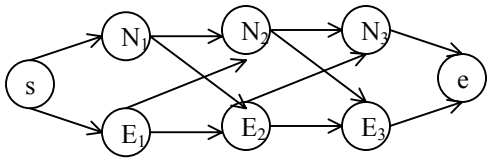

Figure 3. Lattice.

After reading in the last sentence from the input, the extract is created by traversing the path from start state to end state and only outputting the sentences emitted by the E states.

The extracts generated are in size shorter than the original texts. However, the number of sentences that E states emit cannot be predetermined. This results in unpredictable extract length. Most frequently, longer extracts are produced. The system needs more control over how long extracts will be in order for meaningful evaluation to be conducted.

To follow up on the lattice idea, we used the following scoring mechanism:

$$R = P(S) / P'(S)$$

R indicates the desirability ratio of the sentence being in the extract over it being left out. For each sentence from the input, it is assigned an *R* score. Then all the sentences with their *R* scores are sorted in descending order. With respect to the length restriction, we choose only the top *n R*-scored sentences.

3.5 Selecting the Training Domain

There are 463 domains under the 8 categories of YFCC, meaning 463 paired summary-bigram and nonsummary-bigram tables. On average for each domain, the summary-bigram table contains 20000 entries; the nonsummary-bigram table contains 173000 entries. When an unknown text or a set of unknown texts come in to be summarized, the system needs to select the most appropriate pair of bigram tables to create the extract. The most desirable domain for an unknown text or texts contains articles focusing on the same issues as the unknown ones. Two methods are used:

- **topic signature** (Lin and Hovy, 2000): a topic signature is a family of related terms {*topic, signature*}, where *topic* is the target concept and *signature* is a vector of related terms. The topic in the formula is assigned with the domain name. To construct the set of related words, we consider only nouns because we are only interested in the major issues discussed in the domain, not in how those issues evolved. Each noun in the domain

receives a *tf.idf* score. 30 top-scoring nouns are selected to be the signature representing the domain. For each test text, its signature is computed with the same *tf.idf* method against each domain. The domain that has the highest number of overlaps in signature words is selected and its bigram tables are used to construct the extract of the test text. The following table illustrates. Inputs are three sets of 10 documents each from the DUC01 training corpus concerning the topics on Africa, earthquake, and Iraq, respectively. The scores are the total overlaps between a domain and each individual test set. The Three sets are all correctly classified.

domain/input	Africa	Earthquake	Iraq
Africa	24	10	9
Earthquake	7	20	8
Iraq	48	8	97

- **hierarchical signature**: each domain is given a name when it was downloaded. The name gives a description of the domain at the highest level. Since the name is the most informative word, if we gather the words that most frequently co-occur within the sentence(s) that contain the name itself, a list of less informative but still important words can become part of the domain signature. Using this list of words, we find another list of words that most frequently co-occur with each of them individually. Therefore, a three-layer hierarchical domain signature can be created: level one, the domain name; level two, 10 words with the highest co-occurrences with the domain name; level three, 10 words that most frequently co-occur with level two signatures. Again only nouns are considered. For example, for domain on Iraq, the level one signature is "Iraq"; level two signatures are "Saddam", "sanction", "weapon", "Baghdad", and etc.; third level signatures are "Gulf", "UN", "Arab", "security", etc. The document signature for the test text is computed the same way as in the topic signature method. Overlap between the domain signature and the document signature is computed with a different scoring system, in which the weights are chosen by hand. If level one is matched, add 10 points; for each match at level two, add 2 points; for each match at level three, add 1 point. The domain that receives the highest points will be selected. A much deeper signature hierarchy can be created recursively. Through experiment, we see that a three-level signature suffices. The following table shows the effects of this method:

domain/input	Africa	Earthquake	Iraq
Africa	86	7	41
Earthquake	7	74	0
Iraq	15	26	202

Since it worked well for our test domains, we employed the topic-signature method in selecting training domains.

4 Evaluation

4.1 Alignment Choice

To determine which of the alignment methods of Section 3.1 is best, we need true summaries, not monthly or weekly articles from the web. We tested the equivalencies of the three methods on three sets of articles from the DUC01 training corpus, which includes human-generated "gold standard" summaries. They are on the topics of Africa, earthquake, and Iraq. The following table shows the results of this experiment. Each entry demonstrates the cosine similarity, using the *tf.idf* score, of the extracts generated by the system using training data created from the alignment method in the column, compare to the summaries generated by human.

	extraction	similarity	summary
Africa	0.273	0.304	0.293
Earthquake	0.318	0.332	0.342
Iraq	0.234	0.246	0.247

We see that all three methods produce roughly equal extracts, when compared with the gold standard summaries. The summary-based alignment method is the least time consuming and the most straightforward method to use in practice.

4.2 System Performance

There are 30 directories in the DUC01 testing corpus. All articles in each directory are used to make the selection of its corresponding training domain, as described in Section 3.5. Even if no domain completely covers the event, the best one is selected by the system.

To evaluate system performance on summary creation, we randomly selected one article from each directory from the DUC01 testing corpus, for each article, there are three human produced summaries. Our system summarizes each article three times with the length restriction respectively set to the lengths of the three human summaries. We also evaluated the DUC01 single-document summarization baseline system results (first 100 words from each document) to set a lower

bound. To see the upper bound, each human generated summary is judged against the other two human summaries on the same article. DUC01 top performer, system from SMU, in single-document summarization, was also evaluated. In all, 30 * 3! human summary judgments, 30 * 3 baseline summary judgments, 30 SMU system judgments, and 30 * 3 system summary judgments are made. The following table is the evaluation results using the SEE system version 1.0 (Lin 2002), with visualization in Figure 4. Summary model units are graded as full, partial, or none in completeness in coverage with the peer model units. And Figure 5 shows an example of the comparison between the human-created summary and the system-generated extract.

	SRECALL	SPRECISON	LRECALL	LPRECISION
Baseline	0.246	0.306	0.301	0.396
System	0.452	0.341	0.577	0.509
SMU	0.499	0.482	0.583	0.672
Human	0.542	0.500	0.611	0.585

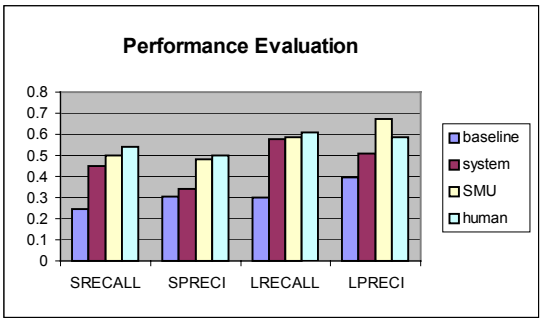

Figure 4. System performance.

The performance is reported on four metrics. Recall measures how well a summarizer retains original content. Precision measures how well a system generates summaries. SRECALL and SPRECISION are the strict recall and strict precision that take into consideration only units with full completeness in unit coverage. LRECALL and LPRECISION are the lenient recall and lenient precision that count units with partial and full completeness in unit coverage. Extract summaries that are produced by our system has comparable performance in recall with SMU, meaning that the coverage of important information is good. But our system shows weakness in precision due to the fact that each sentence in the system-generated extract is not compressed in any way. Each sentence in the extract has high coverage over the human summary. But sentences that have no value have also been included in the result. This causes long extracts on average, hence, the low average in precision measure. Since our sentence ranking mechanism is based on desirability, sentences at the end of the extract are less desirable and can be removed. This needs further investigation. Clearly there is the need to reduce the size of the generated summaries. In order to produce simple and concise extracts, sentence compression needs to be performed (Knight and Marcu, 2000).

Despite the problems, however, our system's performance places it at equal level to the top-scoring systems in DUC01. Now that the DUC02 material is also available, we will compare our results to their top-scoring system as well.

4.3 Conclusion

One important stage in developing a learning-based extraction summarization system is to find sufficient and relevant collections of *(extract, text)* pairs. This task is also the most difficult one since resources of constructing the pairs are scarce. To solve this bottleneck, one wonders whether the web can be seen as a vast repository that is waiting to be tailored in order to fulfill our quest in finding summarization training data. We have discovered a way to find short forms of longer documents and have built an extraction-based summarizer learning from reorganizing news articles from the World Wide Web and performing at a level comparable to DUC01 systems. We are excited about the power of how reorganization of the web news articles has brought us and will explore this idea in other tasks of natural language processing.

5 Future Work

Multi-document summarization naturally comes into picture for future development. Our corpus organization itself is in the form of multiple articles being summarized into one (monthly or weekly). How do we learn and use this structure to summarize a new set of articles?

Headline generation is another task that we can approach equipped with our large restructured web corpus.

We believe that the answers to these questions are embedded in the characteristics of the corpus, namely the WWW, and are eager to discover them in the near future.

Acknowledgement

We want to thank Dr. Chin-Yew Lin for making the Yahoo Full Coverage Collection download.

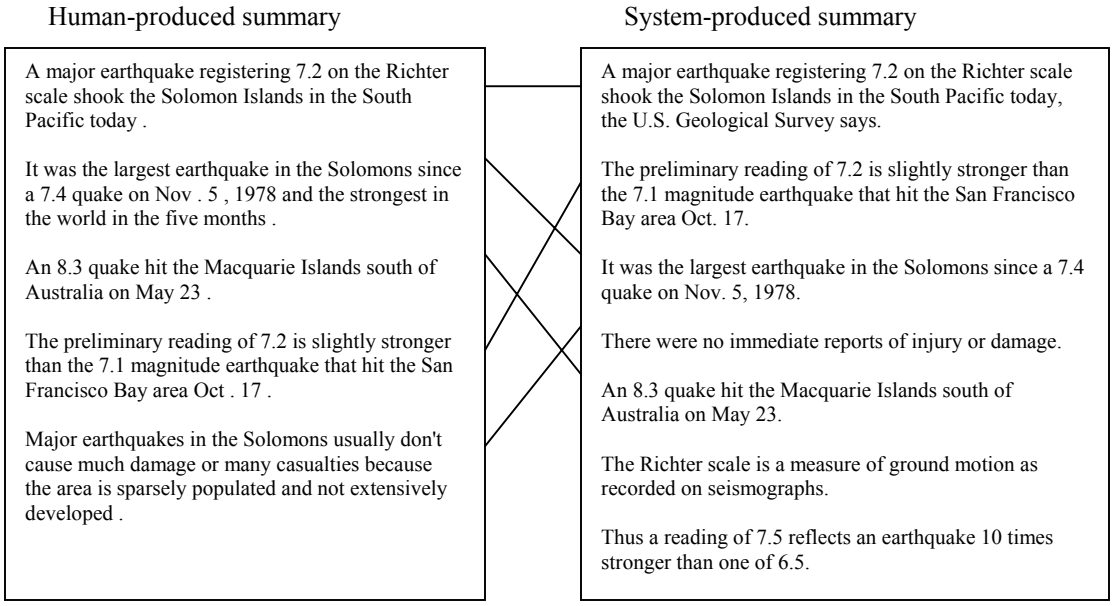

Figure 5. Comparison of summaries generated by human and system.

References

Kenneth W. Church. 1993. Char align: A program for aligning parallel texts at the character level. In *Proceedings of the Workshop on Very Large Corpora: Academic and Industrial Perspectives, ACL*. Association for Computational Linguistics, 1993.

Aidan Finn, Nicholas Kushmerick, and Barry Smyth. 2001. Fact or fiction: Content classification for digital libraries. In *Proceedings of NSF/DELOS Workshop on Personalization and Recommender*.

Pascale Fung and Kenneth W. Church. 1994. K-vec: A new approach for aligning parallel texts. In *Proceedings from the 15th International Conference on Computational Linguistics*, Kyoto.

Kevin Knight and Daniel Marcu (2000). Statistics-Based Summarization--Step One: Sentence Compression. *The 17th National Conference of the American Association for Artificial Intelligence AAAI'2000*, Outstanding Paper Award, Austin, Texas, July 30-August 3, 2000.

Julian Kupiec, Jan Pedersen, and Francine Chen. 1995. A trainable document summarizer. In *SIGIR '95, Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Seattle, Washington, USA., pages 68-73. ACM Press.

Chin-Yew Lin. 2001. Summary evaluation environment. http://www.isi.edu/~cyl/SEE.

Chin-Yew Lin and Eduard Hovy. 2000. The automated acquisition of topic signatures for text summarization. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING 2000)*, Saarbrücken, Germany, July 31- August 4, 2000.

Daniel Marcu. 1999. The automatic construction of large-scale corpora for summarization research. *The 22nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99)*, pages 137-144, Berkeley, CA, August 1999.

I. Dan Melamed. 1999. Bitext maps and alignment via pattern recognition. *Computational Linguistics*, 25(1):107-130.

Gerard Salton. 1975. A vector space model for information retrieval. *Communications of the ACM*, 18(11):613-620, November 1975.

David Zajic, Bonnie Dorr, and Richard Schwartz. 2002. Automatic headline generation for newspaper stories. In *Proceedings of the ACL-2002 Workshop on Text Summarization*, Philadelphia, PA, 2002.