Multi-document summarization using off the shelf compression software

Amardeep Grewal[1], **Timothy Allison**[2], **Stanko Dimitrov**[1], **Dragomir Radev**[1,3]
[1]Department of Electrical Engineering and Computer Science
[2]Department of Classical Studies
[3]School of Information
University of Michigan
{asgrewal,tballiso,sdimitro,radev}@umich.edu

Abstract

This study examines the usefulness of common off the shelf compression software such as gzip in enhancing already existing summaries and producing summaries from scratch. Since the gzip algorithm works by removing repetitive data from a file in order to compress it, we should be able to determine which sentences in a summary contain the least repetitive data by judging the gzipped size of the summary with the sentence compared to the gzipped size of the summary without the sentence. By picking the sentence that increased the size of the summary the most, we hypothesized that the summary will gain the sentence with the most new information. This hypothesis was found to be true in many cases and to varying degrees in this study.

1 Introduction

1.1 The connection between text compression and multidocument summarization

A standard way for producing summaries of text documents is sentence extraction. In sentence extraction, the summary of a document (or a cluster of related documents) is a subset of the sentences in the original text (Mani, 2001). A number of techniques for choosing the right sentences to extract have been proposed in the literature, ranging from word counts (Luhn, 1958), key phrases (Edmundson, 1969), naive Bayesian classification (Kupiec et al., 1995), lexical chains (Barzilay and Elhadad, 1997), topic signatures (Hovy and Lin, 1999) and cluster centroids (Radev et al., 2000).

Most techniques for sentence extraction compute a score for each individual sentence, although some recent work has started to pay attention to interactions between sentences. On the other hand, and particularly in multidocument summarization, some sentences may be redundant in the presence of others and such redundancy should lead to a lower score for each sentence proportional to the degree of overlap with other sentences in the summary. The Maximal Marginal Relevance (MMR) method (Carbonell and Goldstein, 1998) does just that.

In this paper, we are taking the idea of penalizing redundancy for multi-document summaries further. We want to explore existing techniques for identifying redundant information and using them for producing better summaries.

As in many areas in NLP, one of the biggest challenges in multi-document summarization is deciding on a way of calculating the similarity between two sentences or two groups of sentences. In extractive multi-document summarization, the goal is, on the one hand, to select the sentences which best represent the main point of the documents and, on the other, to pick sentences which do not overlap much with those sentences which have already been selected. To accomplish the task of sentence comparison, researchers have relied on stemming and counting n-gram similarity between two sentences. So, for example, if we have the following two sentences: “The dogs go to the parks” and “The dog is going to the park,” they would be nearly identical after stemming: “the dog [be] go to the park,” and any word overlap measure would be quite high (unigram cosine of .943).

In some ways, gzip can be thought of as a radical stemmer which also takes into account n-gram similarity. If the two sentences were in a file that was gzipped, the size of the file would be much smaller than if the second sentence were “A cat wanders at night.” (unigram cosine of 0). By comparing the size of the compressed files, we can pick that sentence which is most similar to what has already been selected for the summary (high compression ratio) or the most different (low compression ratio), depending on what type of summary we would prefer.

On a more information theoretic basis, as Benedetto et al. observe (Benedetto et al., 2002a), comparing the size of gzipped files allows us to roughly measure the distance (increase in entropy) between a new sentence and the already selected sentences. Benedetto et al. (Benedetto et al., 2002a) find that on their task of language classification, gzip's measure of information distance can effectively be used as a proxy for semantic distance. And so, we set out to see if we could usefully apply gzip to the task of multi-document summarization.

Gzip is a compression utility which is publicly available and widely used (www.gzip.org). Benedetto et al. (Benedetto et al., 2002a) summarize the algorithm behind gzip and discuss its relationship to entropy and optimal coding. Gzip relies on the algorithm developed by Ziv and Lempel (Ziv and Lempel, 1977). Following this algorithm, gzip reads along a string and looks for repeated substrings, if it finds a substring which it has already read, it replaces the second occurrence with two numbers, the length of the substring and the distance from that location back to the original string. If the substring length is greater than the distance, then the unzipper will know that the sequence repeats.

In our framework, we use an off-the-shelf extractive summarizer to produce a *base* summary. We then create a number of summaries containing precisely one more sentence than the base summary. If $|S|$ is the total number of sentences in the input cluster, and n is the number of sentences already included in the base, there are $|S| - n$ possible summaries of length $n + 1$ sentences. One of them has to be chosen over the others. In this work, we compress each of the $|S| - n$ candidate summaries and observe the relative increase in the size of the compressed file compared to the compressed base summary. The basic idea is that sentences containing the most new information will result in relatively longer compressed summaries (after normalizing for the uncompressed length of the newly added sentence). We will discuss some variations of this algorithm in the next section.

There are two issues which must be kept in mind in applying gzip to problems beyond data compression. First, because of the sequential nature of the algorithm, compression towards the beginning of the file will not be as great as that later in the file. Second, there is a 32k limit on the length of the window that gzip considers. So, if "abc" appears at the beginning of a string, and then also appears 33k later (but nowhere in between), gzip will not be able to compress the second appearance. This means that our process is "blind" to sentences in the summary which happen 32k earlier. This could potentially be a drawback to our approach, but in practice, given realistic text lengths, we have not found a negative effect.

The impetus for our approach is (Benedetto et al., 2002a; Benedetto et al., 2002b) who report on their use of gzip for language classification, authorship attribution, and topic classification. In their approach, they begin with a set of known documents. For each document, they measure the ratio of the uncompressed document to the compressed document. Then they append an unknown document to each known document cluster, and compress these new documents. Their algorithm then chooses whichever document had the greatest compression in relation to its original. As (Goodman, 2002) observes, using compression techniques for these tasks is not an entirely new approach, nor is it very fast. Nevertheless, we wanted to determine the efficacy of applying Benedetto et al.'s methods to the task of multi-document summarization.

2 Description of the method

The aim of this study was to determine if gzip is effective as a summarization tool when used in conjunction with an existing summarizer. We chose MEAD[1], a public-domain summarization system, which can be downloaded on the Internet (Radev et al., 2002). The version of MEAD used in this experiment was 3.07.

To produce a summary of a target length $n + 1$ sentences, we perform the following steps:

1. First, get MEAD to create a summary of size n sentences, where n is specified in advance. This summary will be called the *base* summary.
2. Compress the base summary using gzip. Let l_0 be the length of the base summary before compression and l'_0 be the size in bytes of its compressed version.
3. Create all possible summaries of length $n + 1$ using the remaining sentences in the input cluster.
4. Compress all summaries using gzip.
5. Pick the summary that results in the greatest increase in size in F, where F is one of a number of metrics, as described in the rest of this section.

Example: if a cluster had five sentences total, and a user wanted to create a summary of one sentence from MEAD and one from gzip, then the program would start with the one sentence generated by MEAD and add each of the four remaining sentences to make a total of five extracts. Four of these extracts would have two sentences and one would have the one sentence created by MEAD. After these extracts have been created they are converted to summaries and the number of characters in each summary is calculated. Then the difference in length between the summaries with the one extra sentence and the original MEAD-only summary is computed and stored. The

[1] `http://www.summarization.com`

next step in the process is to gzip all of the summaries and compute the difference in size between the summaries with the extra sentence and the original MEAD-only summary and store this change in size. After all these steps have been executed, we have a list of all possible sentences, the number of characters they contain and the size increase they produce after being gzipped with the rest of the summary. Based on this information, we can choose the next sentence in summary depending on which sentence increases the size of the gzipped summary the most or which sentence has the best size to length ratio.

We originally considered six evaluation metrics to use in this study. When choosing the next sentence for an existing summary, all possible sentences were added to the summary one at a time. For each sentence, the increase in length of the summary was measured and the increase in size of the gzipped summary was measured. From these two measurements we derived six policies. The top_sizes policy picked the sentence which produced the greatest increase in the size of the summary when gzipped. The bot_sizes policy picked the sentence which produced the smallest increase in the size of the summary when gzipped. The top_lengths policy picked the sentence that increased the number of characters in the summary the most. The bot_lengths picked the sentence that increased the number of the characters in the summary the least. The top_ratios picked the sentence that had the greatest (size increase)/(length increase) and the bot_ratios was the sentence that had the smallest (size increase)/(length increase). All policies except bot_ratios, top_lengths, and top_sizes did not show promising preliminary results and so are not included in this paper. In addition, the top_lengths policy does not really need gzip at all, and so it too is omitted from this paper. More information about the policies is given in the policies section.

2.1 The clusters used

We performed our experiments on a series of clusters. A cluster is a group of articles all pertaining to one particular event or story. There were a total of five such clusters, and the same set of tests was carried out on each cluster independently from the others. All of our tests were conducted on five different clusters of documents, referred to here as the 125 cluster, 323 cluster, 46 cluster, 60 cluster and 1018 cluster. The lengths of each of these clusters in sentences was 232, 91, 344, 150, and 134, respectively. Clusters with such diverse lengths were purposely chosen to determine if the quality of the summaries was in any way related to the length of the source material. The various articles were taken from the Hong Kong News corpus provided by the Hong Kong SAR of the People's Republic of China (LDC catalog number LDC2000T46). This paper contains 18,146 pairs of parallel documents in English and Chinese, in our case only the English ones were used. The clusters were created at the Johns Hopkins University Summer Workshop on Language Engineering 2002.

2.2 An example

Figure 1 shows a 5-sentence summary produced by MEAD from Cluster 125 of the HK News Corpus. The uncompressed length of this summary is 797 bytes whereas its size after gzip compression is 451 bytes.

(1) To ensure broadly the same registration standards to be applied to all drug treatment and rehabilitation centres, Mrs Lo said the proposed registration requirements to be introduced for non-medical drug treatment and rehabilitation centres would be similar to those provisions of Ca.165 which currently apply to medical drug treatment and rehabilitation centres.

(2) Youths-at-Risk of Substance Abuse and Families of Abusers Given Priority in This Year's Beat Drugs Fund Projects

(3) he Action Committee Against Narcotics (ACAN) Research Sub-committee has decided to commission two major research on treatment and rehabilitation for drug abusers in Hong Kong in 1999.

(4) New Initiatives Despite Fall in Number of Reported Drug Abusers

(5) Beat Drugs Fund Grants $16 million in Support of 29 Anti-Drug Projects

Figure 1: "Base" MEAD summary consisting of five sentences.

Cluster 125 includes 10 documents with a total of 232 sentences. In our example, after five of them have already been included in the 5-sentence summary, there are still 227 candidates for the sixth sentence to include in a 6-sentence summary. As in the rest of the paper, we will be comparing summaries of equal length produced by two different methods, either (a) all sentences are chosen by MEAD, or (b) some sentences are chosen by MEAD and then the rest of the sentences until the target length of the summary are added by gzip.

Figure 2 shows some statistics about these 227 sentences.

Figure 3 contains the list of sentences included in the five-sentence base summary.

Figure 4 shows the candidate sentences to be included by the different policies in their six-sentence extracts.

3 Experimental setup

To test the benefit of gzip in the summarization process, extracts were created using a combination of MEAD and gzip. These extracts contained pointers to the actual sentences that would be included in the summary, but not the sentences themselves. A number of extracts were created with varying amounts of sentences per extract. For these

DOCUMENT	SENTENCE	LENGTHORIG+1	SIZEORIG+1AFTGZ	DELTALENGTH	DELTASIZE	RATIO
D-19990729_008	1	847	505	50	54	1.08
D-19990729_008	2	1014	573	217	122	0.56
D-19990729_008	3	1200	664	403	213	0.53
D-19990729_008	4	1039	601	242	150	0.62
D-19990729_008	10	1012	579	215	128	0.60
D-19990729_008	5	1006	588	209	137	0.66
D-19990729_008	11	1064	600	267	149	0.56
D-19990729_008	6	999	580	202	129	0.64
D-19990729_008	12	942	533	145	82	0.57
D-19990729_008	7	922	541	125	90	0.72
D-19990729_008	13	1102	629	305	178	0.58
D-19990729_008	8	1112	621	315	170	0.53
D-19990729_008	14	1008	570	211	119	0.56
D-19990729_008	9	930	543	133	92	0.69
D-19990729_008	15	926	542	129	91	0.71
D-19990729_008	16	1013	578	216	127	0.59
D-19990729_008	17	938	547	141	96	0.68
D-19980430_016	37	927	512	130	61	0.47
D-19980430_016	38	1071	570	274	119	0.43
D-19980430_016	39	962	550	165	99	0.60
D-19980430_016	20	1162	625	365	174	0.48
D-19980430_016	21	883	503	86	52	0.60
D-19980430_016	22	878	520	81	69	0.85
D-19980430_016	23	1019	564	222	113	0.51
D-19980430_016	40	951	563	154	112	0.73
D-19980430_016	24	915	502	118	51	0.43
D-19980430_016	41	944	538	147	87	0.59
D-19980430_016	25	988	555	191	104	0.54
D-19980430_016	42	905	537	108	86	0.80
D-19980430_016	26	1005	570	208	119	0.57
D-19980430_016	27	977	541	180	90	0.50
D-19980430_016	1	864	499	67	48	0.72
D-19980430_016	2	971	534	174	83	0.48
D-19980430_016	3	849	505	52	54	1.04
D-19980430_016	10	924	543	127	92	0.72
...						

Figure 2: A subset of the 227 candidate sentences (from two documents out of a total of ten) to be included as sentence number six in a six-sentence summary. LENGTHORIG is the length in bytes of the summary, consisting of the original five MEAD-generated sentences plus this candidate sentence, before compression. SIZEORIG+1AFTGZ is the length in bytes of the compressed summary. DELTALENGTH is the difference in uncompressed length (which is also the length of the candidate uncompressed sentence). DELTASIZE is the change in compressed size. RATIO is equal to DELTASIZE divided by DELTALENGTH.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE EXTRACT SYSTEM
"/clair/tools/mead/dtd/extract.dtd">

<EXTRACT QID="" LANG="" COMPRESSION="" SYSTEM=""
RUN="">
<S ORDER="1" DID="D-19980430_016.e" SNO="17" />
<S ORDER="2" DID="D-19990425_009.e" SNO="1" />
<S ORDER="3" DID="D-19990829_012.e" SNO="2" />
<S ORDER="4" DID="D-19990927_011.e" SNO="1" />
<S ORDER="5" DID="D-20000408_011.e" SNO="1" />
</EXTRACT>
```

Figure 3: The list of sentence/document IDs for the five sentences in the base summary.

extracts, the number of sentences contributed by MEAD was incremented by ten starting at zero and the number of sentences contributed by gzip was incremented from one to ten, on top of the MEAD sentences. So for any randomly chosens extract of size S, $|S| \pmod{10}$ indicates the number of sentence contributed by gzip. So an extract of fifty-six sentences contains fifty sentences from MEAD and six from gzip. In this way, a total of 110 extracts were created for all clusters except Cluster 323, for which only 80 extracts were created because there were only 91 sentences total in that cluster. For clarification, the 110 sentence extract for each cluster contained 100 MEAD sentences and 10 sentences from the chosen gzip policy. The 10 sentence extract for each cluster contained 0 MEAD sentences and 10 sentences from the chosen gzip policy. In order to have a benchmark to compare the gzip modified extracts to, extracts containing an identical number of sentences were created using only MEAD, so a 110 MEAD extract has all of its sentences chosen by MEAD. Relative utility was run on all types of gzip extracts, as well as only MEAD extracts.

3.1 Evaluation methods

We use the Relative Utility (RU) method (Radev et al., 2000) to compare our various summaries. To calculate RU, human judges read through all sentences in a document cluster and then give scores, from 1 (totally irrelevant) to 10 (central to the topic) to each sentence based on their impression of the importance of each sentence for a summary of the documents. Each judge's score is then normalized by his or her other scores. Finally, for each sentence, the judges' scores are summed and normalized again by the number of judges. Then a final score is given for a summary by summing the utility score for each sentence which was in the summary and then factoring in the

bot_lengths.extract

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE EXTRACT SYSTEM
"/clair/tools/mead/dtd/extract.dtd">

<EXTRACT QID="125" LANG="ENG"
COMPRESSION="" SYSTEM="MEADORIG"
RUN="Mon Mar 24 18:34:38 2003">
<S ORDER="1" DID="D-19980430_016.e" SNO="17" />
<S ORDER="2" DID="D-19990425_009.e" SNO="1" />
<S ORDER="3" DID="D-19990802_006.e" SNO="1" />
<S ORDER="4" DID="D-19990829_012.e" SNO="2" />
<S ORDER="5" DID="D-19990927_011.e" SNO="1" />
<S ORDER="6" DID="D-20000408_011.e" SNO="1" />
</EXTRACT>

bot_ratios.extract

...
<S ORDER="2" DID="D-19980430_016.e" SNO="12" />
...

bot_sizes.extract

...
<S ORDER="3" DID="D-19990802_006.e" SNO="1" />
...

top_lengths.extract

...
<S ORDER="3" DID="D-19990425_009.e" SNO="7" />
...

top_ratios.extract

...
<S ORDER="1" DID="D-19980306_007.e" SNO="16" />
...

top_sizes.extract

...
<S ORDER="3" DID="D-19990425_009.e" SNO="7" />
...

Figure 4: The document/sentence ID picked by each of the six policies to be the sixth sentence in the summary.

upper bound (highest utility scores given by the judges) and lower bound (utility scores from randomly chosen sentences). We use this method because, as (Radev et al., 2002) find, Precision, Recall, and Kappa measures as well as content-based evaluation methods are unreliable for short summaries (5%-30%) and especially in the task of multi-document summarization, where there are likely to be several sentences which would contribute the same information to a summary.

4 Results

4.1 Performance of Bot_Ratios

When this project was in its initial stages, the ratios policy was designed in the hope that it would produce the highest quality sentences. However, it was not the bot_ratios policy which was expected to succeed, but the top_ratios. Top_ratio sentences are ideally the sentences which provide the greatest increase in gzip size, for the smallest increase in summary length. Logically, these are the sentences that would appear to enhance the summary the most for the smallest cost. Bot_ratio sentences are essentially the sentences which provide the greatest increase in summary length, for the smallest increase in size. In many cases, they are simply the longest sentences remaining to be used. The bot_ratios policy was originally

bot_lengths.summary

...
(3) Anti-drug work poses challenge
...

bot_ratios.summary

...
(2) Taking into account these observations, the Government proposes and the Action Committee Against Narcotics supports that a registration scheme should be introduced for non-medical drug treatment and rehabilitation centres, in order to:
...

bot_sizes.summary

...
(3) Anti-drug work poses challenge
...

top_lengths.summary

...
(3) Notable amongst the approved projects for youths-at-risk are the $2 .5 million proposal to be organised by the Hong Kong Federation of Youth Groups featuring preventive education and guidance for 2 500 high-risk youths from primary and secondary schools , as well as from youth centres in Tsuen Wan and Kwai Tsing districts; and the $2 .3 million project by the Hong Kong Christian Service targeting at 3 000 youths-at-risk , including school drop-outs and unemployed young people , with a view to minimising their exposure to social and moral danger which could lead to substance abuse .
...

top_ratios.summary

...
(1) The study will be completed by 2000 .
...

top_sizes.summary

...
(3) Notable amongst the approved projects for youths-at-risk are the $2 .5 million proposal to be organised by the Hong Kong Federation of Youth Groups featuring preventive education and guidance for 2 500 high-risk youths from primary and secondary schools , as well as from youth centres in Tsuen Wan and Kwai Tsing districts; and the $2 .3 million project by the Hong Kong Christian Service targeting at 3 000 youths-at-risk , including school drop-outs and unemployed young people , with a view to minimising their exposure to social and moral danger which could lead to substance abuse .
...

Figure 5: The sentence picked by each of the six policies to be the sixth sentence in the summary. The number in parentheses shows where in the summary this sentence will be added. For example, the first policy, bot_lengths, would insert the short sentence "Anti-drug work poses challenge" between sentences 2 and 3 of the based five-sentence summary.

included in this study only to confirm our initial expectations that the sentence with the smallest (increase in size) / (increase in length) will not improve the summary a great deal. However, we were surprised to find that our expectations for this policy were false. Upon examining the experimental results, it was found that the bot_ratios policy, which is essentially picking the longest sentence in most cases, actually outperformed the existing summarizer by a considerable margin. Although this policy does not prove anything about the use of gzip in summarization, the surprising nature of its performance is certainly worth noting. Figure 6 shows scores for summaries cre-

ated using bot_ratios, top_sizes and scores for summaries created using only MEAD.

Cluster	Avg.MEAD	Avg.Top_Sizes	Avg.Bot_Ratios
46	0.83205	0.83428	0.83604
60	0.77109	0.77382	0.76641
125	0.79931	0.78399	0.76606
323	0.79569	0.77731	0.81034
1018	0.84819	0.83244	0.84417
Average	0.80306	0.80169	0.80427

Figure 6: Average Relative Utility Scores

As is indicated in Figure 6, gzip's bot_ratios policy outperformed MEAD by a significant margin in Cluster 323. There is an explanation for these scores which takes into account the fact that the top_sizes policy had a lower score than MEAD for this cluster. In a cluster of documents, many of the short sentences are the most repetitive ones, usually simply stating the event that occurred or subject of the document and not containing any extraneous information. Most often it is the longer sentences which provide the extra information which makes for rich summaries. Since the ratio being used in this evaluation is size/length, many of the smaller sentences may have been eliminated from being chosen because of reasons mentioned above. This leaves only the longer sentences to choose from. Since the length of most sentences is far greater than the size increase when gzipped, it makes sense that most remaining sentences would have very low ratio scores. In a larger cluster, many of the sentences subsume each other since there are so many similar sentences, but in a small cluster such as 323 there is a great deal less subsumption. If gzip is picking sentences based on the bot_ratios policy, normally it would pick many sentences that were very similar because the bot_ratios policy relies on a greater sentence length as criteria for selection and the small change in gzip size provided by similar sentences would only lower the ratio for a potential sentence even more. However, since there is less repetition in a small cluster, the bot_ratios policy ends up picking sentences which are more different from each other than in a larger cluster. These findings are quite surprising and do not agree with our expectations. The ratio policies were intended to balance the fact that larger sentences will obviously contain more information. The bot_ratios relative utility scores however indicated that choosing larger sentences resulted in better summaries, with the exception of the 125 cluster. This contradicts the view that the sentence with the greatest increase in gzip size is better suited for a summary. The possible reasons for this contradiction are discussed in the next section.

4.2 Clusters and their sizes

Figure 7 shows the size of each cluster in sentences and indicates whether the top_sizes policy performed better than the bot_ratios policy.

Cluster	Length	Better Policy
46	344	equal
60	150	top_sizes
125	232	top_sizes
323	91	bot_ratios
1018	134	bot_ratios

Figure 7: Best Policy vs. Cluster Size

One of the reasons that the bot_ratios policy outscored the top_sizes policy in two out of five clusters may be that the sample size in the clusters in which bot_ratios outperformed top_sizes was not large enough. This is illustrated by examining Clusters 125 and 46. In these clusters, the top_sizes policy and bot_ratios policy were either virtually identical or top_sizes outperformed the bot_ratios by a considerable margin. It is worth noting that Cluster 46 was by far the largest used in this study at 344 sentences and Cluster 125 was the second largest at 232 sentences. The third largest was Cluster 60 with 150 sentences, in which top_sizes also beat bot_ratios. The fact that the top_sizes policy outscored the bot_ratios in these clusters indicates that although in smaller clusters, a larger length indicates a better candidate due to decreased repetition, in a large cluster the sentences with larger length are quite repetitive and picking a sentence based on gzipped size is far more effective for summarization.

This principle is illustrated on a smaller scale when examining the 46 cluster. In the first fifty extracts, the gzip bot_ratios policy outscores the top_sizes policy forty times. However, in the last 60 extracts, bot_ratios outscored top_sizes a mere five times. This indicates that early on the sentence with the longest length contains the most useful information, but as the size of the extract increases, the longer sentences start to become repetitive and therefore decrease the quality of the extract. One solution to this disparity between large and small clusters would be to alter how sentences are chosen based on cluster size or the size of the existing summary. If the cluster or summary was a small one, first all the sentences with the top lengths would be grouped, and of those the sentence with the highest gzip size would be chosen. If the cluster or summary was large, the sentence could be chosen on gzip size alone. Figure 8 is a table indicating scores for both policies and MEAD for this first and last ten sentences of each cluster. For all the clusters with the exception of 125, our hypothesis was correct. The top_sizes method was better in the larger last extracts and the bot_ratios prevailed early on in the small 1-10 sentence extracts.

4.3 Initial Size and RU Scores

Since the sentence that the gzip top_sizes policy chooses is based on the amount of information that already exists

Cluster	Top_Sizes	Bot_Ratios	Greater
46 First 10	0.65447	0.66983	Bot_Ratios
46 Last 10	0.87376	0.86959	Top_Sizes
60 First 10	0.78664	0.90190	Bot_Ratios
60 Last 10	0.83311	0.82783	Top_Sizes
125 First 10	0.72993	0.51713	Top_Sizes
125 Last 10	0.82560	0.82849	Bot_Ratios
1018 First 10	0.67481	0.74116	Bot_Ratios
1018 Last 10	0.89436	0.89367	Top_Sizes
Average First 10	0.71146	0.70750	Top_Sizes
Average Last 10	0.85671	0.85487	Top_Sizes

Figure 8: Top_Sizes vs. Bot_Ratios

in the summary, the quality of sentences chosen should depend on the amount of existing information. Therefore as the size of the extracts increases, the relative-utility scores should also increase for the top_sizes policy. However there is also a general trend in which all relative utility scores increase as a function of extract size. So in order to determine if the top_sizes policy is working correctly, we can compare the difference between MEAD and top_sizes for the first twenty and the last twenty sentences of each extract and the difference should be greater for the last twenty.

Cluster	MEAD	Top_Sizes	Difference
46 First 20	0.80355	0.71992	-0.08364
46 Last 20	0.85816	0.87376	0.01559
60 First 20	0.70489	0.78664	0.08175
60 Last 20	0.85344	0.83311	-0.02034
125 First 20	0.82665	0.72993	-0.09673
125 Last 20	0.82617	0.82560	-0.00057
323 First 20	0.75653	0.66482	-0.09171
323 Last 20	0.83109	0.84960	0.01850
1018 First 20	0.84229	0.74229	-0.10000
1018 Last 20	0.89547	0.89436	-0.00111

Figure 9: Scores vs. Size of Base Extract

In four out of five cases (Figure 8), the top_sizes policy behaved as it should have, increasing performance with increasing size. In the one case of cluster 60 where the performance over MEAD actually decreased as size of extract increased, it should be noted that MEAD improved more in this cluster than any other cluster. So although the top_sizes policy still improved with regard to extract size, it could not improve as quickly as MEAD in that one cluster.

5 Conclusion

Overall, there were many instances when gzip outperformed MEAD. These mainly occurred after the first ten sentences because for the first ten sentences gzip had very little preliminary data to use in choosing the next sentence. Figure 11 lists how many times each policy beat MEAD after the first ten sentences of each cluster and the number of times that MEAD beat both gzip policies.

Cluster	MEAD	Top_Sizes	Bot_Ratios
46	0/100	96/100	93/100
60	77/100	23/100	4/100
125	50/100	33/100	45/100
323	14/80	25/80	66/80
1018	61/100	16/100	38/100

Figure 11: Frequency of higher gzip scores

The general trend was that both gzip policies out performed MEAD in medium length summaries between 20-60 sentences. Furthermore, the top_sizes policy outperformed MEAD more so in large summaries usually with 100+ sentences.

A note on performance. Although theoretically interesting, our method is too slow for practical use in fast paced summarization systems. It takes time roughly proportional to the size, N of the summary desired. The bottleneck in this process is of course, the gzipping process.

5.1 Future Work

These results indicate that gzip can be used to enhance summaries or even produce large summaries from scratch. One metric lacking in our measurements is that of subsumption. If subsumption data were available for each of the clusters used, it would most likely favor gzip summaries as being more accurate because the gzip algorithm is designed to remove the very repetitiveness which subsumption measures. Further work remains to be done on other clusters of various sizes and redundancy as well as with other summarization metrics, such as content based metrics (cosine, overlap, longest-common substring, etc.). Nevertheless, we have established the potential benefits for applying gzip to the task of multi-document summarization.

6 Acknowledgments

This work was partially supported by the National Science Foundation's Information Technology Research program (ITR) under grant IIS-0082884. Our thanks go to Ammar Qusaibaty for help with this project as well as the anonymous reviewers for their very helpful comments.

References

Regina Barzilay and Michael Elhadad. 1997. Using Lexical Chains for Text Summarization. In *Proceedings of the ACL/EACL'97 Workshop on Intelligent Scalable Text Summarization*, pages 10–17, Madrid, Spain, July.

Dario Benedetto, Emanuele Cagliot, and Vittorio Loreto. 2002a. Language trees and zipping. *Physical Review Letters*, (4).

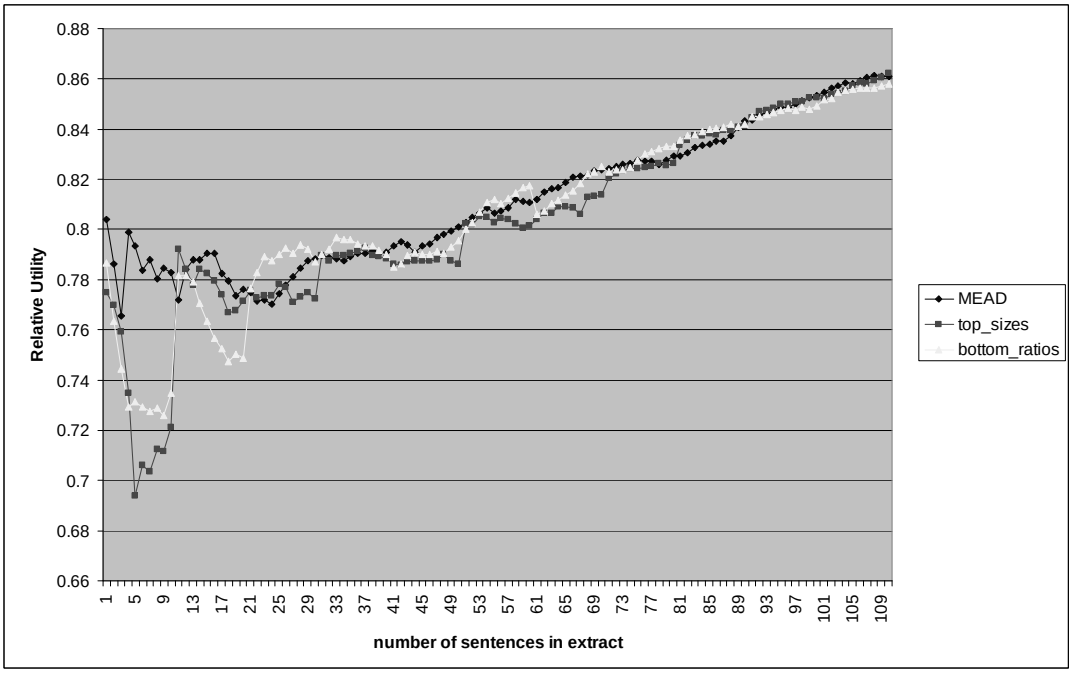

Figure 10: The results across all clusters for the Pure MEAD summaries, and the gzip policies top_sizes and bot_ratios. In shorter length summaries MEAD outscores the gzip policies but as the number of sentences in the summary increase, the gzip policies scores increase enough to be competitive and sometimes better than MEAD.

Dario Benedetto, Emanuele Cagliot, and Vittorio Loreto. 2002b. On J. Goodman's comment to Language Trees and Zipping. *cmp-lg preprint archive*.

Jaime G. Carbonell and Jade Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Research and Development in Information Retrieval*, pages 335–336.

H.P. Edmundson. 1969. New Methods in Automatic Extracting. *Journal of the Association for Computing Machinery*, 16(2):264–285, April.

Joshua Goodman. 2002. Extended comment on language trees and zipping. http://arxiv.org/abs/cond-mat/0202383.

Eduard Hovy and Chin-Yew Lin. 1999. Automated Text Summarization in SUMMARIST. In I. Mani and M.T. Maybury, editors, *Advances in Automatic Text Summarization*, pages 81–94. The MIT Press.

Julian Kupiec, Jan O. Pedersen, and Francine Chen. 1995. A trainable document summarizer. In *Research and Development in Information Retrieval*, pages 68–73.

H.P. Luhn. 1958. The Automatic Creation of Literature Abstracts. *IBM Journal of Research Development*, 2(2):159–165.

Inderjeet Mani. 2001. *Automatic Summarization*. John Benjamins Publishing Company, Amsterdam/Philadephia.

Dragomir R. Radev, Hongyan Jing, and Malgorzata Budzikowska. 2000. Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies. In *ANLP/NAACL Workshop on Summarization*, Seattle, WA, April.

Dragomir Radev, Simone Teufel, Horacio Saggion, Wai Lam, John Blitzer, Arda Çelebi, Hong Qi, Elliott Drabek, and Danyu Liu. 2002. Evaluation of text summarization in a cross-lingual information retrieval framework. Technical report, Center for Language and Speech Processing, Johns Hopkins University, Baltimore, MD, June.

J. Ziv and A. Lempel. 1977. A Universal Algorithm for Sequential Data Compression. *IEEE Transactions on Information Theory*, 23(3):337–343.