Navigating Through Summary Space - Selecting Summaries, Not Sentences

Martin Hassel and Jonas Sjöbergh
KTH CSC
Royal Institute of Technology
100 44 Stockholm, Sweden
{xmartin,jsh}@nada.kth.se

Abstract

We present a novel method for extraction based summarization using statistical lexical semantics. It attempts to give an overview by selecting the summary most similar to the source text from a set of possible candidates. It evaluates whole summaries at once, making no judgments on for instance individual sentences. A simple greedy search strategy can be used to search through a space of possible summaries. Starting the search with the leading sentences of the source text is a powerful heuristic, but we also evaluate other search strategies. The aim has been to construct a summarizer that can be quickly assembled, with the use of only a very few basic language tools. The proposed method is largely language independent and can be used even for languages that lack large amounts of structured or annotated data, or advanced tools for linguistic processing. When evaluated on English abstracts from the Document Understanding Conferences it performs well, though better language specific systems are available. It performs better than several of the systems evaluated there, but worse than the best systems. We have also evaluated our method on a corpus of human made extracts in Swedish. It performed poorly compared to a traditional extraction-based summarizer. However, since these man-made extracts were not produced to reflect the whole contents of the texts, but rather to cover only the main topic, this was expected.

1 Introduction

Summaries are an important tool when familiarizing oneself with a new subject area. They are also essential when deciding whether reading a document in whole is necessary or not. In other words, summaries save time in daily life and work. To write a summary of a text is a non-trivial process. The contents of the text itself should be analyzed and the most central information should be extracted. The intended readers should also be considered, taking into account what knowledge they already have, possible special interests and so on. Today numerous documents, papers, reports and articles are available in digital form, most of which lack summaries. The information is often too abundant for it to be possible to sift through it manually and choose what information to acquire. The information must instead be automatically filtered and extracted to avoid drowning in it.

Automatic text summarization is a technique where a computer summarizes a text. A text is given to the computer and the computer returns a shorter, less redundant extract of the original text. So far automatic text summarization has not yet reached the quality possible with manual summarization, where a human interprets the text and writes a completely new shorter text with new lexical and syntactic choices, and may never do. However, automatic text summarization is untiring, consistent and always available.

1.1 Language Independent Automatic Text Summarization

Today most research in automatic text summarization is focused on knowledge rich, and in practice language specific, approaches using tools and annotated resources simply not available for many languages. Justifiably so, these knowledge rich systems do in general perform better than earlier knowledge poor approaches. It is however easy to see that there is a clear need for automatic summarization also for the languages less in focus in this research area than the major European, Asian or Mid Eastern languages.

The experiments reported herein concern an attempt to develop such a method for largely language independent automatic text summarization. The aim has been to construct a summarizer that can be quickly assembled, with the use of only a few basic language tools, for languages that lack large amounts of structured or annotated data or advanced tools for linguistic processing. We try to accomplish this by trying to capture the essence of a document being summarized. For this we use computational semantics by first building semantic, or conceptual, representations for each word based on a large free-text corpus. Simply put, a word space. These conceptual representations in turn are then used to build a document space where a set of summaries can be evaluated against the original text.

2 Word Spaces

Word space models, most notably Latent Semantic Analysis/Indexing (Deerwester *et al.* 1990, Landauer *et al.* 1998), enjoy considerable attention in current research on computational semantics. Since its introduction in 1990 Latent Semantic Analysis (LSA) has more or less spawned an entire research field with a wide range of word space models as a result, and numerous publications reporting exceptional results in many different tasks, such as information retrieval, various semantic knowledge tests such as the TOEFL test (Educational Testing Service 2006), text categorization and word sense disambiguation.

The general idea behind word space models is to use statistics on word distributions in order to generate a high-dimensional vector space. In this vector space the words are represented by context vectors whose relative directions are assumed to indicate semantic similarity. The basis of this assumption is the *distributional hypothesis* (Harris 1968), according to which words that occur in similar contexts also tend to have similar properties (meanings/functions). From this follows that if we repeatedly observe two words in the same, or very similar, contexts, then it is not too far fetched to assume that they also mean similar things (Sahlgren 2006).

2.1 Random Indexing

In the major part of the experiments herein we have employed Random Indexing (Sahlgren 2005), which presents an efficient, scalable and inherently incremental alternative to standard word space methods. As an alternative to LSA-like models that first construct a huge cooccurrence matrix and then use a separate dimension reduction phase, Random Indexing (RI) instead accumulates context vectors on-the-fly based on the occurrence of words (tokens) in contexts, without a need for a separate dimension reduction phase.

The construction of context vectors using RI can be viewed as a two-step operation. First, each token in the data is assigned a unique and (usually) randomly generated label. These labels can be viewed as sparse, high-dimensional, and ternary vectors.[1] Their dimensionality (d) is usually chosen to be in the range of a couple of hundred up to several thousands, depending on the size and redundancy of the data. They consist of a very small number, usually about 1-2%, of randomly distributed +1s and -1s, with the rest of the elements of the vectors set to 0.

Next, the actual context vectors are produced by scanning through the text and each time a token w occurs in a context (e.g. in a document or paragraph, or as a

[1]The extremely sparse random labels are handled internally as short lists of positions for non-zero elements, and are generated on-the-fly whenever a never before seen token is encountered in the context during indexing.

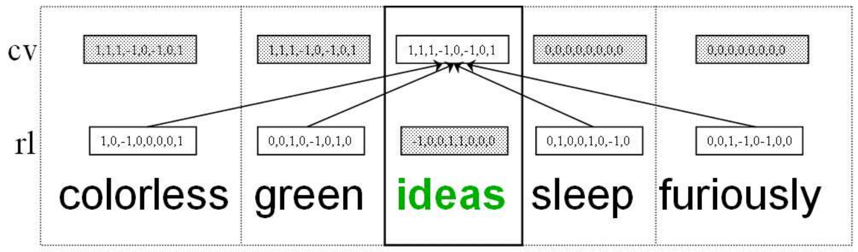

Figure 1: A Random Indexing context window focused on the token "ideas", taking note of the cooccurring tokens. The row marked as `cv` represents the continuously updated *context vectors*, and the row marked as `rl` the static *random labels* (acting as addable meta words). Grayed-out fields are not involved in the current token update.

word within a sliding context window), that context's d-dimensional random label is added to the context vector for the token w. We use a sliding context window, i.e. all tokens that appear within the context window contribute to some degree with their random labels to the context vector for w. Words are in this way effectively represented by d-dimensional context vectors that are the sum of the random labels of the cooccurring words, see Figure 1. When using a sliding context window it is also common to use some kind of distance weighting in order to give more weight to tokens closer in context.

This technique can readily be used with any type of linguistic context and can be used to index using a more traditional bag-of-words approach as well as using a sliding context window (i.e. cooccurrence between tokens) capturing sequential relations between tokens. These tokens can be the word simply represented by its lexical string or its lemma, or more elaborate approaches utilizing tagging, chunking, parsing or other linguistic units can be employed.

One of the strengths of Random Indexing is that we can in a very elegant way fold the document currently being processed into the Random Index, thus immediately taking advantage of, possibly genre or text type specific, distributional patterns within the current document. Apart from the advantage of eliminating the risk of lack of data due to unknown words, we also have a system that learns over time. The problem of sparse data cannot be completely avoided, since a never before seen word will only have as many contextual updates as the number of times it occurs in the current document. This is however far better than no updates at all.

As with all LSA-like models Random Indexing needs, for good performance, large amounts of text (millions of words) when generating the conceptual representations. Since Random Indexing is resource lean and only requires access to raw (unannotated) text, this is generally not a problem.

There are a few implementations of Random Indexing available. We used a freely available tool-kit called JavaSDM (Hassel 2006). It should be noted that the proposed method, at least in theory, could employ any word space model, such as LSA or Hyperspace Analogue to Language (Lund *et al.* 1995), albeit waiving some of the benefits of using RI in this context.

3 Experimental Setup

The main part of these experiments have been carried out for English. Mainly because there is a large amount of reference summaries and evaluation schemes developed for this language, as well as several other summarization systems to use as reference points. For English we build our conceptual representations for each word based on a large corpus, the British National Corpus (Burnard 1995), as well as the documents themselves as they are being summarized. The data being used for building these representations thus is comprised of 100 million words from BNC and roughly 2 million words contained in 291 document sets provided for DUC 2001-2004 (DUC 2007). After stop word filtering and stemming this results in almost 290,000 unique stems taken from 4415 documents.

A minor experiment has also been carried out for Swedish in order to test the thesis of language independence. One must however keep in mind that the obvious lack of suitable and fairly large evaluation corpora render these results less reliable than their English counterparts. These results are nevertheless reported below.

3.1 Preliminary Experiment: Selecting Sentences

The first approach in our series of experiments was to build a context vector for each extraction unit, in this case each sentence, in the text being summarized. This was done by adding the context vectors for each token (word) in each individual sentence. This was also done for the complete text. All sentence vectors were then compared for similarity using the cosine angle between each sentence vector and the document vector, and the closest match was chosen. The words in the chosen sentence were then temporarily removed from the remaining sentences and their respective content vectors recalculated, and the closest match again chosen for inclusion in the summary. This procedure was repeated until the summary reached the desired length.

Different weighting and normalization schemes were tested, for example sentence length normalization and only counting each occurrence of a word in a sentence once. None of these strategies did however beat the chosen baseline summary - the first *N* sentences up to the desired summary length.[2]

This approach does, in practice, not differ particularly from most traditional extractive summarization approaches in the respect that it ranks individual extract segments for inclusion in the concatenated summary. Another criteria for selecting extraction units, using our measure of semantic similarity, was clearly in need.

3.2 Selecting Summaries: The Basic Method for English

After the preliminary experiment, we instead focused on finding summaries of a given length that are as similar to the original texts as possible. This method would aim at producing overview summaries. One way to accomplish this would be to generate all possible extracts and see which one is most similar to the original text. Besides being computationally cumbersome, the difficulty here lies in judging how similar two texts are. Most methods that compare two documents use measures like word or n-gram overlap. Since all candidate summaries here are extracts from the original text, all words in all summaries overlap with the original text. This is thus not a good way to differentiate between different candidates.

3.2.1 Evaluating Candidate Summaries

Our method makes use of Random Indexing to differentiate between different summaries. As described above, Random Indexing gives each word a context vector that in some sense represents the semantic content of that word, as defined by its use. We make use of these vectors when calculating a measure of similarity between two texts. Each text is assigned its own vector for semantic content, which is simply the (weighted) sum of all the context vectors of the words in the text. This can be seen as projecting the texts into a high-dimensional vector space where we can relate the texts to each other. Similarity between two texts is then measured as the similarity between the directions of the semantic vectors of the texts, in our case between the vector for the full text and the vectors for each of the candidate summaries. Similar approaches have also been applied to for instance text categorization (Sahlgren and Cöster 2004).

When constructing the semantic vector for a text, the context vector for each word is weighted with the term frequency and the inverse document frequency, by making the length of the vector be $tf \cdot \log(idf)$. If desired, other weighting criteria can easily be added, for instance for slanted or query based summaries where some words are deemed more important, or by giving words occurring early in the document, in document or paragraph headings etc. higher weight.

[2]This baseline summary is often referred to as *lead*.

Words in a text that have never been encountered during the calculation of a word space representation generally degrade the performance, since no information regarding their distributional properties is available at run-time. Since RI allows for continuous updates this is here trivially solved by simply adding the new text to the index immediately before summarizing it. This means in effect that all words in the relevant texts will have been encountered at least once.

Also, since our method does not give any consideration to the position in the text a sentence is taken from (though that is possible to do if one so wishes), it is relatively straightforward to use for multidocument summarization as well. In fact, some of the reference summaries in the English evaluation corpus have been built from multiple news texts covering the same event. In this case we have used the same set of source documents concatenated into one single document sent to the summarizer.

In the following section we present an extraction based technique to generate a set of summary candidates. However, the method for differentiating between the summary candidates does not require that the candidates consist solely of segments from the source text. Since the comparison of the semantic vectors does not measure lexical or syntactic similarity, but attempts to optimize semantic similarity between the summary and the text being summarized, the summary candidates could in practice be generated by any means, even being man-made.

3.2.2 Finding a Better Summary

To find a good summary we start with one summary and then try to see if there is another summary that is “close” in some sense, that is also a better summary. Better in this context means more similar to the original text. The reason we do not exhaustively pursue the best summary of all possible summaries is that there are exponentially many possible summaries. Comparing all of them to the original text would thus not be feasible even for documents with fairly few extraction units (in our case sentences).

It has been shown that the leading sentences of an article, especially within the news domain, are important and constitute a good summary (Zechner 1996). Therefore, the “lead” summary, i.e. the first sentences from the document being summarized up to a specified length, was used in our experiments both as a baseline and as the starting point for our search for a better summary. When used for multidocument summarization we simply take the concatenated set of documents covering the same topic as source text and the leading sentences of the top-most document as the starting point.

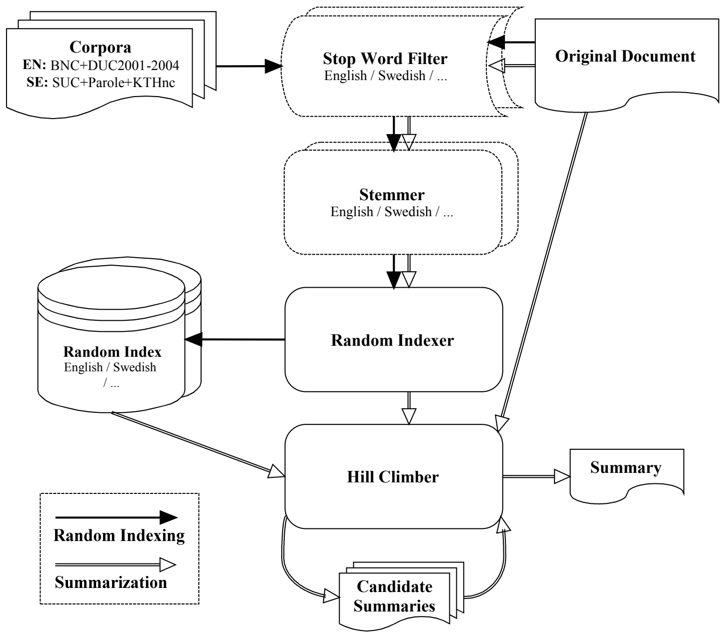

Figure 2: HolSum system layout. The candidate summaries are iteratively generated and evaluated (i.e. compared for semantic similarity against the original document).

Using a standard hill-climbing algorithm we then investigate all neighbors, looking for a better summary. The summaries that are defined as neighbors to a given summary are simply those that can be created by removing one sentence and adding another. Since sentences vary in length we also allow removing two sentences and adding one new, or just adding one new sentence. This allows for optimizing the summary size for the specified compression rate.

When all such summaries have been investigated, the one most similar to the original document is updated to be the currently best candidate and the process is repeated. If no other summary is better than the current candidate, the search is terminated. It is also possible to stop the search at any time if so desired, and return the best candidate so far. A schematic layout of the complete system can be found in Figure 2.

In our experiments on the texts provided for the Document Understanding Conferences (DUC 2007) the generated summaries are very short, about three sentences. This means that there are usually quite few, typically around five, search

```
Azerbaijani President Heydar Aliyev, who is considered the
most likely to win the presidential elections, cast his
vote today, Sunday, at one of the polling centers near
his residence in the center of the capital and took the
opportunity to attack his main opponent, Etibar Mammadov.
The president, who was elected in September 1993, said in a
statement to reporters that "one of the candidates, and you
know who I mean, asserts that he has a team and a program,
but when the country was on the verge of civil war in 1993,
Etibar Mammadov was involved in the political scene so why
did he not do anything and why did he not try to stop" the
tragedy.
```

Figure 3: Lead summary used as starting point for greedy search (ROUGE-1 37.8%, cosine 0.0310).

iterations. Some documents require quite many iterations before a local maximum is found, but these constitute a fairly small amount of the texts in the data set.

Example of a lead summary used as starting point for the greedy search can be found in Figure 3. As we can see, the lead summary is just the leading sentences within one document, and as such only covers the aspects of the document chosen to be presented there. Since our method tries to find a summary that is more similar to the view it has of the whole document, it thus transforms the initial summary into a summary with a wider coverage (if no slanting strategies are applied).

The local maximum summary, with a ROUGE-1 score of 44.0% and a 0.995 cosine closeness to the full document, reached from the lead summary given in Figure 3 is presented in Figure 4. Typically you will want as high ROUGE score as possible as this has been shown to correlate with summaries humans perceive as good summaries for a certain text (Hovy and Lin 2002, Lin and Hovy 2003a). The cosine angle between the summary vector and the document vector, both located in the same vector space, indicates the closeness, or likeness, between the current summary and the full document. This varies between -1 and 1 where 1 indicates complete similarity.

3.2.3 Evaluation

For reasons of comparability and the benefit of a human ceiling, we have chosen to mimic the evaluation set-up for task 2 in DUC 2004 (Over and Yen 2004). As in this evaluation campaign we have carried out our evaluation using ROUGEeval (Lin 2003) with the same data and model summaries. While our method itself

```
Supporters of Azerbaijani President Heydar Aliyev proclaimed
today, Monday, that he was reelected for a new term from
the first round that took place yesterday, Sunday, while
his main opponent Etibar Mammadov, declared that a second
round ought to be held.  The 4200 polling offices, under
the supervision of 180 observers from the Security and
Cooperation Organization in Europe, will remain open till
20:00 local time.  In order to win in the first round as
Aliyev hopes, a candidate must win more than 75% of the
votes with a turnout of over 25%.
```

Figure 4: Local maximum summary scoring ROUGE-1 44.0%, with a cosine similarity of 0.995.

is largely language independent, and thus should work comparably well on many other languages given enough raw text, the data prepared for the DUC evaluations is widely used and as such forms a basis for comparison with other systems and methods. The evaluation was carried out by first using all manually created 100 word summaries provided for DUC 2004 as reference summaries, trimming our system with different basic tokenizers and preprocessors (i.e. sentence splitting, stop word filtering and stemming), comparing our results to those reported in (Over and Yen 2004). Having reached a reasonable level of success we then compared against the complete set of man-made 100 word summaries from DUC 2001-2004 in order to verify our method on a larger test set.

The evaluation has been carried out by computing ROUGE scores on the system generated summaries using manual summaries provided for DUC as reference, or model summaries. The ROUGE score is a recall-based n-gram cooccurrence scoring metric that measures content similarity by computing the overlap of word n-grams occurring in both a system generated summary as well as a set of, usually man-made, model summaries. Throughout the evaluations we have, as in DUC 2004, used ROUGEeval-1.4.2 with the following settings:

```
rouge -a -c 95 -b 665 -m -n 4 -w 1.2
```

This means that we use a 95% confidence interval, truncate model and peer at 665 bytes, Porter Stem models and peers and calculate ROUGE-1..4. Also, stop words are not removed when calculating the score. ROUGE scores have in several studies been shown to correlate highly with human evaluation and has high recall and precision in predicting statistical significance of results comparing with its human counterpart (Lin and Hovy 2003b).

	DUC 2004	DUC 2001 - 2004
Human mean	42.6	39.7
Holistic-1000	34.1	32.4
Holistic-500	34.2	32.3
Holistic-250	33.9	32.0
Holistic-RAW	32.7	30.9
Holistic-noRI	30.3	28.5
Baseline-Lead	31.0	28.3

Table 1: ROUGE-1 scores, in %, for different dimensionality choices of the context vectors. RAW indicates no use of stemming and stop word filtering, and noRI uses a traditional $tf \cdot idf$ weighted vector space model instead of Random Indexing.

In our experiments ROUGE scores are in the case of DUC 2004 calculated over 114 system generated summaries, one for each document set, and in the case of DUC 2001-2004 over 291 summaries. A human ceiling (see Table 1) has for reference been calculated by, for each document set, taking the mean of the ROUGE scores for each man-made summary compared to the remaining man-made summaries (i.e. in turn treating each human-written summary as a system summary). On average there are about four man-made summaries available for each set. Also, we evaluate a baseline (lead), which is the initial sentences in each text up to the allowed summary length.

3.2.4 Results

In the evaluations here we have removed stop words and used stemming. Two brief evaluations not using these two strategies showed that both approaches result in considerable improvements, although even without the use of these techniques the system still improves on lead. We also evaluate the impact of the dimensionality chosen for the Random Indexing method by running our experiments for three different values for the dimensionality, building semantic representations using 250, 500 and 1000 dimensions. Our results show little variation over different dimensionalities. This means that as long as we do not choose too few dimensions, the dimensionality is not a parameter that needs considerable attention.

For each dimensionality we also calculated the mean performance using ten different random seeds, since there is a slight variation in how well the method works with different random projections. The dimensionality showing the most variation in our experiments spanned 33.8-34.4% ROUGE-1. Variations for the other dimensions were slightly less. As shown in Table 1, our best run resulted in a mean performance of 34.2%.

A ROUGE-1 score of about 34% on the DUC 2004 data set is not very impressive, but neither is it very bad. The best systems participating in the DUC 2004 evaluation campaign scored roughly 39% (Over and Yen 2004), with many systems scoring around 34% and some below. Concerning scores for ROUGE-2..4 our system unsurprisingly follows the pattern of the results reported in the DUC 2004 evaluation campaign, with considerably lower ROUGE-2 (mean 7.2% with 500 dimensions) and almost non-existing scores for ROUGE-3 (mean 2.3%) and ROUGE-4 (mean 1.0%).

Some naïve attempts at sentence compression by removing "uninteresting" text, such as removing anything mentioned within parenthesis were done. We also tried joining sentences together if the second sentence began with 'but', 'and', 'however', 'although' or similar text binding markers, indicating that the sentences were in some sense dependent. All such experiments, however, degraded the performance.

3.3 Trying Another Language: Swedish

Since the summarization method described above is relatively language independent, we decided to also evaluate it on Swedish. For this purpose we used the KTH Extract Corpus (Hassel and Dalianis 2005), a corpus of human produced extractive summaries of Swedish newspaper articles. These extracts were however not produced to give an overview of the whole contents of the texts, which our method attempts to do. The humans were instead more focused on finding the most important topic in the text and then providing mostly information relevant to that.

There are only 15 relatively short documents in this corpus. On average there are 20 human generated extracts for each document. These vary quite a lot in compression rate, even for a specific document. There are usually some sentences that are included in almost all extracts, though, so there is agreement on what the main topic is. In Figure 5 an example of the variation in selected sentences for one of the texts from the extract corpus is shown. As can be seen in this figure, the HolSum system tries to represent all parts of the text in the same proportion as in the source document. This is here illustrated by the system covering all three "information spikes", as chosen by the human informants.

As reference texts for the Random Indexing method we here used the Swedish Parole corpus (Gellerstam *et al.* 2000), 20 million words, the Stockholm-Umeå Corpus (Ejerhed *et al.* 1992), 1 million words, and the KTH News Corpus (Hassel

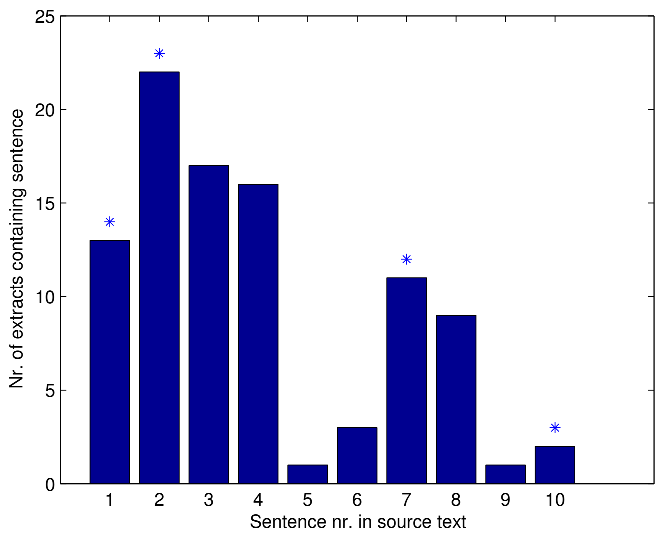

Figure 5: The number of human produced extracts that included each sentence from one of the Swedish corpus texts. There are a total of 27 human produced extracts for this text. This particular text contains 10 sentences, and sentences marked with a * are those selected by our system.

2001), 13 million words. We also used stemming and stop word filtering, since this worked well on the English texts.

3.3.1 Evaluation

When evaluating the Swedish summaries we calculated a weighted precision. The score for a sentence included in the summary is the number of human produced extracts that also included this sentence divided by the total number of human produced extracts for that text. The precision for the summary is then the average for all sentences in the summary.

A recall-like measurement was also calculated, since otherwise it would be best to simply pick a single sentence that the system is sure should be included. Each sentence that was included in at least one human produced extract, but not included in the summary to be evaluated, was also given a score as above, i.e. how often it was included by humans. The recall-like measurement is then the average score for all sentences not included in the summary but included in some human produced

	Included	Ignored	Perfect
Human	53	27	8
Lead, Short	55	29	2
Lead, Long	48	26	2
Random, Short	33	36	0.3
Random, Long	34	37	0
SweSum-above	53	28	3
SweSum-below	54	30	0
Holistic-500, Short	42	34	1
Holistic-500, Long	38	35	0

Table 2: Proportion of human produced extracts that included the sentences chosen by the system, in % (higher is better), and sentences ignored by the system but included by at least one human, also in % (lower is better). "Perfect" indicates for how many of the 15 documents a system generated an extract that was exactly the same as one of the human produced extracts.

extract. Sentences ignored by both the system and the humans have no impact in the evaluation.

Since the extracts vary so much in length we generated two different sets of summaries using our method. The first, called Holistic-long, was the summary most similar to the original text that was longer than the shortest human produced extract and shorter than the longest. This generally produced long summaries, since it is easier to achieve good coverage of the original text with many words than with few. Since long summaries will have lower precision we also generated summaries, called Holistic-short, that, while longer than the shortest human produced extract, were never longer than the average extract.

For both sets of summaries four different Random Indexes generated with four different seeds were used. The results in Table 2 are the mean values of these four sets. All values are within 1.5 percentage units of the mean value. We also compared our system to two baselines: *Lead*, the first sentences of the original text with a size as close to the system generated summary as possible; and *Random*, randomly chosen sentences up to the same size. We also calculated the agreement between the humans, by taking the average over all human produced extracts when treating them one at a time as a system generated summary instead.

Finally, we include figures for another summarization system, SweSum (Dalianis 2000, Hassel 2004), that has also been evaluated on this data set. SweSum uses both statistical and linguistic methods, as well as some heuristics, and its main domain is newspaper text. SweSum creates extracts, by scoring sentences for various criteria, then extracting high scoring sentences in the original text and joining them together. The sentence scores are calculated based on e.g. sentence position, occurrence of numerical data and highly frequent keywords. Two different sets of summaries were generated with SweSum, one with summaries strictly below the average human produced extract length and one with the shortest summary possible above the average length.

3.3.2 Results

As can be seen in Table 2, our system does not generate the same type of summaries as the others. Since our system tries to include the same proportions regarding different topics in the summary as was found in the original text, it has a quite low score with the precision-like measurement. This is natural, since the reference extracts normally only cover one topic. This also leads to a high (i.e. bad) score on the recall-like measurement, since the reference extracts include so much information regarding the main topic that our method discards some of it as redundant.

When generating shorter summaries the same sentences are of course still considered redundant by our method, so the recall-like figure is more or less unchanged. Since the extract is shorter, there is room for less information. This gives higher precision, since our method still agrees that the main topic should be covered, but now includes less information regarding other topics. As expected, it seems like using our method when single topic summaries is wanted does not give the best results. It can also be seen that outperforming the lead baseline on newspaper texts is very hard, since it performs on par with humans when generating shorter extracts. This means that this type of text is not very exciting to do summarization experiments on.

3.4 New Weighting Criteria: Keywords Come in Bursts

When constructing the semantic vector for a text, the context vector for each word is weighted with the importance of this word, by simply making the length of the vector proportional to the importance of the word. The weight could for instance be something simple, such as like in the previous sections making the length of the vector be $tf \cdot \log(idf)$, i.e. the term frequency and inverse document frequency. The term frequency is the frequency of the term within the given document and gives a measure of the importance of the term within that particular document. The inverse document frequency, on the other hand, is a measure of the general importance of the term – i.e. how specific the term is to said document (Salton and Buckley 1987).

In addition to the highly traditional $tf \cdot \log(idf)$ weighting scheme, we have also experimented with utilizing the "burstiness" of a word for term weighting.

	DUC 2004	DUC 2001 – 2004
Human	43	40
Burstyness, 1000	33.9	32.2
Burstyness, 500	33.7	32.1
Burstyness, 250	33.6	31.9
$tf \cdot \log(\mathit{idf})$, 1000	34.1	32.4
$tf \cdot \log(\mathit{idf})$, 500	34.2	32.3
$tf \cdot \log(\mathit{idf})$, 250	33.9	32.0
Baseline-Lead	31.0	28.3

Table 3: ROUGE-1 scores, in %, for burst weighting as well as the standard weighting criteria for reference. There are 114 documents from DUC 2004 and 291 from DUC 2001 – 2004.

Ortuño *et al.* (2002) have shown that the spatial information of a word, i.e. the way in which it is distributed in the text (independently of its relative frequency), is a good measure of the relevance of the word to the current text.

The burstiness of a word is here based on the standard deviation of the distance, in words, between different occurrences of this word in the text. Words that occur only with large distances between occurrences usually have a high standard deviation by chance, so the standard deviation is divided by the mean distance between occurrences. The final weight of a word is thus:

$$tf \cdot \tfrac{\sigma}{\mu}$$

where μ is the mean and σ the standard deviation of the distances between occurrences, in words.

3.4.1 Results

As before, we evaluated on three different dimensionality choices, 250, 500 and 1,000. Generally, as low dimensionality as possible is desirable, since processing times and memory usage is then lower. In Table 3 it can be seen that the variation between different dimensionalities is quite low. It is largest for $tf \cdot \log(\mathit{idf})$, where the mean value for dimensionality 250 is 32.0% and the mean value for 1,000 is 32.3% in the DUC 2001 – 2004 data set. This is nice, since it seems to be unimportant to spend a lot of time optimizing the choice of this parameter.

For each choice of dimensionality the mean performance using ten different random seeds was calculated. The impact of the randomness used in the method

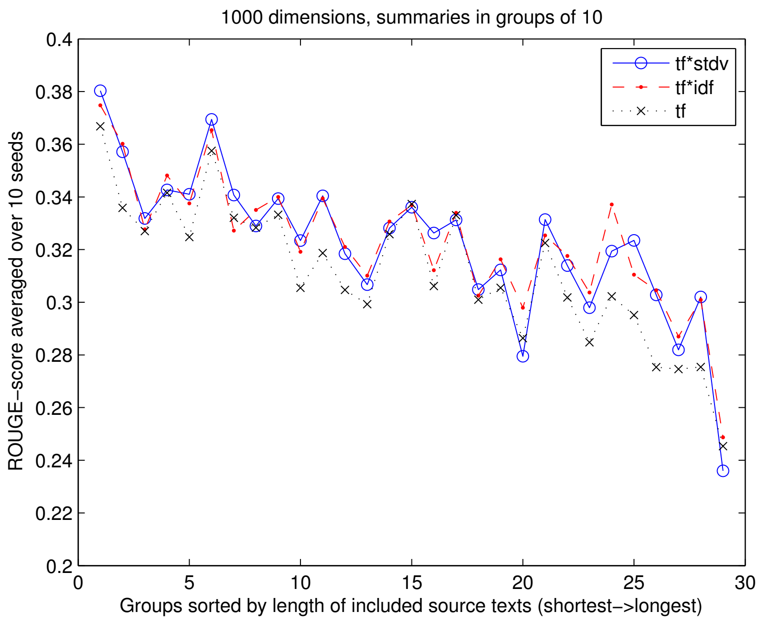

Figure 6: ROUGE-1 scores for three weighting schemes, divided into 29 groups of 10 summaries each sorted by compression rate. The leftmost group contains the summaries for the 10 shortest source texts while the rightmost group contains the summaries for the 10 longest.

seems larger than the impact of the dimensionality choice. The largest variation was for the dimensionality 500, spanning 33.1% – 34.3 % ROUGE-1 score in the DUC 2004 data set. Variations for the other dimensionalities were slightly less.

The choice between $tf \cdot \log(idf)$ or burstyness seems to have very little impact, the results are nearly identical in ROUGE-1 scores. This is further supported when plotting a graph, showing the ROUGE scores for three different weighting schemes. The first weighting scheme is $tf \cdot \log(idf)$, the second is burst weighting and the third is weighting only by the term frequency. In Figure 6 we can see that it is the term frequency that is pulling the most weight and that the inverse document frequency and the standard deviation seem to add roughly the same improvement.

It should, however, not come as such a surprise that the term frequency has the most impact during the accumulation of the context vectors. Since we apply stop word filtering prior to this step, we have already filtered out most of the highly frequent function words. This means that the remaining high frequent words are content words and as such good descriptors of the document being summarized.

In Figure 6 we can also see that summarizer performs best at low compressions rates. This is due to the fact that the more of the source text that is included in the summary, the higher the chance of selecting the same sentences, or choice of words, as the man-made summaries we are using as gold standard.

3.5 A New Search Strategy: Simulated Annealing

One obvious thought is that the greedy hill climbing might be a too simple search strategy and thus miss the best candidates available in the summary space. The best summaries may not lie down the path of always choosing the best neighbor. What if beyond one of the lesser neighbors lies an even better summary?

The method we used for investigating this idea is simulated annealing (Kirkpatrick *et al.* 1983), augmented with back-off heuristics. Instead of in each step choosing the best neighbor as our next transition point we may go to a randomly chosen neighbor, as long as it is better than the current summary. However, in doing this we also keep track of the best neighbor so far, and in the case that we venture to far down a slope[3] we can always go back to the best neighbor previously visited and start our search anew. A ban list containing all visited summaries, excluding the best summary so far, effectively hinders us from going down the same path again (not that it would have mattered much, bar computing time). This means that the annealing procedure will always perform at least on par with the greedy search regarding cosine scores.

With simulated annealing the cooling schedule is of great importance (Laarhoven and Aarts 1987). The cooling schedule is the factor that in each transition governs the probability of choosing a random better neighbor instead of the best neighbor. Two common formulas for calculating the cooling factor were used in these experiments. The first schedule was calculated using the following formula:

$$T_i = T_0 \left(\frac{T_N}{T_0}\right)^{\frac{i}{N}}$$

In this formula T_i is the probability of choosing a random better neighbor in step i, where i increases from 0 to $N = 100$ transitions. The initial probability T_0 is set to 100% and the lowest allowed probability to $T_N = 5\%$. This schedule starts with a high probability for random behavior and then rapidly reverts to a traditional greedy search. The second cooling schedule, using the same notation as above but

[3]In our case ten transitions without finding a new summary that is better than best one seen so far.

	DUC 2004	DUC 2001 – 2004
Human	43	40
Schedule 1, 1000	34.1	32.4
Schedule 1, 500	34.2	32.3
Schedule 1, 250	33.9	32.0
Schedule 2, 1000	34.2	32.4
Schedule 2, 500	34.2	32.3
Schedule 2, 250	34.0	32.0
Holistic-1000	34.1	32.4
Holistic-500	34.2	32.3
Holistic-250	33.9	32.0
Baseline-Lead	31.0	28.3

Table 4: ROUGE-1 scores, in %, for the the two annealing schedules as well as the standard greedy search for reference.

with T_N set to zero, was designed to revert to a greedy search more linearly:

$$T_i = T_0 - i\frac{T_0 - T_N}{N}$$

The algorithm was in both cases set to break when no known neighbors are better than the current summary and no previous state or neighbor has been better, in terms of cosine closeness, or the maximum number of 100 transitions has been reached. At this point the best state, current or previously visited, is returned. In most cases the maximum number of transitions was never reached.

3.5.1 Results

As can be seen in Table 4 the resulting summaries were in almost all cases identical to the summaries generated using the bare greedy search algorithm. In the as few as 7 cases out of 2910 where the summaries generated with a dimensionality of 500 differed, the second cooling schedule resulted in slightly higher ROUGE scores, but not enough to warrant the radically added computation time. For the same dimension the first schedule resulted in only one higher scoring summary.

Of course, a formula with a slower descent into a traditional greedy search could be used. However, this would probably lead to even further increased run times, depending on whether the cooling schedule in fact reaches a local optimum in fewer transitions or not. As it is, simulated annealing, using the two cooling

	DUC 2004	DUC 2001 – 2004
Human	43	40
rand, 1000	33.2	31.1
rand, 500	33.0	31.2
rand, 250	33.1	31.1
randlead, 1000	33.1	31.3
randlead, 500	33.2	31.3
randlead, 250	33.1	31.3
lead, 1000	34.1	32.4
lead, 500	34.2	32.3
lead, 250	33.9	32.0
Baseline-Lead	31.0	28.3

Table 5: ROUGE-1 scores, in %, for the two different random starting point strategies as well as the standard lead starting point for reference.

schedules presented here, in general takes about three times as long to generate the set of summaries evaluated in each run, compared to the standard greedy search.

3.6 Expanding the Search Scope: Different Points of Departure

Considering the approaches above, we have still only investigated a small fraction of the high-dimensional vector space representing all possible summaries. As previously stated it is simply not feasible to exhaustively search all possible summaries in pursuit of the best summary. Another option is to again put the greedy search to use, but this time giving it randomly chosen starting points. The idea here is that there may be better starting points than the leading sentences of the original text, thus taking other paths to possibly better summaries.

We have tried two approaches, where the first simply choses sentences randomly from the source text and concatenates them into an initial summary of desired length. The second, and slightly less naive approach, picks a random sentence in the source text and grabs it and the following couple of sentences to use as the initial summary for that text. After this the algorithm proceeds as before, transforming the initial summary until no better summary is found.

3.6.1 Results

One would like to believe that some difference in the results would show between these two approaches since the first obviously disregards any coherency in the text, while the other at least retains some. The second approach does however potentially breach coherency somewhat in that it may start e.g. in the middle of one paragraph and continue half-way into the next, or, when dealing with a concatenated set of topically related texts, come to span over a document boundary. However, as can be seen in Table 5, the results from both approaches are strikingly similar, giving further support to the notion that leading sentences of a document constitutes a stable starting point.

4 Conclusions

We have presented and evaluated an extraction based summarization method based on comparing whole summaries, not ranking individual extraction segments. It produces extracts that include the same proportions of topics as the original text. The method is largely language independent and requires no sophisticated tools, though stop word filtering and simple stemming was used in our experiments. For good performance, access to large amounts of raw text is needed, but for many languages this is readily available.

In the major part of our experiments we have used the leading sentences of a text as a starting point for our system since this itself usually constitutes a good summary. Though by doing this we limit our search for a better summary to a very limited area of the high-dimensional summary space. Since an exhaustive search of the vector space is not reasonable we have also sampled the space using some randomly chosen starting points, as well as used simulated annealing with the leading sentences as starting point. The results, however, show that using the lead summary as a starting point is a reliable heuristic also in this application.

Due to the fact that our method tries to cover all topics covered in the original text, it did not perform very well when evaluated against man-made extracts produced to cover mostly the main topic of a text. It did however perform well on short extracts derived from fairly long news texts when compared to man-made summaries, such as those used in the DUC 2004 summarization evaluation campaign. On this task the proposed method performs better than several of the systems evaluated there, but worse than the best systems.

Even though the HolSum summarizer does not outperform the best systems for English it is trivial to port to other languages. It also has the intuitively appealing property of optimizing semantic similarity between the generated summary and the text being summarized. Also, this property is not constrained to extractive summarization, even though we here use it to differentiate between extractive summaries. The summaries being evaluated and selected from could in practice be generated by any means, even being man-made.

References

Lou Burnard. 1995. The Users Reference Guide for the British National Corpus.

Hercules Dalianis. 2000. SweSum - A Text Summarizer for Swedish. Technical Report TRITA-NA-P0015, IPLab-174, KTH NADA, Sweden.

Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society of Information Science*, 41(6):391–407.

DUC. 2007. Document Understanding Conferences. http://duc.nist.gov/.

Educational Testing Service. 2006. Test of English as a Foreign Language (TOEFL). http://www.ets.org/toefl.

Eva Ejerhed, Gunnel Källgren, Ola Wennstedt, and Magnus Åström. 1992. *SUC - The Stockholm-Umeå Corpus*, version 1.0 (suc 1.0). CD-ROM produced by the Dept of Linguistics, University of Stockholm and the Dept of Linguistics, University of Umeå. ISBN 91-7191-348-3.

Martin Gellerstam, Yvonne Cederholm, and Torgny Rasmark. 2000. The bank of Swedish. In *In the proceedings of Second International Conference on Language Resources and Evaluation. LREC-2000*, pages 329–333, Athens, Greece.

Zelig S Harris. 1968. *Mathematical Structures of Language*. New York: Wiley.

Martin Hassel. 2001. Internet as Corpus - Automatic Construction of a Swedish News Corpus. In *Proceedings of NODALIDA'01 - 13th Nordic Conference on Computational Linguistics*, Uppsala, Sweden, May 21-22 2001.

Martin Hassel. 2004. *Evaluation of Automatic Text Summarization - A practical implementation*. Licentiate thesis, Department of Numerical Analysis and Computer Science, Royal Institute of Technology, Stockholm, Sweden.

Martin Hassel. 2006. JavaSDM - A Java tool-kit for working with Random Indexing. http://www.nada.kth.se/∼xmartin/java/JavaSDM/.

Martin Hassel and Hercules Dalianis. 2005. Generation of Reference Summaries. In *Proceedings of 2nd Language & Technology Conference: Human Language Technologies as a Challenge for Computer Science and Linguistics*, Poznan, Poland, April 21-23 2005.

Eduard Hovy and Chin-Yew Lin. 2002. Manual and Automatic Evaluation of Summaries. In Udo Hahn and Donna Harman, editors, *Proceedings of the Workshop on Text Summarization at the 4Oth Meeting of the Association for Computational Linguistics*.

Scott Kirkpatrick, C. Daniel Gelatt, and M. P. Vecchi. 1983. Optimization by simulated annealing. *Science, Number 4598, 13 May 1983*, 220, 4598:671–680.

Peter J. M. Laarhoven and Emile H. L. Aarts, editors. 1987. *Simulated annealing: theory and applications*. Kluwer Academic Publishers, Norwell, MA, USA. ISBN 9-027-72513-6.

Thomas K. Landauer, Peter W. Foltz, and Darrell Laham. 1998. Introduction to Latent Semantic Analysis. *Discourse Processes*, 25:259–284.

Chin-Yew Lin. 2003. ROUGE: Recall-oriented understudy for gisting evaluation. http://www.isi.edu/∼cyl/ROUGE/.

Chin-Yew Lin and Eduard Hovy. 2003a. Automatic Evaluation of Summaries Using n-gram Co-occurrence Statistics. In *Proceedings of 2003 Language Technology Conference (HLT-NAACL 2003)*, Edmonton, Canada, May 27 - June 1 2003.

Chin-Yew Lin and Eduard Hovy. 2003b. The potential and limitations of automatic sentence extraction for summarization. In Dragomir Radev and Simone Teufel, editors, *HLT-NAACL 2003 Workshop: Text Summarization (DUC03)*, Edmonton, Alberta, Canada, May 31 - June 1 2003. Association for Computational Linguistics.

Kevin Lund, Curt Burgess, and Ruth Ann Atchley. 1995. Semantic and associative priming in high-dimensional semantic space. In *Proceedings of the Cognitive Science Society*, pages 660–665, Hillsdale, N.J.: Erlbaum Publishers.

M. Ortuño, P. Carpena, P. Bernaola-Galvan, E. Munoz, and A. Somoza. 2002. Keyword detection in natural languages and DNA. *Europhysics Letters*, 57: 759–764.

Paul Over and James Yen. 2004. An Introduction to DUC 2004 Intrinsic Evaluation of Generic New Text Summarization Systems.
http://www-nlpir.nist.gov/projects/duc/pubs/2004slides/duc2004.intro.pdf.

Magnus Sahlgren. 2005. An Introduction to Random Indexing. In *Methods and Applications of Semantic Indexing Workshop at the 7th International Conference* *on Terminology and Knowledge Engineering, TKE 2005)*, Copenhagen, Denmark, August 16 2005.

Magnus Sahlgren. 2006. *The Word-Space Model: Using distributional analysis to represent syntagmatic and paradigmatic relations between words in high-dimensional vector spaces*. Doctoral thesis, Department of Linguistics, Stockholm University, Stockholm, Sweden.

Magnus Sahlgren and Rickard Cöster. 2004. Using Bag-of-Concepts to Improve the Performance of Support Vector Machines in Text Categorization. In *Proceedings of the 20th International Conference on Computational Linguistics, COLING 2004*, Geneva, Switzerland, August 23-27 2004.

Gerard Salton and Chris Buckley. 1987. Term weighting approaches in automatic text retrieval. Technical report, Ithaca, NY, USA.

Klaus Zechner. 1996. Fast generation of abstracts from general domain text corpora by extracting relevant sentences. In *The 16th International Conference on Computational Linguistics, COLING 1996*, pages 986–989, Center for Sprogteknologi, Copenhagen, Denmark, August 5-9 1996.