Extraction based summarization using a shortest path algorithm

Jonas Sjöbergh [1,2]

[1] KTH Nada
Sweden
jsh@nada.kth.se

Kenji Araki [2]

[2] Graduate School of Information Science and Technology
Hokkaido University
Japan
araki@media.eng.hokudai.ac.jp

Abstract

We present an extraction based method for automatic summarization. It is based on finding the shortest path from the first sentence to the last sentence in a graph representing the original text. Nodes represent sentences and edges represent similarity between sentences. Simple word overlap is used for similarity. Traditional sentence weights are also used, making edges to important sentences cheaper. Evaluation using ROUGE scores on DUC texts give scores outperforming human interagreement on 200 words and 400 words extracts, while performance on 100 words is less impressive.

1 Introduction

Automatic text summarization has been an active research area for a long time (Luhn, 1958; Edmundson, 1969; Salton, 1988). Currently, most systems use text extraction, i.e. parts of the original text that are deemed interesting are extracted and presented as a summary. Here, a graph based text extraction method is presented. Other graph algorithms have successfully been used for summarization (Mihalcea, 2004). Our method uses a shortest path algorithm in a graph where sentences are nodes and edges represent word overlap between sentences.

Extraction based summarization normally produces summaries that are somewhat unappealing to read. There is a lack of flow in the text, since the extracted parts, usually sentences, are taken from different parts of the original text. This can for instance lead to very sudden topic shifts. The idea behind the presented method of extracting sentences that form a path where each sentence is similar to the previous one is that the resulting summaries hopefully have better flow. This quality is however quite hard to evaluate. Since the summaries are still extracts, high quality summaries should still not be expected.

2 Building the graph

When a text is to be summarized, it is first split into sentences and words. The sentences become the nodes of the graph. Sentences that are similar to each other have an edge between them. Here, similarity simply means word overlap, though other measures could also be used. Thus, if two sentences have at least one word in common, there will be an edge between them. Of course, many words are ambiguous, and having a matching word does not guarantee any kind of similarity. Since all sentences come from the same document, and words tend to be less ambiguous in a single text, this problem is somewhat mitigated.

All sentences also have an edge to the following sentence. There are two reasons for this. The most important is that the method will only work if there is a path from the first sentence to the last, which will be guaranteed by this step. The second is that since these two sentences were put next to each other in the original text, it would still be a smooth text if they are next to each other in the summary too.

Edges are given costs (or weights). The more similar two sentences are, the less the cost of the edge.

The further apart the sentences are in the original text, the higher the cost of the edge. To favor inclusion of "interesting" sentences, all sentences that are deemed relevant to the document according to classical summarization methods have the costs of all the edges leading to them lowered.

The cost of an edge from the node representing sentence number i in the text, S_i, to the node for S_j is calculated as:

$$cost_{i,j} = \frac{(i-j)^2}{overlap_{i,j} \cdot weight_j}$$

and the weight of a sentence is calculated as:

$$\begin{aligned} weight_j = (1 + overlap_{title,j}) \\ \cdot \left(\frac{1 + \sum_{w \in S_j} tf(w)}{\sum_{w \in text} tf(w)} \right) \\ \cdot early(j) \cdot \sqrt{1 + |edges_j|} \end{aligned}$$

where $early(j)$ is 2 if $j < 10$ and 1 otherwise, $overlap_{i,j}$ is simply the number of words in common between sentences S_i and S_j, and only words of four or more characters are counted in the tf (term frequency in the document) calculations.

Since similarity is based on the number of words in common between two sentences, long sentences have a greater chance of being similar to other sentences. Favoring long sentences is often good from a smoothness perspective. Summaries with many short sentences have a larger chance for abrupt changes, since there are more sentence breaks. The contents of a single sentence are normally smooth, so the main problem is when changing from one sentence to the next.

3 Constructing the summary

When the graph has been constructed, the summary is created by taking the shortest path that starts with the first sentence of the original text and ends with the last sentence. Since the original text also starts and ends in these positions, this will hopefully give a smooth but shorter set of sentences between these two points.

The N shortest paths are found by simply starting at the start node and adding all paths of length one to a priority queue, where the priority value is the total cost of a path. The currently cheapest path is then examined and if it does not end at the end node, all paths starting with this path and containing one more edge are also added to the priority queue. Paths with loops are discarded. Whenever the currently shortest path ends in the end node, another shortest path has been found, and the search is continued until the N shortest paths have been found.

This gives wildly varying lengths (in the number of words) of the summaries for different texts. This is often not desirable. Usually, the summarization task has a predetermined expected length of the summary. To allow for this, the N shortest paths are generated and the one closest to the desired length is chosen.

If none of the summaries are of an appropriate length, two heuristics are used. If the summaries are too long, the shortest one is selected and simply cut of at the desired length. If the summaries are too short, the longest is selected and padded to the desired length by adding previously unselected sentences, starting with the sentence with the highest importance weight, to the end of the summary until the desired length is reached. These heuristics are not very good when it comes to producing smooth summaries, though.

4 Evaluating the summaries

While the original idea for the method was based on the hope for smoother extracts, this quality is hard to evaluate. For a summary to be useful it is not enough to be easy to read, the contents are also very important. This is much easier to evaluate. This first evaluation of the shortest path method thus concentrates on evaluating if the produced extracts contain the important information.

The shortest path summarizer was evaluated on test texts of various lengths from the Document Understanding Conference, DUC (DUC, 2005). Human written summaries of lengths of 100 words, 200 words and 400 words are available in these data sets. The texts to be summarized are about 7,000 words long and mostly consist of newspaper texts.

The automatic evaluation method ROUGE was used for evaluating how well the extracts correspond to the manually written summaries. ROUGE more or less measures word overlap between texts in different ways, and ROUGE scores have been shown to

	100 words, 2004	100 words, 2001 – 2004	200 words	400 words
Shortest path	35 / 31 / 11	33 / 29 / 10	41 / 38 / 12	54 / 49 / 15
Lead	31 / 27 / 10	28 / 25 / 9	38 / 35 / 11	51 / 46 / 14
Agreement	43 / 38 / 13	40 / 36 / 13	40 / 37 / 12	41 / 37 / 11

Table 1: The shortest path method, the lead baseline and human interagreement, ROUGE-1 / ROUGE-L / ROUGE-W scores for texts from the DUC data sets from 2001 – 2004. There are 291 documents of 100 words in this set, 114 of which are from the year 2004. There are 87 documents with 200 words summaries and only 28 with 400 words summaries.

correlate well with human evaluation (Hovy and Lin, 2002; Lin and Hovy, 2003a; Lin and Hovy, 2003b). The results are shown in table 1.

Three ROUGE metrics are reported in the table. ROUGE-1 measures word overlap. ROUGE-L measures the longest common word sequence. ROUGE-W is also based on the idea of long common word sequences, but weighted to favor sequences where consecutive words from the respective documents are used. There are also word n-gram based ROUGE metrics, but the scores for the best systems (and human interagreement) are very low, so differences between systems are not very clear using these metrics. The inter system ranking is usually the same regardless of which ROUGE metric is used, though.

The system is compared to the baseline called lead, simply taking the desired number of words from the start of the original text. The system is also compared to the interagreement between humans. This was done by simply evaluating each of the human written summaries as if it was produced by an automatic system, comparing it to the remaining human written summaries. The reported figure is the mean value for all such summaries.

While the system does not perform badly on 100 words, nor does it perform very well. These summaries are too short for the system, which rarely finds a short path that contains only 100 words. Thus the aggressive cutting heuristic is normally used. On the 100 words texts from DUC 2004, the best systems had a ROUGE-1 score of about 39%, using the same evaluation method and data sets. Many systems performed similarly to the shortest path method, with around 34% ROUGE-1 scores.

On longer texts, ROUGE scores are as far as is known to us not generally available for other systems. The shortest path system does however outperform the lead baseline, which is usually a quite good summarizer on newspaper texts. It even outperforms human interagreement, which should be considered quite good. Of course, for summaries of 400 words, the baseline also outperforms human interagreement, so this may not mean so much.

5 Discussion

The system is quite simple, using no language resources other than word tokenization and sentence splitting. It is easy to implement and should be relatively language independent, though it was only evaluated on English texts. For longer documents the processing time for the shortest path algorithm can be quite long, although the current implementation is not at all optimized for speed.

Some possible problems with the method is that it could quite possibly keep a lot of the redundancy in the original text by selecting sentences that are too similar. It could also miss the main point of the text completely, though the weighting of "important sentences" helps in avoiding this. In practice it seems to work quite well.

When looking at the generated summaries, they are often somewhat "smooth" to read. This smoothness is hard to quantify objectively, though, and the extracts are by no means as smooth as a manually written summary.

When it comes to including the important facts from the original text, the weighting of sentences using traditional extraction weighting methods seems to be the most important part. Taking a path from the first to the last sentence does give a spread to the summary, making it more likely that most parts of the original text that are important will be included and making it unlikely that too much information is included from only one part of the original text.

Acknowledgments

We thank Viggo Kann for contributing useful ideas and helpful suggestions and Martin Hassel for helping out with the evaluations.

This work has been funded by The Swedish Research Council (VR).

References

DUC. 2005. Document understanding conferences. http://duc.nist.gov/.

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

Eduard Hovy and Chin-Yew Lin. 2002. Manual and Automatic Evaluation of Summaries. In Udo Hahn and Donna Harman, editors, *ACL02-WS*, July 11–12.

Chin-Yew Lin and Eduard Hovy. 2003a. Automatic Evaluation of Summaries Using N-gram Co-occurrence Statistics. In Udo Hahn and Donna Harman, editors, *Proceedings of the 2003 Human Language Technology Conference (HLT-NAACL 2003)*, Edmonton, Canada, May 27–June 1.

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. Association for Computational Linguistics.

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

Rada Mihalcea. 2004. Graph-based ranking algorithms for sentence extraction, applied to text summarization. In *The Companion Volume to the Proceedings of 42st Annual Meeting of the Association for Computational Linguistics*, pages 170–173, Barcelona, Spain.

Gerard Salton. 1988. *Automatic Text Processing*. Addison-Wesley Publishing Company.