CIS 530 Final Project: Automatic Summarization Systems

Jingyi Wu
wujingyi@seas.upenn.edu

Emily Boggs
emboggs@seas.upenn.edu

Abstract

For this project, we have implemented three basic summarizers: a TF*IDF summarizer, a LexRank summarizer, and a KL-divergence summarizer. In addition, we have designed a new summarizer that ranks sentences using a feature-based classifier.

1 General Parameters

Some system parameters are consistent for all of the summarizers.

- Each summarizer takes two parameters: an input collection with groups of documents to summarize, and the output folder in which to save the summaries.
- The summarizer output is named by prefixing the name of the current subdirectory with ”sum_”. For instance, the input collection subdirectory ”dev_00” would produce the summary ”sum_dev_00.txt”.
- The sentences of the input collections are always lowercased.
- Only sentences that are between 9 and 45 words long are considered for inclusion in the summary.
- To reduce redundancy, sentences that exceed a certain threshold of similarity with any sentence already in the summary are not added. The exact value of the threshold differs between implementations.
- For TF*IDF calculations, IDF was computed from the New York Times corpus. Because this corpus does not contain every word in the test inputs, IDF calculations used add-one smoothing.

2 Basic Systems

2.1 TF*IDF System

The TF*IDF Summarizer ranks sentences by calculating the average TF*IDF score for the words in each sentence. In this implementation, stopwords are included in the calculation, because the proportion of stopwords to content-ful words is an important aspect of the TF*IDF ranking. The sentence score is calculated over all word types, instead of all tokens. In tests with the development data, using types instead of tokens was found to result in a small increase in ROUGE-2 recall. In reducing redundancy, the best similarity threshold was found to be 0.8, which resulted in the highest ROUGE-2 recall for the development data.

2.2 LexRank System

DESCRIBE LEXRANK HERE

2.3 Kl Divergence System

The KL Summarizer selects sentences to add to the summary by greedily choosing the sentence that will minimize KL divergence between the summary and the input. For this implementation, stopwords are ignored in all calculations. Because unigram distribution of stopwords is less likely to be substantially different between the summary and the input, removing them would highlight the meaningful words in the distribution. No smoothing was performed in calculating the distributions; since KL is a sum of probability calculations, a zero term does not significantly affect the output. The threshold for removing redundant sentences is set to 1, which means that sentences are not ignored on the basis of similarity to sentences already in the summary. This value was chosen based on tests with development data; one possible explanation for the result is that the KL computation takes care of redundancy issues itself, causing any further effort to be counterproductive.

2.4 Performance on Development Set

System	TF*IDF	LexRank	KL
Rouge-2 Recall	*.**	*.**	*.****

3 Your Summarization System

3.1 System Design

This part shows general idea of your system. You may use flowchart, graphics or pseudo-code to describe your algorithm.

3.2 Resources & Tools Used

SRILM is used to create a language model of the dev_model summaries. Sentences are ranked by their perplxity in the language model, to automatically generate labels for training the classifier. Sentences with lower perplexity are assumed to be more likely to be in the summary.

WordNet is used in one of the classifier feature groups. The specificity of each word in a sentence is computed as the distance between the word and the root hypernym in WordNet. Each word is designated as specific, general, or medium according to pre-determined thresholds. The counts of each of these word types, as well as the average specificity of the words in the sentence, are used as feature values. The motivation for this feature is the idea that summaries might be more likely to contain general terms instead of specific ones, or vice versa.

Stanford NER is also used in a feature group, by counting the number of each type of named entity (person, organization, and location) in a sentence. The motivation for this feature is the intuition that named entities provide important detail about the topic being discussed (such as the ”who, what, where” of an article).

TopicS is used for another feature pertaining to the topic words in set of documents. Sentences contain topic words are likely to be discussing the general topic of an article, as opposed to a specific detail; these sentences should be more highly ranked.

MPQA/GenInq?? is used for a feature based on sentiment analysis. By characterizing the sentiment content of a sentence, it might be possible to gain an idea of its likelihood of occurring in a summary. For example, summaries might contain more neutral language, leaving the stronger sentiments to the more detailed descriptions in a full article. On the other hand, the sentences with strong positive and negative sentiment words may be considered more important to include in a summary.

3.3 Performance

4 Discussion and Analysis

References

Alfred V. Aho and Jeffrey D. Ullman. 1972. *The Theory of Parsing, Translation and Compiling*, volume 1. Prentice-Hall, Englewood Cliffs, NJ.

American Psychological Association. 1983. *Publications Manual*. American Psychological Association, Washington, DC.

Association for Computing Machinery. 1983. *Computing Reviews*, 24(11):503–512.

Ashok K. Chandra, Dexter C. Kozen, and Larry J. Stockmeyer. 1981. Alternation. *Journal of the Association for Computing Machinery*, 28(1):114–133.

Dan Gusfield. 1997. *Algorithms on Strings, Trees and Sequences*. Cambridge University Press, Cambridge, UK.