# Summarizing a Blog Search Engine Hits

Michel Généreux
Laboratoire d'Informatique de Paris-Nord
CNRS UMR 7030, Université Paris 13
93430 Villetaneuse — France
genereux@lipn.fr

## ABSTRACT

This paper presents a strategy to subjective document summarization based on a set of relevant features. After showing that the approach can be used to build a competitive summarizer for opinionated texts, we go a step further and explore the limits of the strategy to help users in their search for information in relevant blog posts.

## Categories and Subject Descriptors

H.5.2 [**Information Systems**]: Information Interfaces and Presentation—*User Interfaces, Natural language*

## General Terms

Experimentation, Languages

## Keywords

Summarization, Search, Evaluation

## 1. INTRODUCTION

With the advent of the blogosphere, there is a growing need for systems that would help navigating through a large amount of subjective texts, and summarization is one key research area to lean on. The aim of this paper is twofold: investigate a new approach to summarization to cater for opinionated texts such as blogs and evaluate its use to summarize hits from a blog search engine. Therefore, the first part of this paper gives a technical description and the result of a system we built for the Opinion Summarization task of the Text Analysis Conferences (TAC) 2008 where we obtained very competitive results. The second part of the paper describes the use of a simplified version of the summarizer to produce summaries of relevant documents (*hits*) found by a typical blog search engine and compare them to what is usually provided by the engine as summaries (*fragments*).

## 2. SUMMARIZING BLOG POSTS

We propose here an approach which combines traditional summarization with sentiment detection. Our system's overall architecture is not without resemblance to MEAD [4], a technique that is centred on the computation of cluster *centroïds* to evaluate a sentence relevance to a set of clustered documents. Although our approach relies only incidentally on *centroïds*, our system has preserved a few other recognizable aspects of MEAD, such as a certain number of feature computations and re-ranking. Our system departs mainly from MEAD in a number of ways: the inclusion of sentiment analysis to account for opinionated texts, computing of a score for additional relevant features and post-processing (merging) sentences.

### 2.1 Producing summaries

There are three elements which serve as input to the summarizer: the XML *targets/queries/documents* file associates each *target:query* to a set of relevant blog posts, the posts themselves, and a few optional snippets of answers to each queries drawn from the blog posts and provided by TAC. The output is a summary of the documents (blog posts) corresponding to a target and queries. Producing a summary involves:

**Cleaning** Each XML formatted blog post goes through boilerplate stripping as well as a basic screening to discard noisy input.

**Computing Features** Once we have clean data, we are in a position to compute a number of relevant features for each sentence. Apart from the length of a sentence, there are eight distinct features, presented in section 2.2.

**Clustering** Each cluster gathers blog posts associated with a unique *target:query* combination as provided with the data.

**Query Classification** The clusters were further grouped in two categories according to the sentiment expressed or sought after in the query. Two broad categories were considered: positive and negative. Therefore, a two-class SVM [1] classifier was developed and trained on the manually tagged queries from the training data provided earlier in TAC. The idea behind the grouping of clusters is to improve summaries by selecting sentences having the same opinionated polarity as the query.

**Summarizing** The selection of relevant sentences for summarization is based on a weighted sum of feature values. To avoid including very short sentences in the


*WWW2009*, April 20-24, 2009, Madrid, Spain.
.

summary, sentences of length (in words) below a predefined threshold were given a score of zero. Otherwise, the scoring formula for $n$ features is as follows:

$$\sum_{i=0}^{n} w_i * f_i \quad (1)$$

where $w_i$ represents the weight of feature $f_i$. For TAC, weights have been determined experimentally and reflect how much they impact positively on the results. They are shown in table 1.

| Feature | Pos. Query | Neg. Query |
|---|---|---|
| sim...Snippets | +100 | +100 |
| sim...Query | +40 | +40 |
| sim...Target | +20 | +20 |
| sentiment | +20 | -20 |
| sim...FirstSent. | +10 | +10 |
| centroïd | +10 | +10 |
| position | +10 | +10 |
| isLongestSent. | -10 | -10 |

**Table 1: Feature weights**

We can see that an important weight was attributed to sentences with a high degree of similarity with the snippets. Long sentences were marginally penalized. A negative *sentiment* weight for negative queries can be explained by the fact that the value of a negative sentiment feature was -1.

**Re-ranking** This phase is meant to discard (all but one) too similar sentences from the final summary.

**Merging** At this point we have a number of distinct *target:query* clusters which have been summarized. The final task is to merge clusters with the same *target*. This process involves ranking sentences according to their original position in the summary to eliminate redundant (that is too similar) sentences in the same fashion as we did in re-ranking.

## 2.2 Feature computation

This section presents the computation methodology for each of the eight sentence features.

**sentiment** We have attempted to give each sentence a meaningful positive or negative polarity by using a supervised approach where a set of labelled documents were used for training two SVM classifiers, one for categorising queries and one for categorising sentences from blog posts. Queries had rather regular structures with a few repetitive patterns that could be learned rather easily using only a few training queries. The queries provided in an early TAC release of a sample queries served this purpose. The learned classifier was then used to classify each query of the test data. The classification of sentences from the blog posts presented a rather more challenging task, given the variable length, structures and domain targeted. All these obstacles were arguments to adopt the following safer, although not necessarily more accurate, strategy: a document classifier was built to classify each blog post, in which each sentence was attributed the same polarity as the blog post itself. The document classifier was trained on the self-annotated movie reviews corpus [3].

**isLongestSentence** The longest sentence for each post is attributed a value of 1, 0 otherwise.

**similarityWithTarget** The cosine similarity of a sentence and the target is attributed a value between 0 and 1.

**similarityWithQuery** The cosine similarity of a sentence and the query is attributed a value between 0 and 1.

**similarityWithFirstSentence** The cosine similarity of a sentence and the first sentence of a post is attributed a value between 0 and 1.

**similarityWithSnippets** A list of snippets were provided for each target. A snippet was a piece of information responding to the information need of queries. Each sentence was attributed a value of similarity between 0 and 1 corresponding to the similarity with the most similar snippet.

**centroïd** The *centroïd* value measures how significant a sentence is with regards to other sentences in a text, based on the TF*IDF technique. The *centroïd* of sentence $s$ is calculated as follows:

$$centroid_s = \sum_{word} tf_{word,s} * idf_{word,s} \quad (2)$$

The final *centroïd* values for a sentence is normalized to a value between 0 and 1 by dividing each *centroïd* by the highest *centroïd* in the text.

**position** This value reflects how close to the top (sno = 1) of the text sentence $s$ is located.

$$position_s = \sqrt{1/sno_s} \quad (3)$$

## 2.3 Results

As can be seen in table 2, our run obtained more than competitive results.

| Pyramid | Grammaticality |
|---|---|
| 0.393 F-measure | 6.636 score |
| (best: 0.534, worst: 0.101) | (best: 7.545, worst: 3.545) |
| **Non-redundancy** | **Structure/Coherence** |
| 6.818 score | 3.045 score |
| (best: 8.045, worst: 4.364) | (best: 3.591, worst: 2.000) |
| **Fluency/Readability** | **Responsiveness** |
| 4.591 score | 4.500 score |
| (best: 5.318, worst: 2.636) | (best: 5.773, worst: 1.682) |

**Table 2: Evaluation results**

If we group the results by *content*, *fluency/readability* and *overall responsiveness*[1], then we have the following rankings:

[1]We group the different result categories as suggested in the README file provided by NIST with the results: 1. Content = Pyramid; 2. Fluency/Readability = Grammaticality, Non-redundancy, Structure/Coherence and Fluency/Readability; and 3. Overall responsiveness = Responsiveness

**Content** F-measure = 0.40
(best:0.53 worst:0.10) is ranked fifth behind one manual run and three automatic runs

**Fluency/readability** score = 4.22
(best:4.87 worst:2.73) is ranked fourth behind one manual run and two automatic runs

**Overall responsiveness** score = 4.50
(best: 5.32 worst: 2.64) is ranked eighth behind one manual run and six automatic runs

If we give each of the six categories the same weight, then our system is ranked first with an average score of 0.49 (worst:0.29). There were 36 runs submitted by all participants.

# 3. SUMMARIZING HITS

In this section we apply automatic summarization to the hits returned by a blog search engine. Hits from a typical engine would be roughly summarized by extracting a few sentences that hold terms from the query (minus stopwords). Some of the selected sentences might be truncated in order to keep the length of the resulting summary (we call it *fragment*) within a reasonable size. For example, the following fragments are actual hits returned by a blog search engine for the query *obama OR clinton OR bush*:

**Hit 1** In an Ideas piece, authors say managing relations with lawmakers is one of **Obama's** most important roles. ... Mikes Playbook. Secretary **Clinton** urges China to continue buying U.S. debt. ...

**Hit 2** How **Bush's** Policies in the Muslim World Played into Terrorists' Hands - Can **Obama** Reverse Course? - The Huffington Post.

**Hit 3** Why is the media so desperate to create a divide between **Obama** and **Clinton**? The NYT in particular; I think they are still butt hurt about **Obama** not giving them the first post-election interview. No biggie, they'll be gone soon. ...

We are interested in comparing such search engine fragments with genuine short summaries built using the method previously described. In this comparison the fragments represent the baseline. Given a query, how much is worth the effort of producing a summary from each hit instead of a fragment highlighting (a few) terms in context from the query? Ideally, this question would be answered on the basis of experiments with real users by asking them to complete an information search task [2]: can users prune irrelevant information and navigate more efficiently through web pages when presented with summaries that combine local (contextual) and global cues? This is still an open question, but this paper makes the hypothesis that better summaries would serve as good proxies to more efficient searches. It is a matter for system designers to decide whether this added value is worth implementing. Our experiment aims at giving an evaluation of the worthiness from summaries by applying the following methodology:

1. We select three groups of queries/blogs/nuggets from TAC 2008. Nuggets were described in TAC as synthesizing all the information relevant to all query terms in all blog posts. Group 1009 consists of 29 blog posts (averaging 51 sentences per post and 19 words per sentence), 40 nuggets and the query words *architecture*, *Frank*, *Gehry*, *structure*, *complaint* and *Frank*, to answer the query *What complains or complaints are made concerning his structures?* about the *architecture of Frank Gehry* (the target). Group 1019 consists of 11 blog posts (averaging 66 sentences per post and 17 words per sentence), 26 nuggets and the query words *China*, *one-child*, *family*, *law*, *complaints* and *approval*, to answer the query *What complaints or reason for approval are made about China's one-child per family law?* about *China's one-child per family law* (the target). Group 1033 consists of 17 blog posts (averaging 181 sentences per post and 18 words per sentence), 47 nuggets and the query words *World*, *Bank*, *Wolfovitz*, *scriticized* and *praised*, to answer the query *What actions by Wolfovitz as president of the World Bank are criticized or praised?* about the *World Bank* (the target). All in all, blog posts average 93 sentences (1696 words) per post and 19 words per sentence. An excerpt from the list of nuggets for group 1009 is presented in table 3.

2. For each group, we built one fragment per blog post by concatenating full sentences (from a blog) that contain at least one query term until all query terms have been exhausted or we reach the end of the blog. In this manner we will create fragments that shows at least as good recall as fragments from actual search engines. Evaluation results for fragments are presented in table 5.

3. For each group, we built one summary per blog post using our own summarizer. We constrain our summarizer in order to pick sentences at least five words long having the best combine weigthed score of *centroïd* and *simWithQuery* (two key features of the system) with a given compression rate. Evaluation results are presented in tables 6 and 7.

4. We take as gold standard summary for each blog post the concatenation of all sentences from that post that contain nuggets for that group. The resulting gold standard summary for each post is compared with each fragment and automatic summary using ROUGE. Properties of the gold standard summaries are shown in table 4.

| |
|---|
| Overly dramatic |
| Titanium is too shiny |
| Office interior is banal |
| Buildings are fortress architecture |
| Sophisticated |

**Table 3: Five nuggets for group 1009**

## 3.1 Results

Let us first look at the gold standard summaries from table 4. Recall that such a summary is built using a sentence extraction method: a blog post is summarized by extracting from that post only sentences holding relevant semantic information (nuggets, as provided by TAC) related to a particular target and query pair used as search keywords. In

relation to the original blog posts, gold standard summaries displays a (sentence) compression rate of 11% (10/93) and a (word) compression rate of 18% (297/1696). The summaries also exhibit longer sentences (29 to 19 words per sentence). Second, we examine the (simulated) fragments as produced

| Group ⟶ | 1009 | 1019 | 1033 | Average |
|---|---|---|---|---|
| Nb. Sent./Sum. | 13 | 5 | 9 | 10 |
| Nb. Words/Sent. | 329 | 27 | 32 | 29 |
| Nb. Words/Sum. | 361 | 131 | 296 | 297 |

**Table 4: Gold standard summaries**

by a typical blog post search engine in table 5. The fragments are two sentences (61 words) long on average and present a recall of 0.24, a precision of 0.48 and a F-score of 0.27. Selected sentences have roughly the same length as the original blog posts (30 to 29 words).

| Group ⟶ | 1009 | 1019 | 1033 | Average |
|---|---|---|---|---|
| Nb. Sent./Snip. | 2.10 | 2.36 | 1.65 | 2 |
| Nb. Words/Sent. | 32 | 29 | 28 | 30 |
| Nb. Words/Snip. | 68 | 70 | 46 | 61 |
| Recall | 0.19 | 0.49 | 0.16 | 0.24 |
| Precision | 0.39 | 0.57 | 0.59 | 0.48 |
| F-score | 0.21 | 0.48 | 0.23 | 0.27 |

**Table 5: Search engine fragments**

We are now in a position to make a comparative evaluation of automatic summaries versus the fragments. The first type of summary we turn our attention to in table 6 is produced by keeping the two highest weighted scored sentences from the post with the following weights: *centroïd* 0.5 and *simWithQuery* 0.5. This weighting scheme will therefore select, on average, one sentence on the basis of its similarity with the query terms and one sentence because it represents a good summary of the post as a whole. This approach yields longer summaries (94 to 61) and sentences (47 to 30) than the fragments, but also better performance (0.32 to 0.27). Although its better recall (0.31 to 0.24) can be partly attributed to longer summaries, it still maintains a slightly better precision (0.51 to 0.48). The second type of summary in table 7 is produced by keeping the three highest weighted scored sentences from the post with the following weights: *centroïd* 0.33 and *simWithQuery* 0.67. This weighting scheme will therefore select, on average, two sentences on the basis of their similarity with the query terms and one sentence because it represents a good summary of the post as a whole. This approach also yields longer summaries (135 to 61) and sentences (45 to 30) than the fragments, but also significantly better performance (0.37 to 0.27). As previously, its better recall (0.36 to 0.24) can be partly attributed to longer summaries, yet it still remains slightly more precise (0.50 to 0.48).

| Group ⟶ | 1009 | 1019 | 1033 | Average |
|---|---|---|---|---|
| Nb. Sent./Sum. | 2 | 2 | 2 | 2 |
| Nb. Words/Sent. | 45 | 40 | 54 | 47 |
| Nb. Words/Sum. | 91 | 79 | 109 | 94 |
| Recall | 0.31 | 0.27 | 0.33 | 0.31 |
| Precision | 0.52 | 0.23 | 0.67 | 0.51 |
| F-score | 0.30 | 0.23 | 0.41 | 0.32 |

**Table 6: Automatic summaries with two sentences**

| Group ⟶ | 1009 | 1019 | 1033 | Average |
|---|---|---|---|---|
| Nb. Sent./Sum. | 3 | 3 | 3 | 3 |
| Nb. Words/Sent. | 45 | 37 | 49 | 45 |
| Nb. Words/Sum. | 136 | 111 | 147 | 135 |
| Recall | 0.32 | 0.40 | 0.40 | 0.36 |
| Precision | 0.52 | 0.26 | 0.61 | 0.50 |
| F-score | 0.36 | 0.29 | 0.44 | 0.37 |

**Table 7: Automatic summaries with three sentences**

# 4. CONCLUSION

We have presented an architecture to summarize opinionated texts which combines fairly standard features for text analysis with more exploratory features ensuing from sentiment analysis. Given that our system has had very good results for evaluation measures for quality and content, we conclude that this combination appears to be a good starting point to capture key elements that authors wish to express in subjective texts such as blog posts.

Applying a simplified version of our system to blog search engine hits summarization, our results show that automatic summarization provides the user with a better synthetic view of documents that may hold key information with regards to his search. As with fragments, automatic summaries provides local (the *simWithQuery* feature) information about a document to a user, but crucially, it also provides global (the *centroïd* feature) information about the same document that may contribute to faster completion of a search task. However, it is a matter for system designers to balance the benefit of such summaries over fragments versus implementation and processing costs. Feature work should look at the contribution of the *sentiment* feature for blog hits summarization.

# 5. ACKNOWLEDGMENTS

This work has been conducted under the project *Infom@gic* as part of the French Business Cluster *Cap Digital*.

# 6. REFERENCES

[1] T. Joachims. *Learning to Classify Text Using Support Vector Machines*. Kluwer Academic Publishers, 2001.

[2] A. Klein, I. Schwank, M. Généreux, and H. Trost. Evaluating multi-modal input modes in a wizard-of-oz study for the domain of web search. In *People and Computer XV – Interaction without Frontiers: Joint Proceedings of HCI 2001 and IHM 2001*, pages 475–483. Springer, 2001.

[3] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *EMNLP '02: Proceedings of the ACL-02 conference on Empirical methods in natural language processing*, pages 79–86, Morristown, NJ, USA, 2002. Association for Computational Linguistics.

[4] D. Radev, H. Jing, M. Styś, and D. Tam. Centroid-based summarization of multiple documents. *Information Processing and Management*, 40:919–938, December 2004.