Classifying Sentence-Based Summaries of Web Documents

Yiu-Kai D. Ng
ng@cs.byu.edu

Maria Soledad Pera

Follow this and additional works at: https://scholarsarchive.byu.edu/facpub

Part of the Computer Sciences Commons

BYU ScholarsArchive Citation
Ng, Yiu-Kai D. and Pera, Maria Soledad, "Classifying Sentence-Based Summaries of Web Documents" (2009). Faculty Publications. 116.
https://scholarsarchive.byu.edu/facpub/116

This Peer-Reviewed Article is brought to you for free and open access by BYU ScholarsArchive. It has been accepted for inclusion in Faculty Publications by an authorized administrator of BYU ScholarsArchive. For more information, please contact scholarsarchive@byu.edu, ellen_amatangelo@byu.edu.
Classifying Sentence-Based Summaries of Web Documents

Maria Soledad Pera        Yiu-Kai Ng
Computer Science Department
Brigham Young University
Provo, Utah, U.S.A.
Email: {mpera@cs.byu.edu, ng@cs.byu.edu}

Abstract

Text classification categories Web documents in large collections into predefined classes based on their contents. Unfortunately, the classification process can be time-consuming and users are still required to spend considerable amount of time scanning through the classified Web documents to identify the ones that satisfy their information needs. In solving this problem, we first introduce CorSum, an extractive single-document summarization approach, which is simple and effective in performing the summarization task, since it only relies on word similarity to generate high-quality summaries. Hereafter, we train a Naïve Bayes classifier on CorSum-generated summaries and verify the classification accuracy using the summaries and the speed-up during the process. Experimental results on the DUC-2002 and 20 Newsgroups datasets show that CorSum outperforms other extractive summarization methods, and classification time is significantly reduced using CorSum-generated summaries with compatible accuracy. More importantly, browsing summaries, instead of entire documents, classified to topic-oriented categories facilitates the information searching process on the Web.

1 Introduction

The rapid growth of the Web has dramatically increased the complexity of Web query processing, and locating relevant documents from a huge text repository has always been a significant challenge for Web users. Even though popular Web search engines are efficient and effective in retrieving information from the Web, users must scan through the retrieved documents to determine which ones are comparatively more relevant than others, which could be a time-consuming task. Hence, there is an increasing demand for advanced, scalable technologies that can identify the content of Web documents promptly. This task can be accomplished by creating a summary that captures the main content of a Web document \( D \) and reduce the time required for users to examine \( D \) prior to exploring its full content.

Text classification can take advantages of summaries to further assist Web users in locating desired information among categorized Web documents with minimized effort, such as RSS news feed, since classified Web document summaries on various topics provide a quick reference guide. Moreover, classifying summaries requires only a fraction of the processing time compared with classifying the entire documents due to the reduced size of the summaries.

In this paper, we first introduce a simple and yet effective summarization technique, called CorSum, which is based on word similarity to extract sentences from a Web document \( D \) that are representative of the content of \( D \). As opposed to other summarization approaches, CorSum does not require previous training, is computational inexpensive, and relies solely on word/sentence similarity to generate an accurate summary. In addition, CorSum is easy to implement, since the word-correlation (i.e., similarity) factors are precomputed and only simple mathematical operations, such as addition, multiplication, and division, are invoked. CorSum is not domain specific and thus can be applied to generate summaries of document collections with diverse structures and contents. Moreover, CorSum can summarize multi-lingual document collections, if the proper word-correlation factors are available. We train a Naïve Bayes classifier using CorSum-generated summaries to demonstrate the benefits of using summaries to facilitate the classification of Web documents, which include speeding up the classification process with high accuracy.

The remaining paper is organized as follows. In Section 2, we discuss existing summarization and text classification approaches. In Section 3, we present CorSum and detail the classifier we adopt for text categorization. In Section 4, we evaluate the performance of CorSum and compare its performance with others, in addition to validate the efficiency and effectiveness of the chosen classifier using CorSum-generated summaries for classification. In Section 5, we give a conclusion and directions for future work.
2 Related work

A significant number of text summarization methods have been proposed in the past which apply different methodologies to perform the summarization task. As shown in [18], extraction, fusion, abstraction, and compression are four well-established summarization techniques. Extraction identifies representative sections of a text \( T \) which yield the summary of \( T \), whereas fusion extracts and combines sentences in \( T \) to create revised sentences as the summary of \( T \). Abstraction, on the other hand, summarizes the content of \( T \) with new concise sentences, and compression removes sections in \( T \) that are considered relatively unimportant and retains the remaining sentences as the summary of \( T \). In addition, existing summarization approaches can be grouped into two categories: single-document and multi-document. Single-document summarization captures the main content of a document as its summary [3], whereas multi-document summarization creates a single summary for a document collection which describes the overall content of the collection coherently [3]. Radev et al. [18] claim that the extraction method for summarizing single documents is the most promising method, which is the strategy we adopt in our proposed (single-document) summarization approach and the focus of the subsequent discussion.

Kupiec et al. [9] determine which sentence \( S \) in a document \( D \) should be included in the summary of \( D \) via a Naïve-Bayes classifier which relies on features such as sentence length, thematic words, or uppercase words. The summarization approach in [2] uses a Hidden Markov Model (HMM) to estimate the likelihood of each sentence \( S \) in \( D \) for representing the content of \( D \) and selects the ones with the highest likelihood to be included in the summary of \( D \). The features considered for developing HMM are (i) the position of \( S \) in \( D \), (ii) the number of words in \( S \), and (iii) the probability of occurrence of a particular word in \( S \) given \( D \).

Gong [4] applies Latent Semantic Analysis (LSA) for summarization. LSA first establishes (i) inter-relationships among words in a document \( D \) by clustering semantically-related words and sentences in \( D \) and (ii) word-combination patterns that recur in \( D \) which describe a topic or concept. LSA selects the highest-scored sentences that contain recurring word patterns in \( D \) to form the summary of \( D \).

CollabSum in [20] creates the summary of \( D \) using sentences in \( D \) and documents in the cluster to which \( D \) belongs. To create the summary of \( D \), CollabSum implements four different methods: (i) Inter-Link, which captures the cross-document relationships of a sentence with respect to the others in the same cluster, (ii) Intra-Link, which reflects the relationships among sentences in a document, (iii) Union-link, which is based on the inter- and intra-document relationships \( R \), and (iv) Uniform-Link, which uses a global affinity graph to represent \( R \).

Besides using text summarization for capturing the main content of Web documents, constructed summaries can be further classified. Yang and Pedersen [22] present several feature-selection approaches for text classification and compare the performance of two classifiers, \( K \) Nearest Neighbor (KNN) and Linear List Squares Fit mapping (LLSF). The classifiers compute the confidence score \( CS \) of a document \( D \) in each category. \( CS \) in KNN is determined by the degrees of similarity of \( D \) with respect to the \( K \) nearest training documents in each category, whereas LLSF calculates \( CS \) of \( D \) in each category using a regression model based on the words in \( D \).

McCallum and Nigam [12] discuss the differences between the Multi-variate Bernoulli and Multinomial Naïve Bayes classifiers. Multi-variate Bernoulli represents a document \( D \) using binary attributes, indicating the absence and occurrence of words in \( D \), whereas the Multinomial classifier captures the content of \( D \) by the frequency of occurrence of each word in \( D \). Regardless of the classifier, the classification is performed by computing the posterior probability of each class given an unlabeled document \( D \), and assigning \( D \) to the class with the highest probability.

Nigam et al. [15] rely on Maximum Entropy to perform text classification. Maximum Entropy, which estimates probability distributions of data on a class-by-class basis, represents a document \( D \) by its word count feature. Maximum Entropy assigns \( D \) to a unique class based on the frequency of occurrence of words in \( D \) that is more alike to the word occurrence distribution of a particular class.

Using the Dempster-Shafer Theory (DST), the authors of [19] combine the outputs of several subclassifiers (trained on different feature sets extracted from the same document collection \( C \)) and determine to which class a document in \( C \) should be assigned. As claimed by the authors, sub-classifiers reduce computational time without sacrificing classification performance, and DST fusion outperforms traditional fusion methods, such as plain voting.

3 Summarization and classification

In this section, we first discuss the overall design of CorSum, our proposed extractive, single-document summarization approach, which uses the precomputed word-correlation factors to identify representative sentences in a document \( D \) to create the summary of \( D \). Hereafter, we present the Multinomial Naïve Bayes classifier, which we adopt for classifying CorSum-generated summaries of Web documents in large collections.

3.1 Our summarization approach

Mihalcea and Tarau [14] propose a sentence-extraction summarization method that applies two graph-based rank-
ing algorithms, PageRank [1] and HITS [6], to determine the rank value of a vertex (i.e., a sentence in a document D) in a graph based on the global information computed using the entire graph, i.e., the similarity of sentence pairs in D, which is calculated as a function of content overlap. Hereafter, sentences are sorted in reversed order of their rank values, and the top-ranked sentences are included in the summary of D. CorSum also depends on ranked sentences, but the rank values are computed according to (i) the word-correlation factors introduced in [7], and (ii) the degrees of similarity of sentences. The higher ranked sentences are the most representative sentences of D, which form the summary of D.

3.1.1 Word-Correlation factors and sentence similarity

The word-correlation matrix $M$ [7], which includes the correlation factors of non-stop, stemmed words, is a 54,625 × 54,625 symmetric matrix. The correlation factor of any two words $w_i$ and $w_j$, which indicates how closely related $w_i$ and $w_j$ are semantically, is computed based on the (i) frequency of co-occurrence and (ii) relative distance of $w_i$ and $w_j$ in any document D and is defined as follows:

$$wcf(w_i, w_j) = \sum_{x \in V(w_i)} \sum_{y \in V(w_j)} \frac{1}{d(x, y)}$$

(1)

where $d(x, y)$ denotes the distance (i.e., the number of words in) between $x$ and $y$ in $D$ plus 1, and $V(w_i)$ (V($w_j$), respectively) is the set of words that include $w_i$ ($w_j$, respectively) and its stem variations in a document.

The word-correlation factors in $M$ were computed using the Wikipedia Database Dump (http://en.wikipedia.org/wiki/Wikipedia:Databasedownload), which consists of 930,000 documents written by more than 89,000 authors on various topics, and hence is diverse in content and writing styles and an ideal choice for measuring word similarity. Using the word-correlation factors in $M$, we compute the degree of similarity of any two sentences $S_1$ and $S_2$ by adding the word-correlation factors of each word in $S_1$ with every word in $S_2$ as follows:

$$Sim(S_1, S_2) = \sum_{i=1}^{n} \sum_{j=1}^{m} wcf(w_i, w_j)$$

(2)

where $w_i$ ($w_j$, respectively) is a word in $S_1$ ($S_2$, respectively), $n$ ($m$, respectively) is the number of words in $S_1$ ($S_2$, respectively), and $wcf(w_i, w_j)$ is given in Equation 1. $Sim(S_2, S_1)$ can be defined accordingly.

\[\text{Stopwords are commonly-occurred words such as articles, prepositions, and conjunctions, which are poor discriminators in representing the content of a sentence (or document), whereas stemmed words are words reduced to their grammatical root.}\]

3.1.2 Most representative sentences

The goal of CorSum is to identify sentences in a document $D$ that most accurately represent the content of $D$ during the summarization process. To determine which sentences should be included in the summary of $D$, CorSum computes the overall similarity of each sentence $S_i$ in $D$, denoted $OS(S_i)$, with respect to the other sentences in $D$ as

$$OS(S_i) = \sum_{j=1, j \neq i}^{n} Sim(S_i, S_j)$$

(3)

where $n$ is the number of sentences in $D$, $S_i$ and $S_j$ are sentences in $D$, and $Sim(S_i, S_j)$ is defined in Equation 2.

We rely on the Odds ratio [5], which is defined as the ratio of the probability ($p$) that an event occurs to the probability (1 - $p$) that it does not, i.e., $Odds = \frac{p}{1 - p}$, to compute the Rank value of $S_i$ in $D$. We treat $OS(S_i)$ as the positive evidence of $S_i$ in representing the content of $D$. The Rank value of $S_i$, which determines the content significance of $S_i$ in $D$, is defined as

$$Rank(S_i) = \frac{OS(S_i)}{1 - OS(S_i)}$$

(4)

Having computed the Rank value of each sentence in $D$, CorSum chooses the top $N$ ($\geq 1$) higher ranked sentences in $D$ as the most representative sentences of $D$ to create the summary of $D$. In establishing the proper value of $N$, i.e., the number of representative sentences to be included in a summary, we follow the results of a study conducted by [13] on two different popular datasets (i.e., Reuters-21578 and WebKB) using different classifiers, which concludes that in general a summary with six sentences can accurately represent the overall content of a document. More importantly, Mihalcea et al. [13] show in their study that using summaries with six sentences for clustering/classification purpose achieves the highest accuracy, an assumption we adopt in designing CorSum. If a document $D$ contains less than six sentences, CorSum includes the entire content of $D$ as the summary of $D$.

Example 1 Figure 1 shows a document $D$ from the DUC-2002 dataset (to be introduced in Section 4) in which the six most representative sentences (i.e., sentences with their Rank values higher than the remaining ones) are highlighted, whereas Table 1 shows the Rank values of the sentences in $D$ and Table 2 includes the degrees of similarity of the highest ranked (i.e., Sentence 10) and lowest ranked (i.e., Sentence 11) sentences with the remaining sentences in $D$. [\[\square\]

3.2 The Naïve Bayes classifier

To verify that classifying text documents using their summaries, instead of the entire documents, is cost-
effective, we apply a Naive Bayes classifier, which is one of the most popular text classification tools, since Naive Bayes is simple, easy to implement, robust, highly scalable, and domain independent [8], to classify CorSum-generated summaries. Moreover, even though Naive Bayes assumes mutual independence of attributes, it achieves high accuracy in text classification [12].

Among the variants of the Naive Bayes classifier, we choose the multinomial implementation of the Naive Bayes classifier [12], denoted MNB, since MNB is one of the most widely-used text classifiers [8]. MNB uses the frequency of word occurrence to compute the probability of a document to be assigned to a particular class. During the training process, MNB first estimates the probability of a word \( w_i \) in a natural class \( c_j \), which is based on the frequency of occurrence of \( w_i \) in each pre-classified, labeled document \( d_k \), i.e., a CorSum-generated summary in our case, in a collection \( D \), and is formally defined as

\[
P(w_i | c_j) = \frac{1 + \sum_{i=1}^{[D]} N_{it} P(c_j | d_i)}{|V| + \sum_{s=1}^{[V]} \sum_{i=1}^{[D]} N_{is} P(c_j | d_i)}
\]  

where \( N_{it} \) (\( N_{is} \), respectively) denotes the frequency of occurrence of \( w_i \) (word \( w_s \), respectively) in \( d_i \). \(|D|\) is the number of documents in \( D \), \(|V|\) is the number of distinct words in \( D \), and \( P(c_j | d_i) \) is 1, if \( c_j \) is the pre-assigned label of \( d_i \), i.e., \( d_i \) is pre-assigned to \( c_j \), and 0, otherwise.

Having determined the required probabilities during the training step of MNB, the probability that a given (unlabeled) document \( d_k \) belonged to the class \( c_j \), denoted \( P(d_k | c_j) \), is computed at the classification phase as

\[
P(d_k | c_j) = P(|d_k| | c_j) \prod_{t=1}^{[V]} \frac{P(w_t | c_j)^{N_{kt}}}{N_{kt}!}
\]  

where \( |d_k| \) denotes the number of words in \( d_k \), and \(|V|\), \( N_{kt} \), and \( P(w_t | c_j) \) are as defined in Equation 5, and the probability of a document \( d_k \) in \( D \) is defined as

\[
P(d_k) = \sum_{j=1}^{[C]} P(c_j) P(d_k | c_j)
\]  

where \([C]\) is the number of predefined natural classes, \( P(c_j) \) is the fraction of documents in \( D \) that belong to \( c_j \), which is determined at the training phase, and \( P(d_k | c_j) \) is as computed in Equation 6.

In classifying \( d_k \), i.e., a summary in our case, MNB assigns to \( d_k \) the class label \( c_j \), if the computed probability \( P(c_j | d_k) \) is the highest among all the probabilities \( P(c_i | d_k) \) for each predefined natural class \( c_i \). \( P(c_j | d_k) \) is computed as follows, which is the well-known Bayes’ Theorem:

\[
P(c_j | d_k) = \frac{P(c_j) P(d_k | c_j)}{P(d_k)}
\]  

where \( P(d_k | c_j) \), \( P(d_k) \), and \( P(c_j) \) are as defined earlier.

During the implementation process of MNB, the probability of a word in a given class is smoothed by using the Laplace approach, also known as add-one smoothing [12], which adds the values 1 and \(|V|\) as shown in Equation 5.
Table 1. Rank values of the sentences in the document shown in Figure 1

<table>
<thead>
<tr>
<th>Sentence</th>
<th>Rank value</th>
<th>Sentence</th>
<th>Rank value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-1.100</td>
<td>7</td>
<td>-1.047</td>
</tr>
<tr>
<td>2</td>
<td>-1.050</td>
<td>8</td>
<td>-1.090</td>
</tr>
<tr>
<td>3</td>
<td>-1.050</td>
<td>9</td>
<td>-1.055</td>
</tr>
<tr>
<td>4</td>
<td>-1.055</td>
<td>10</td>
<td>-1.045</td>
</tr>
<tr>
<td>5</td>
<td>-1.070</td>
<td>11</td>
<td>-1.142</td>
</tr>
<tr>
<td>6</td>
<td>-1.083</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2. The degrees of similarity of Sentence 10 (the highest ranked) and 11 (the lowest ranked) with respect to the others in the document shown in Figure 1

\[
\begin{array}{c|c|c|c|c}
S_i = 10 & S_i = 11 \\
S_j & \text{Sim}(S_i,S_j) & S_j & \text{Sim}(S_i,S_j) \\
1 & 0.000007 & 1 & 1.000000 \\
2 & 2.000008 & 2 & 1.000003 \\
3 & 6.000005 & 3 & 0.000001 \\
4 & 3.000000 & 4 & 2.000000 \\
5 & 2.000003 & 5 & 0.000001 \\
6 & 0.000007 & 6 & 0.000003 \\
7 & 3.000004 & 7 & 1.000001 \\
8 & 2.000002 & 8 & 0.000001 \\
9 & 3.000001 & 9 & 1.000001 \\
10 & 1.000000 & 10 & 1.000000 \\
\end{array}
\]

Probability smoothing is often applied to solve the zero-probability problem that occurs when a word not seen in training is in a document to be classified [17, 21].

4 Experimental results

In this section, we describe the datasets used for experimentation and present the evaluation measures which are adapted for (i) quantifying the quality of CorSum-generated summaries, (ii) comparing the performance of CorSum with other well-known extractive summarization approaches, and (iii) demonstrating the effectiveness and efficiency of using MNB for classifying CorSum-generated summaries, instead of entire documents.

4.1 Datasets

To assess and compare the performance of CorSum, we rely on the widely-used Document Understanding Conference (DUC) 2002 dataset (http://www-nlpir.nist.gov/projects/duc/past_duc/duc2002/data.html). DUC-2002 includes 533 news articles divided into 60 clusters, each containing approximately 10 articles retrieved from popular news collections such as the Wall Street Journal, AP Newswire, Financial Times, and LA Times. The dataset also includes two summaries, called reference summaries, created by human experts for each news article, based on which the performance of any single-document or multi-document summarization approach can be evaluated.

To determine the suitability of using summaries, instead of the entire documents, in text classification, we applied the MNB classifier on the 20 Newsgroup (http://kdd.ics.uci.edu/databases/20newsgroups/20newsgroups.html) dataset from where our CorSum-generated summaries are created. We rely on 20NG for evaluating the performance of the MNB classifier using (CorSum-generated) summaries, since 20NG is a popular news document collection used for verifying the accuracy of text classification or text clustering tools. The 20 Newsgroup dataset (20NG) [10] consists of 19,997 articles retrieved from the Usenet newsgroup collection that are clustered into 20 different categories. For evaluation purpose, 80% of the documents in 20NG were used for training MNB, and the remaining 20% for classification evaluation.

4.2 Evaluation measures

To evaluate the performance of CorSum, we have implemented a widely used summarization measure, the Recall-Oriented Understudy for Gisting Evaluation (ROUGE) [11]. ROUGE includes measures that quantify the quality of a summary S created using a summarization approach by comparing the number of overlapping units, such as n-gram, word sequences, or word pairs, in S with the ones in the expert-created reference summaries of the same document. Four different ROUGE measures are known for summarization evaluation: ROUGE-N, ROUGE-L, ROUGE-W, and ROUGE-S, which are based on n-gram overlap, least-common substrings, weighted least-common substrings, and skipping bigrams co-occurrence, respectively. We have considered ROUGE-N, as opposed to the other ROUGE variations, since as shown in [11], the unigram-based ROUGE-N score, i.e., ROUGE-1, is the most accurate measure for establishing the closeness between automatically-generated summaries and their corresponding reference summaries. In addition, ROUGE-N for N = 1 and N = 2) is the most ideal choice for evaluating short summaries and single-document summaries [11]. ROUGE-N of an automatically-generated summary S is defined as

\[
\text{ROUGE-N} = \frac{\sum_{HS \in \text{ReferenceSummaries}} \sum_{\text{gram}_n \in HS} \text{Count}_{\text{match}}(\text{gram}_n)}{\sum_{HS \in \text{ReferenceSummaries}} \sum_{\text{gram}_n \in HS} \text{Count}(\text{gram}_n)}
\]

where \(\text{gram}_n\) is an n-gram, \(\text{Count}_{\text{match}}(\text{gram}_n)\) is the number of common n-grams in S and one of the corre-
sponding reference summaries HS, and Count(gram_n) is the number of n-grams in HS.

We have used ROUGE-1 and ROUGE-2, i.e., compared the unigram and bigram overlap between S and HS, respectively, to assess the performance of CorSum and other summarization techniques discussed in Section 2.

To evaluate the performance of MNB on summaries, instead of the corresponding entire documents, we use the classification accuracy measure as defined below.

\[
\text{Accuracy} = \frac{\text{Number of Correctly Classified Documents}}{\text{Total Number of Documents in a Collection}}
\]

4.3 Performance evaluation of CorSum

We have verified the effectiveness of CorSum based on the ROUGE-1 and ROUGE-2 evaluation measures, which calculate the number of overlapped n-grams \((1 \leq n \leq 2)\) in each CorSum-generated summary and the corresponding reference summaries. Table 3 shows the ROUGE-1 and ROUGE-2 values computed using CorSum-generated and the reference summaries of a few documents in the DUC-2002 dataset. On an average, CorSum obtained 0.54 (0.27, respectively) ROUGE-1 (ROUGE-2, respectively) value, which implies that 54% (27%, respectively) of the unigrams (bigrams, respectively) in the reference summaries are included in its corresponding CorSum-generated summary. ROUGE-1 is twice as high as ROUGE-2, which is anticipated, since ROUGE-1 considers overlapped unigrams which includes overlap of stop-words in the summaries, whereas bigrams limit the extent of overlap between two summaries as a result of matching two adjacent words, which is lower in probability than matching two single words. More importantly, as previously stated, we compare CorSum-generated summaries with reference summaries that are not extractive, which were created by human experts using new sentences that capture the gist of the test documents. For this reason, achieving high ROUGE-1 and ROUGE-2 values is not a simple task. A high ROUGE-N value provides further evidence of the high quality of the summary created using CorSum compared with other summarization methods (as shown in Section 4.3.1), even though CorSum-generated summaries are extractive and its creation process is relatively simple and straightforward compared with the rewriting approach.

Example 2 Figures 1 and 2 show the CorSum-generated summary CS and one of its corresponding reference summary RS in the DUC-2002 dataset, respectively. Out of the 99 words in RS, 80 of its unigrams and 41 of its bigrams are in CS. Some sentences in RS have been rephrased and include synonyms of words used in the original document d, which result in mismatched unigrams and bigrams between CS and RS. However, CorSum extracts sentences in d that are highly similar to the ones in RS and achieve higher ROUGE-N values compared with other existing summarization approaches, which have been verified.

4.3.1 Comparing the performance of CorSum

To further assess the effectiveness of CorSum on summarization, we compared CorSum’s performance, in terms of ROUGE-1 and ROUGE-2 measures, with other well-established summarization approaches that adopt different methodologies for text summarization (as discussed in Section 2) using the DUC-2002 dataset. The various summarization strategies to be compared include the ones implemented in CollabSum [20], i.e., Uniform-Link, Union-Link, Inter-Link, and Intra-Link, which rely on inter- and intra-document relationships in a cluster to generate a summary, the Latent Semantic Analysis (LSA) summarization method in [4], and the Top-N method in [16]. As shown in Figure 3, CorSum outperforms all of these summarization approaches by 5-8% (8-10%, respectively) based on unigrams (bigrams, respectively), which verifies that CorSum-generated summaries are more reliable in capturing the content of documents than their counterparts.
4.3.2 Observations on the summarization results

The summarization approach based on LSA [4] chooses the most informative sentence in a document \(D\) for each salient topic \(T\), which is the sentence with the largest index value on \(T\). The drawback of this approach is that sentences with the largest index values of a topic, which may not be the overall largest among all topics, are chosen even if they are less descriptive than others in capturing the content of \(D\).

The Top-N summarizer [16] is considered a naive summarization approach, which extracts the first \(N\) (= 6, in our study) sentences in a document \(d\) as its summary and assumes that introductory sentences contain the overall gist of \(d\). The Top-N summarization approach is applicable to documents that contain an outline in the beginning paragraph. As shown in Figure 3, the Top-N summarizer achieves relatively high performance, even though its accuracy is still lower than CorSum, since the news articles in DUC-2002 dataset are structured such that the first few lines of each article often contain an overall gist of the article. However, the Top-N approach is not suitable for summarizing general text collections with various document structures, as opposed to CorSum which is domain-independent.

Since the summarization approaches in CollabSum, i.e., Inter-Link, Intra-Link, Uniform-Link, and Union-Link [20], must capture the inter- and intra-document relationships of documents in a cluster to generate the summary of a document, this process increases the overall summarization time. More importantly, inter- and intra-document information used by CollabSum are not as sophisticated as the word-correlation factors of CorSum in capturing document contents, since CollabSum approaches yield lower ROUGE-N values than CorSum (as shown in Figure 3).

4.4 Classification performance evaluation

We have evaluated the effectiveness and efficiency of classifying summaries, as opposed to entire documents, using MNB on the 20NG dataset. Figure 4 shows the classification accuracy achieved by MNB using automatically-generated summaries, as well as the entire content of the documents, in 20NG for comparison purpose. Using CorSum-generated summaries, MNB achieves a fairly high accuracy, i.e., 73%, even though using the entire documents MNB achieves a higher classification accuracy of 82%, which is less than 10% difference. However, the training and classification processing time of MNB is significantly reduced when using CorSum-generated summaries as opposed to the entire document contents as shown in Figure 4—the processing time required for training the MNB classifier and classifying on entire documents is reduced by approximately 70% when using CorSum-generated summaries. In comparing with the classification accuracy of Top-N and LSA summaries, CorSum outperforms both of them. This is because using CorSum-generated summaries, MNB can extract more accurate information based on the probability of words belonged to different classes (as computed in Equation 5) in a labeled document collection, which translates into fewer mistakes during the classification process of MNB. Furthermore, MNB performs classification faster using the CorSum-generated summaries than the LSA or Top-N summaries as shown in Figure 4.

In Figure 5, we show the classification accuracy using MNB on CorSum-generated summaries for each natural class in 20NG, and the corresponding labeled classes are (1) sci.electronics, (2) comp.sys.mac.hardware, (3) soc.religion.christian, (4) comp.windows.x, (5) comp.sys.ibtm.pc.hardware, (6) comp.graphics, (7) misc.forsale, (8) rec.motorcycles, (9) comp.os.ms-windows.misc, (10) rec.sport.hockey, (11) talk.politics.misc, (12) alt.atheism, (13) sci.crypt, (14) talk.politics.guns, (15) rec.sport.baseball, (16) sci.space, (17) talk.politics.mideast, (18) sci.med, (19) rec.autos, and (20) talk.religion.misc. Figure 5 also shows the number of false positives, which is the number of documents assigned to a class when they should not be, and false negatives, which is the number of documents that were not assigned to a class to which they belong. We observe that except for classes 4, 5, 6, and 8, the average number of false positives for each of the remaining classes in 20NG is 30, which constitutes approximately 12% of the classified news articles. The same applies to the number of false negatives. Except for classes 1, 11, 14, 16, and 18, the average number of mislabeled articles is 33, which constitutes approximately 13% of the articles used for the classification purpose. The overall average number of false positives and false negatives is 41, an average of 23%, per class.
5 Conclusions

Locating relevant information on the Web in a timely manner is often a challenging task, even using well-known Web search engines, due to the vast amount of data available for the users to process. Although retrieved documents can be pre-categorized based on their contents using a text classifier, Web users are still required to analyze the entire documents in each category (or class) to determine their relevance with respect to their information needs. To assist Web users in speeding up the process of identifying relevant Web information, we have introduced CorSum, an extractive summarization approach which requires only precomputed word similarity to select the most representative sentences of a document $D$ (that capture its main content) as the summary of $D$. We have also used CorSum-generated summaries to train a multinomial Naive Bayes (MNB) classifier and verified its effectiveness and efficiency in performing the classification task. The empirical study conducted using the DUC-2002 dataset has verified that CorSum creates high-quality summaries and outperforms other well-known extractive summarization approaches. Furthermore, applying the MNB classifier on CorSum-generated summaries of the news articles in the 20 Newsgroup dataset, we have validated that in classifying a large document collection $C$, the classification task using CorSum-generated summaries is in the order of magnitude faster than using the entire content of documents in $C$ with compatible accuracy.

For future work, we will consider applying feature extractors and selectors, such as sentence length, topical words, mutual information, or loglikelihood ratio, on a classifier to (i) assess the degree of enhancement on classification accuracy using CorSum-generated summaries and (ii) minimize the classifier’s training and classification time.

References