Query-Relevant Sentiment Summarization Based on Facet Identification and Sentence Clustering

Matthew Moulton, Sophie Gao, and Yiu-Kai Ng
Computer Science Department
Brigham Young University
Provo, Utah, USA

moulton.matthewj@gmail.com,sophiegao99@gmail.com,ng@compsci.byu.edu

ABSTRACT

Multi-document sentiment analysis is an important natural language processing problem. Summaries generated by these analyzers can greatly reduce the time necessary to read a collection of topically-related documents to locate the desired information needs of a user. With the ever-increasing globalization and technology of the modern day, analysis of online user reviews on different products is an especially pertinent application of the aforementioned problem. At present there are way too many user reviews on popular products for potential buyers to spend adequate time to read and extract the most salient product details and opinions of previous buyers. In solving this problem, we propose a fully-automated summarizer to reduce the workload of online customers. The proposed system takes a user query and extracts the most relevant and essential comments made by individual reviewers. As opposed to existing multi-document summarization approaches, our summarizer compiles comprehensive reviews by extracting important facets and sentiment information based on various sentence features rather than applying complex machine learning algorithms. The design of our summarizer is easy to understand and implement, without the required massive training data and excessive training time. The conducted empirical study shows that the proposed summarization system outperforms current state-of-the-art multi-document sentiment summarization approaches.

CCS CONCEPTS

• **Information systems** → **Information retrieval**; *Retrieval models and ranking*; *Similarity measures*; • **Applied computing** → **Document management and text processing**.

KEYWORDS

Sentiment analysis, summarization, sentence features, user reviews

ACM Reference Format:
Matthew Moulton, Sophie Gao, and Yiu-Kai Ng. 2022. Query-Relevant Sentiment Summarization Based on Facet Identification and Sentence Clustering. In *The 37th ACM/SIGAPP Symposium on Applied Computing (SAC '22), April 25–29, 2022, Virtual Event, .* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3477314.3507125

ACM SAC 2022, April 25-29, 2022,

ACM ISBN 978-1-4503-8713-2/22/04…$15.00
https://doi.org/10.1145/3477314.3507125

1 INTRODUCTION

With the advance of the information technology, there is an abundance of data to be analyzed and understood, which is applicable to the task of processing sentiment of data. Extracting the opinions contained in the document text, however, is not a straightforward nor simple task. Sometimes the author of these documents can have mixed feelings about the topic of concern overall, or like some features of a product but not others. For this reason, multi-document sentiment analysis is a challenging and time-intensive process for people to handle manually. Research has shown that the average product review is 582 characters in length, with some outliers at upwards of 30,000 [21] and an average adult can read about 987 characters per minute [20]. It would take an ordinary person about 6 minutes to read 10 reviews. To appropriately form a justified opinion, a user would need to thoroughly investigate several related products in an attempt to select the best one. With the time constraint, users tend to read only a few reviews or skim through them. In fact, over 51% of online customers read fewer than 5 reviews before purchasing a product [3]. This is problematic, since a user can miss important details and overvalue others.

To reduce the workload of analyzing sentiment embedded in reviews and the amount of text to be considered, multi-document sentiment analysis systems have been developed in the past [2, 19]. These systems create a short snippet of the documents to be processed instead of the documents in their entirety, which offer the users a more reasonable time frame to extract the desired information. As the usefulness of such an approach is dependent upon the accuracy of representing the content of the snippets, it is critical for the summary of the reviews contain the most significant sentiments and salient details of the corresponding documents.

To facilitate the task of synthesizing opinions expressed in user reviews on a particular product, we introduce a query-based, multi-document sentiment summarization approach. Our solution is focused on generating multi-document sentiment summaries that are (i) cohesive, (ii) non-redundant, and (iii) diverse in terms of user opinion views. In addition, summaries generated by us achieve high coverage (of information included in the summary) and contain meaningful and relevant sentences. Our approach is simple and its summaries serve as a product guide to the users.

During the process of creating a summary, we (i) identify products, facets, and sentiment keywords in a query to determine the information needs of a user, (ii) detect different facets of a product P and cluster sentences in online reviews *on-the-fly*, (iii) find

the *most-informative* sentences in a top-rated review that capture the expressed opinions on P, and (iv) generate a *concise* summary of multiple reviews for answering the information need expressed in a user (sentiment) query. Our research work advances the technologies in developing summarization approaches on user reviews.

2 RELATED WORK

The area of study in extractive, sentimentally-representative summarization of online content has been conducted in the past. Huang et al. [10] present a summarization approach whereby "meaningful content" can be extracted from any arbitrary XML document for processing, and Bahrainian and Dengel [2] utilize extensive preprocessing on Twitter and other social media posts, which include removing URLs, stripping punctuation in hashtags, and splitting each tweet into smaller text segments based upon punctuation.

Ni et al. [17] devise a system using a trained classifier to select justification sentences from a review. These sentences contain facets of a product that assist a user in making a choice of favorability and other users in deciding between several potential options. Their system employed *BERT* developed by Devlin et al. [5] to fine-tune the label classifier. Aker et al. [1], on the other hand, propose a graph-based approach to labeling of topic clusters for any type of documents from reader comments to online news articles. They adopt Word2Vec word embeddings for clustering topic labels.

Jeong et al. [11] design a three-part system to extract keywords, generate an extractive summary of the text, and provide a simple search engine to help users find desired documents. Keywords have their importance estimated with statistical relevance weighting and are sorted thereupon. Ganesan and Zhai [8] introduce a special type of document ranking based on the individual qualities of multiple facets. The ranking, however, requires a specific type of query that consists of explicit sub-queries delimited by commas.

In generating a snippet for a document, the procedure outlined by He et al. [9] considers not only sentences that contain query keywords, but also selecting sentences for the snippet that are representative of the document as a whole. In analyzing sentiment, Farooq et al. [7] focus on the effect of negation on the sentiment polarity of a document. They identified and treated three types of negation differently: syntactic, morphological, and diminishers. Nallapati et al. [16] employ a summarization approach that relies on neural networks and is uniquely interpretable by allowing for the visualization of abstractive details. These document details include information content, salience, and novelty.

3 OUR SENTIMENT SUMMARIZER

Our sentiment summarizer on user reviewers addresses various key design issues of a summarization system, which include simplicity to build, easy to use, and capable of capturing essential information. Our summarizer does not rely on complex machine learning algorithms. Although very helpful in different applications, machine learning approaches can be complicated to design and develop and require a training process using abundance of training data before becoming functional. We introduce a simple, and yet effective, sentiment summarization system that tailors towards the information needs of users. The information filtering and sentiment analysis procedures are straightforward, which are simply based on sentence scoring and ranking. We apply part-of-speech tagging for facet detection, various sentence heuristics to compute the score of a sentence in user reviews to determine its degree of significance in capturing essential information, and a sentence clustering approach to *avoid redundancy* and *maximize* the *coverage* of information included in a summary that meet the user's information needs, which are the novelty of our summarizer.

3.1 Identifying Users' Information Needs

Given a user query Q, which inquires on feedback on a particular product[1], we detect and segregate non-stop (key)words in Q, a process which identifies *products* and *facets* that describe different aspects of a product, and filters *sentiment keywords* from *non-essential* ones such as stopwords, to recognize the information needs specified in Q. In accomplishing this task, we adopt a one-against-all [14] implementation of a multi-class SVM to identify information needs expressed in a query, which is a robust methodology that achieves state-of-the-art performance on classification.

To develop a multi-class SVM, each instance of the SVM is an input vector of a non-numerical, non-stopword[2] K in a query Q and is a succession of '1' ('0', respectively), each of which represents the presence (absence, respectively) of an SVM feature F (defined by us below) if F applies (does not apply, respectively) to K.

- **Capitalized** is set if the first letter of K is capitalized. The first character of a *product* is often capitalized.
- **Adjective** is set if K is given an *adjective* part-of-speech (POS) tag. We employ the Stanford POS tagger[3] for assigning POS tags, such as noun, verb, adjective, etc., to keywords (in Q). *Sentiment keywords* specified in Q are often adjectives which describe different aspects of a product specified in Q.
- **Sentiment** is set if K is a *sentiment keyword*, which is determined by using a list of more than 4,000 sentiment keywords provided by the General Inquirer[4].
- **After-Preposition** is set if K appears immediately after a preposition, identified using the *Stanford POS* tagger. Both *products* and *facets* tend to occur after a preposition in Q.
- **After-Apostrophe** is set if K appears immediately after a term in a Saxon genitive form, i.e., a traditional term for the apostrophe-s. *Facets* often appear after a term in the Saxon genitive form in Q.
- **Before-Sentiment** is set if K appears immediately before a sentiment keyword in Q. Both *products* and *facets* are often followed by a sentiment keyword in Q.
- **Stopword** is set if K is a stopword, which is a *non-essential* term. We compiled our own list of 865 stopwords using multiple stopword lists posted online for this feature.
- **Is-5W1H** is set if K is one of the keywords frequently used in formulating questions, i.e., "what", "when", "where", "who",

[1] Since the design goal of our sentiment summarization approach is to synthesize archived feedback provided by web users on services and products, we process only queries with products explicitly specified.
[2] *Stopwords* are commonly-occurred words, such as articles, prepositions, and conjunctions, which carry little meaning.
[3] nlp.stanford.edu/software/tagger.shtml
[4] wjh.harvard.edu/~inquirer/homecat.htm

Table 1: Keyword types that are identified by each of the previously introduced SVM features

SVM Features	Product	Facet	Sentiment Keyword	Non-Essential Term
Capitalized	X			
Adjective			X	
Sentiment			X	
After-Preposition	X	X		
After-Apostrophe		X		
Before-Sentiment	X	X		
Stopword				X
5W1H				X

"why", and "how". 5W1H terms are treated as *non-essential* terms, since "when", "where", "who", and "why" do not appear often in sentiment questions, whereas "what" and "how", which appear more often, do not have a direct impact on the information needs specified in users' questions.

To verify that each of the chosen SVM features listed above is accurate in identifying keywords (in users' queries) that are either *Products*, *Facets*, *Sentiment Keywords*, or *Non-Essential Terms*, we conducted an empirical study using a dataset, denoted *Property-DS*, which does not overlap with the dataset introduced in Section 4.1, for analyzing the performance of our multi-class SVM. *Property-DS* consists of 3,000 opinion questions randomly extracted from Yahoo! Answers[5] and WikiAnswers[6]. Keywords in each of the questions in *Property-DS* were identified as products, facets, sentiment keywords, or non-essential terms by independent assessors prior to conducting the evaluation.

Table 1 shows the types of SVM features that are supposed to be identified, and we computed the percentages of keywords (in the questions in *Property-DS*) belonged to each keyword type, i.e., *Products, Facets, Sentiment Keywords*, and *Non-Essential Terms*, that are accurately identified by the aforementioned SVM features. The accuracy ratios of identifying *Product*-type keywords in *Property-DS* are 96%, 92%, and 83%, respectively that are either *capitalized*, appear *after* a *preposition*, or appear *before* a *sentiment* keyword, whereas the percentage of misclassified *products* in the 3,000 questions in *Property-DS* identified by each remaining SVM feature is below 15%. For the *Facet* keywords, the accuracy ratios are 80%, 90%, and 85% for appearing *after* a *preposition*, *after* an apostrophe, and *before* a *sentiment* keyword, respectively, whereas the accuracy ratio for detecting *Sentiment*-type keywords are 90% and 97% for appearing as *adjective* and *sentiment*, respectively. As expected, our SVM achieves a 100% accuracy in recognizing all the *stopwords* and *5WIH* keywords as non-essential terms. The empirical study validates that the chosen SVM features used by our multi-class SVM adequately classify the types of keywords as designed.

3.2 Creating Sentence Clusters

In this section, we first introduce our approaches in creating and ranking sentence cluster labels for downstream processing. Hereafter, we discuss our strategy in choosing sentences extracted from user reviews that are assigned to different labeled clusters to be included in the summary generated in response to a user query.

3.2.1 Creating Cluster Labels. We create concise and accurate cluster labels that reflect the *facets* mentioned in the top-100 user reviews[7], denoted *TopRev*, using the suffix array algorithm, which has been proved to be *efficient* and *effective* in discovering key phrases in large text collections [4]. The algorithm generates a list of cluster labels by simply extracting all the suffixes in reviews that are sorted alphabetically. Since the generated list of suffixes may include labels that are not representative of facets describing the product P in *TopRev*, we removes labels that (i) are *numeric*, (ii) cross sentence *boundaries*, since sentence markers indicate a topical shift, (iii) are *incomplete*, i.e., included as substrings in other labels, (iv) end in the *Saxon genitive form*, or (v) are *sentiment* keywords, i.e., terms that express a positive or negative polarity (which are considered only in generating our sentiment summaries).

3.2.2 Ranking Cluster Labels. To capture the content-significance of the created cluster labels, we proceed to rank the labels using various measures, which are effective in identifying representative cluster labels and are defined as follows:

- The **frequency** of a label L, denoted $Freq(L)$, reflects the *frequency of occurrence* of L in the top-100 user reviews T. The higher $Freq(L)$ is, the higher the ranking position of L among the cluster labels.
- The **stability** of a label, denoted $Stability(L)$, measures the *mutual information* (i.e., dependence[8]) of L. Given that L may contain multiple keywords, i.e., L = "$c_1 \ldots c_n$", where c_i $(1 \leq i \leq n)$ is a keyword in L, $Stability(L)$ is defined as

$$Stability(L) = \frac{f(L)}{f(L_L) + f(L_R) - f(L)} \quad (1)$$

  where L_L = "$c_1 \ldots c_{n-1}$", L_R = "$c_2 \ldots c_n$", and $f(L)$, $f(L_L)$, and $f(L_R)$ are the *frequencies of occurrence* of L, L_L, and L_R, respectively in T.
- The **significance** of a label L, $Sig\_L(L)$, is a function that assigns *more* weight to *longer* cluster labels, since longer labels are more meaningful, i.e., descriptive.

$$Sig\_L(L) = f(L) \times g(|L|) \quad (2)$$

  where $f(L)$ is the *frequency of occurrence* of L in T, $|L|$ is the number of keywords in L, and $g(x)$ is a function such that $g(1) = 0$, $g(x) = log_2\, x$ (if $2 \leq x \leq 8$), and $g(x) = 3$ (if $x > 8$).

We compute a *ranking score* for each cluster label L, denoted $LRank(L)$, which reflects the significance of L in capturing the content of the reviews in T, by combining *Freq*, *Stability*, and Sig_L[9] of L using the *Stanford Certainty Factor* [15].

$$LRank(L) = \frac{Freq(L) + Stability(L) + Sig\_L(L)}{1 - Min\{Freq(L), Stability(L), Sig\_L(L)\}} \quad (3)$$

[5] answers.yahoo.com
[6] www.answers.com
[7] One hundred reviews is an *ideal* set for creating summaries [6].
[8] *Dependence* identifies labels that characterize the contents of sentences in one cluster in contrast to others. The higher the *mutual* information of L is, the more *dependent* L is as a cluster label.
[9] Since *Freq*, *Stability*, and Sig_L are in different numerical scales, we first normalize the values using a logarithmic equation so that they are in the same range.

3.2.3 Assigning Sentences to Clusters. Using the set of identified cluster labels, we assign sentences in the top-100 user reviews, i.e., *TopRev*, to different clusters using the word-correlation factors. The *word-correlation factors* (wcf) in our *word-similarity* matrix, denoted $WS\text{-}matrix$, is a 54,625 × 54,625 symmetric matrix. $WS\text{-}matrix$ was generated using a set of approximately 880,000 Wikipedia documents[10] written by more than 89,000 authors on various topics and writing styles. The wcf of any two words[11] is computed using their (i) *frequency of co-occurrence* and (ii) *relative distances* in each Wikipedia document.

$$wcf(i,j) = \frac{\sum_{D \in Wiki}\left(\frac{\sum_{k_i \in D}\sum_{k_j \in D}\frac{1}{d(k_i,k_j)+1}}{N_i \times N_j}\right)}{|Wiki|} \tag{4}$$

where $|Wiki|$ is the number of documents in the Wikipedia collection, i.e., $Wiki$, $d(k_i, k_j)$ denotes the *distance* (i.e., the number of words in) between words i and j or their stems in a Wiki document D in which they co-occur, and N_i (N_j, respectively) is the number of times word i (j, respectively) and its *stems* variations appeared in D. Compared with WordNet[12] in which each pair of words is not assigned a *similarity weight*, word-correlation factors offer a more sophisticated measure of word similarity.

To cluster sentences in *TopRev* that address the same or similar facets, we compute the *degree of similarity* between each sentence S and label L in the set of identified cluster labels as follows:

$$LS\_Sim(L,S) = \frac{\sum_{i=1}^{|S|}\sum_{j=1}^{|L|} wcf(w_i, w_j)}{|S|} \tag{5}$$

where $|S|$ ($|L|$, respectively) is the number of words in S (L, respectively), w_i (w_j, respectively) is a keyword in S (L, respectively), and $wcf(w_i, w_j)$ is the word-correlation factor of w_i and w_j. Since the longer S is, the higher $LS\_Sim(L, S)$ is, we normalize $LS\_Sim(L, S)$ by dividing the accumulated word-correlation factors by the number of words in S, i.e., $|S|$.

Having computed $LS\_Sim(L, S)$, S is assigned to the cluster C with label L_C such that the $LS\_Sim(L_C, S)$ score is the *highest* among all the LS_Sim scores of S and other labels.

3.3 Ranking Sentences in Clusters

Each sentence S in the top-100 retrieved user reviews, i.e., *TopRev*, T, is assigned a *relevance score*, denoted RS, which indicates its relative significance in capturing the content of the reviews in T. To compute RS of S, we utilize the sentence *features* presented between Section 3.3.1 and Section 3.3.6.

3.3.1 Significance Factor. We rank each sentence S in a *TopRev* review using *significance factor* [4]. The *significance factor* for S relays how significant S is based on the significance of the words in S. *Significant words* are defined as words of medium frequency in the reviews, where *medium* means that the frequency is between predefined high-frequency and low-frequency cutoff values. Intuitively, higher scores are given to sentences with more *significant words*. Given that $f_{r,w}$ is the *frequency* of word w in the review r, then w is a significant word if (i) it is not a stopword, which eliminates the high-frequency, non-essential words, and (ii)

$$f_{r,w} \geq \begin{cases} 7 - 0.1 \times (25 - Z) & \text{if } Z < 25 \\ 7 & \text{if } 25 \leq Z \leq 40 \\ 7 + 0.1 \times (Z - 40) & \text{otherwise} \end{cases} \tag{6}$$

where Z is the number of sentences in r, and 25 and 40 are the low- and high-frequency cutoff values, respectively.

Once we know which words in a user review are significant, we can calculate the significance factor (SF) of a sentence S, i.e.,

$$SF(S) = \frac{|significant\text{-}words|^2}{|S|}, \tag{7}$$

where $|S|$ is the number of words in S and $|significant\text{-}words|$ is the number of significant words in S.

3.3.2 Sentiment Value. The *sentiment value* of a sentence S is determined by using SentiWordNet, a lexical resource in which a word w is associated with three numerical scores, i.e., $Obj(w)$, $Pos(w)$, and $Neg(w)$, describing how Objective (i.e., neutral), Positive, and Negative w are. We compute the *sentiment value* of S by computing the absolute value of the *sum* of $Obj(w)$, $Pos(w)$, and $Neg(w)$ of each non-stopword w in S.

3.3.3 Sentence-Label Similarity. Since a cluster label L captures the facet of a product specified in the sentences of its cluster C, we compute the SLB_Sim score that measures the *degree of resemblance* between the label L (of C) and each sentence S in C as

$$SLB\_Sim(L,S) = \frac{\sum_{i=1}^{|S|}\sum_{j=1}^{|L|} wcf(w_i, w_j)}{|S|} \tag{8}$$

where $|S|$ ($|L|$, respectively) is the number of words in S (L, respectively), w_i (w_j, respectively) is a keyword in S (L, respectively), and $wcf(w_i, w_j)$ is the word-correlation factor of w_i and w_j. As the length of S can potentially affect $SLB\_Sim(L, S)$, since the longer S is, the higher $SLB\_Sim(L, S)$ is, the accumulated word-correlation factors of $SLB\_Sim(L, S)$ is divided by the number of words in S.

3.3.4 Sentence-to-Sentence Similarity. In order to avoid choosing (very) similar sentences to be included in a summary, we prioritize sentences that are unique based on the wcf value of the words in each sentence in *TopRev* T. The *degree of similarity* of a sentence S_i with respect to the others in T indicates the relative degree of S_i in capturing the overall semantic *content* of T, denoted $Sim(S_i)$. We compute $Sim(S_i)$ using (i) the wcf of every word in S_i and words in each remaining sentence S_j in T and (ii) the *Odds ratio* = $\frac{p}{1-p}$ [12].

$$Sim(S_i) = \frac{\sum_{j=1, i\neq j}^{|S|}\sum_{k=1}^{n}\sum_{l=1}^{m} wcf(w_k, w_l)}{1 - \sum_{j=1, i\neq j}^{|S|}\sum_{k=1}^{n}\sum_{l=1}^{m} wcf(w_k,\ w_l)} \tag{9}$$

where $|S|$ is the number of sentences in T, n (m, respectively) is the number of words in S_i (S_j, respectively), which is a sentence in T, and w_k (w_l, respectively) is a word in S_i (S_j, respectively).

[10] en.wikipedia.org/wiki/Wikipedia:Database_download

[11] Words in the Wikipedia documents were *stemmed*, i.e., reduced to their grammatical roots, after the stopwords were removed which, as an effect, minimize the number of keywords to be considered.

[12] wordnet.princeton.edu

3.3.5 Sentence Length. We penalize sentences that are either *too short* (< 15 words) or *too long* (> 30 words) [19]. Short sentences are detrimental to our summarization task, since they require some introduction or do not have as much information included, whereas long sentences have a higher probability of discussing multiple topics and can be found somewhere else in a user review. We compute the *Sentence Length*, denoted SL, of a sentence S as

$$SL(S) = \begin{cases} -1 & \text{if } |S| < 15 \text{ or } |S| > 30 \\ 0 & \text{otherwise} \end{cases} \quad (10)$$

where $|S|$ is the number of (stop)words in S.

3.3.6 Named Entity. An entity can be any word or a series of words that consistently references to the same concept. It is well-known that a sentence that contains a named entity usually captures useful information in a document [18]. Named Entity Recognition (NER) focuses on (i) determining if a word w is part of a named entity and (ii) assigning w to the correct entity. In Natural Language Processing (NLP), this can be accomplished by categorizing each word w into a category, such as a person, organization, time, and location, assuming that w belongs to a name entity. To determine the *named entity weight* of a sentence S, denoted $NE(S)$, we consider the number of named entities in S. By summing the number of named entities in S, we can prioritize sentences that are more informative, i.e., with more named entities, than others.

$$NE(S) = \frac{\sum_{i=1}^{|E|} f(E_i)}{f(E)} \quad (11)$$

where $|E|$ is the number of named entities in S, $f(E_i)$ is the frequency of occurrence of entity E_i in *TopView* T, and $f(E)$ is the sum of the frequency of occurrence of all named entities in T.

3.3.7 CombMNZ. Based on the respective scores of the features discussed in Sections 3.3.1 through 3.3.6 that are computed for each sentence in a user review, we rank all the sentences in the top-100 user reviews accordingly. To compute a single score on which the cumulative effect of the six different features of each sentence are used for ranking propose, we rely on the CombMNZ model. CombMNZ is a well-established data fusion method for combining multiple ranked lists on an item I, i.e., a sentence in our case, to determine a *joint* ranking of I, a well-known rank-aggregation task or data fusion task.

$$CombMNZ_I = \sum_{c=1}^{N} I^c \times |I^c > 0|, \text{ where } I^c = \frac{S^I - I^c_{min}}{I^c_{max} - I^c_{min}} \quad (12)$$

where N is the number of ranked lists to be fused, which is *six* in our case, I^c is the normalized score of I in the ranked list c, and $|I^c > 0|$ is the number of non-zero, normalized scores of I in the lists to be fused. Prior to computing the ranking score of a sentence S, we transform the original scores in each feature ranked list of S into a *common range* $[0, 1]$ such that S^I is the score of I in the ranked list c to be normalized, I^c_{max} (I^c_{min}, respectively) is the maximum (minimum, respectively) score available in c.

3.4 Our Approach in Creating Summaries

In creating a user review summary, we include some sentences in clusters created in Section 3.2.3. To determine which sentences are to be extracted from which cluster and included in the summary, we rely on the ranked cluster labels introduced in Section 3.2.2. Using the ranked labels, we include in the summary *Sum* of a user query one sentence from a cluster at a time, starting from the cluster with the *highest-ranked label* (based on its *LRank* score defined in Equation 3), up till the limit of the summary size is reached. Note that the selection terminates whenever the length of the newly-selected sentence and other sentences that are already included in *Sum* exceeds *250* words, which is recommended by the *Text Analysis Conference (TAC)*[13] for a multi-document summary.

If the number of sentences that should be included in a summary *exceeds* the number of generated clusters after selecting the highest-ranked sentence in each cluster, then in the subsequent iterations we select the next highest-ranked sentence S' in cluster C with the *lowest similarity score*, denoted *LSS*, with respect to the sentence(s) S in C that has (have) already been included in the summary for Q. The *LSS* score of S' is computed as the *sum* of the word-correlation factors between each non-stop, stemmed word in S' and S. By considering the *LSS* score of a sentence S' in a cluster with respect to S in the summary being constructed for Q, we ensure that S and S' are *distinct* in contents, which *avoids redundancy* and *maximizes coverage* in terms of information included in the summary, a novelty of our summarization approach. Moreover, if a facet F is preceded by a negation term in Q, any cluster label that includes F is excluded from the sentence selection process.

3.5 Generating Different Types of Summaries

We create a single summary in response to the information needs specified in a user query Q. A summary is (i) *General*, if Q inquires on common feedback of a particular product P, (ii) *Sentiment-Specific*, if Q asks for positive or negative information about P, (iii) *Facet-Specific*, if Q queries on specific facets of P, or (iv) *Facet-Sentiment-Specific*, if Q looks for sentiment information on specific facets of P.

3.5.1 General Summaries. A *General* summary addresses different facets and sentiments of a product being reviewed. It consists of the highest-ranked sentences (regardless of their polarity), along with probably the ones with the lowest *LSS* scores, in (highly ranked) clusters (with the highly ranked labels determined using Equation 3) created in Section 3.2.3, which are chosen by following the procedure established in Section 3.4 to be included in the summary.

3.5.2 Sentiment-Specific Summaries. A *Sentiment-Specific* summary is created in the same manner as a *General* summary, except that only the highly-ranked sentences (along with probably the ones with the lowest *LSS* score) in clusters which satisfy the *sentiment* (i.e., positive or negative) specified in Q (identified using the keyword tagger introduced in Section 3.1) are included in the summary for Q. To determine the positive or negative polarity of a sentence S in a cluster, we calculate the overall sentiment score of S by *subtracting* the sum of its *negative* word SentiWordNet scores from the sum of its *positive* word SentiWordNet scores that reflects the

[13] nist.gov/tac

(degree of) sentiment of S such that if its sentiment score is positive (negative, respectively), then S is labeled as positive (negative, respectively). Employing this sentence-based, sentiment approach, we include in each *Sentiment-Specific* (*Facet-Sentiment-Specific*, respectively as introduced in Section 3.5.4) summary sentences reflecting the sentiment specified in the corresponding query.

3.5.3 Facet-Specific Summaries. To create the *Facet-Specific* summary for Q, we first identify the labels (from the set of labels created in Section 3.2.1) that are highly similar to each of the facets F (determined using the keyword tagger in Section 3.1) specified in Q. To identify cluster labels *highly similar* to F, we employ a reduced version of the word-similarity matrix which contains 13% of the most frequently-occurring words (based on their *frequencies of occurrence* in the Wikipedia documents), and for the remaining 87% of the less-frequently-occurring words, only exact-matched correlation factors, i.e., word correlation factors of values 1.0, are used. A label L (and its cluster) is considered highly similar to F only if the word-correlation factor of (w_1, w_2) between the non-stop, stemmed word w_1 in L and w_2 in F exists in the *reduced* matrix, which significantly minimizes the processing time to identify the desired labels without affecting the quality of the created summaries.

In creating the *Facet-Specific* summary, we follow the same procedure as detailed in Section 3.4 for selecting sentences, regardless of their polarity, to be included in the summary. Instead of considering ranked labels, we rely on the *ranking* of the *highly similar* labels with respect to F computed using the reduced word-correlation matrix. Moreover, the content of the *Facet-Specific* summary of Q is uniformly divided among each of the facets specified in Q, i.e., the number of sentences to be included in the summary is uniformly extracted from the sentence clusters with labels highly-similar to or the same as each facet specified in Q.

3.5.4 Facet-Sentiment-Specific Summaries. The *Facet-Sentiment-Specific* summary is created in response to Q by including solely the sentences in clusters which (i) reflect the polarity specified in Q (as detailed in Section 3.5.2) and (ii) belong to the clusters with labels highly similar to or the same as a facet in Q (as in a *Facet-Specific* summary). The process of including sentences in a *Feature-Sentiment-Specific* summary is the same as the one detailed in section 3.5.3 for *Feature-Specific* summaries, except that only sentences with the same polarity as the one specified in Q are included in the summary.

4 EXPERIMENTAL RESULTS

To assess the performance of our multi-document sentiment summarization approach, we first present the datasets used for the empirical study (in Section 4.1) and define the evaluation metrics used for analyzing the performance of our summarizer (in Section 4.2). Hereafter, we introduce the *criteria* for evaluating the *quality* of generated summaries using different measures (in Section 4.3). Furthermore, we compared summaries created by our summarizer and the well-known TAC-08 summaries (in Section 4.4).

4.1 Datasets

We rely on the benchmark dataset set up for the Opinion Summarization Pilot task of the 2008 Text Analysis Conference, denoted TAC-08[14], a dataset that includes a set of 87 (squishy-list) questions, to assess the effectiveness of our sentiment summarization approach in (i) identifying information needs specified in a user's query Q, (ii) creating summaries that satisfy the information specified in Q, and (iii) generating a high-quality summary of multiple documents on a topic. For each question Q in TAC-08, which is treated as a query, there is (i) a set of documents D (extracted from the TREC Blog06 collection[15]) that serves as the source for creating a summary (of D) and (ii) a list of *expert-created* sentences/phrases that are expected to be included in a summary (of D) with respect to Q. Since the goal of the Opinion Summarization task is to evaluate multi-document sentiment summaries created in response to a question, TAC-08 is ideal for evaluating the effectiveness of our sentiment summarization approach, including keyword tagging.

4.2 Performance Evaluation

We define in this section the various evaluation metrics used for assessing our summarization approach in (i) identifying the information needs expressed in a query Q (in Section 4.2.1), (ii) generating a summary that satisfies Q (in Section 4.2.2), and (iii) creating a high-quality multi-document summary for Q (in Section 4.3).

4.2.1 Accuracy on Keyword Tagging. To assess the performance of the multi-class SVM adopted by our sentiment summarization approach for *identifying* the types of keywords expressed in users' queries, as well as keywords in the user reviews, we compute the *accuracy* ratio, which is defined as the proportion of the number of keywords *correctly identified* by the multi-class SVM as *products, facets, sentiment keywords, or non-essential terms* over the total number of keywords used for evaluating the SVM, i.e., the number of keywords in the 87 queries in TAC-08 and the TREC Blog06 collection, which are treated as user reviews for our empirical study.

To create the multi-class SVM (introduced in Section 3.1) for identifying products, facets, sentiment keywords, and non-essential terms in users' queries and reviews, we constructed a dataset, denoted *SVM-Data*, which consists of approximately 32,000 keywords extracted from 300 (opinion) queries and 2,800 responses, in addition to the TAC-08 and TREC Bolg06, which were employed as user queries and reviews, retrieved from WikiAnswers and Yahoo! Answers. To validate the effectiveness of the multi-class SVM[16], we adopted the 10-fold cross-validation approach so that in each of the 10 repetitions, 90% of the instances in *SVM-Data* were used for classification and the remaining 10% for validation purpose.

The *accuracy* of our sentiment summarization approach in identifying the types of keywords specified in user queries and keywords in reviews was computed using the trained multi-class SVM on each keyword in each of the 87 queries in TAC-08 and the user reviews in the TREC Blog06 collection. As shown in Figure 1, the

[14] www.nist.gov/tac/data/index.html
[15] TREC Blog06 is a collection of blog posts downloaded from the Web between December 2005 and February 2006.
[16] Keywords in *SVM-Data* were previously labeled by independent assessors as *Products*, *Facets*, *Sentiment Keywords*, or *Non-Essential Terms*.

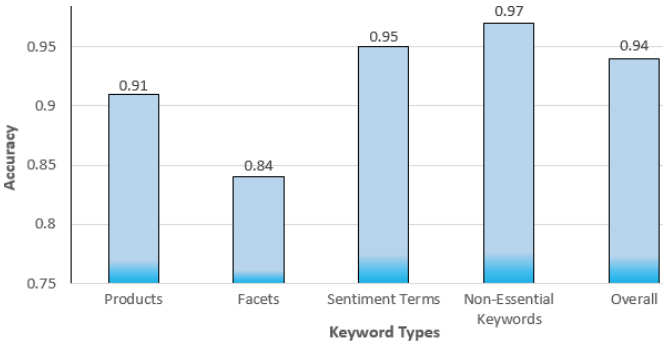

Figure 1: (Overall) Accuracy of our multi-class SVM for identifying users' information needs specified in TAC-08 query set and keywords in Blog06 collection and SVM-Data

(overall) accuracy of the multi-class SVM on identifying the keywords in TAC-08 queries, Blog06 posts and SVM-Data is 94%. The accuracy ratios achieved on correctly detecting products, facets, sentiment keywords, and non-essential terms, respectively, are also shown in Figure 1. Note that the low accuracy ratio of facets as shown in the figure, in comparing with products, sentiment keywords, and non-essential terms, is due to the fact that facets are more difficult to classify than the rest because of their variations; however, the accuracy ratio for identifying facets is still in the mid-80% range, which is a high percentage.

4.2.2 Nugget Pyramid Score. To verify whether our generated summaries *satisfy the information needs specified in user queries*, we rely on the *Nugget-Pyramid* score [13]. Consider a test query Q in TAC-08, "How good is a town house in Brooklyn?" A *human assessor* creates a list of relevant *nuggets*, which are expert-created phrases/sentences, e.g., "Brooklyn is livable," and "As a current owner of a town house in Brooklyn, I feel good about its safety", that address different aspects of Q. It is expected that a "good" summary includes (the majority of the) nuggets in the corresponding list of *relevant nuggets*. We evaluate a summary S generated by our approach in response to Q by verifying the (non-)existence of a *conceptual match* between each provided nugget and S, which is a match independent of the distinct wording used in S and the nugget.

The *Nugget-Pyramid* score of S [13], which is the *weighted harmonic mean* between (*nugget*) *precision* and (*nugget*) *recall* that favors recall (which is controlled by a parameter β that is set to 3 based on our empirical study), is calculated as

$$Nugget\_Pyramid(S) = \frac{(\beta^2+1) \times Precision \times Recall}{\beta^2 \times Precision + Recall} \tag{13}$$

$$Precision = \begin{cases} 1 & \text{if } Length < Allowance \\ 1 - \frac{Length - Allowance}{Length} & \text{otherwise} \end{cases} \tag{14}$$

where $Allowance = 100 \times the\_number\_of\_nuggets\_included\_in\_S$ and $Length$ is the total number of non-white-space characters in S.

$$Recall = \frac{\sum_{m \in A} w_m}{\sum_{n \in V} w_n} \tag{15}$$

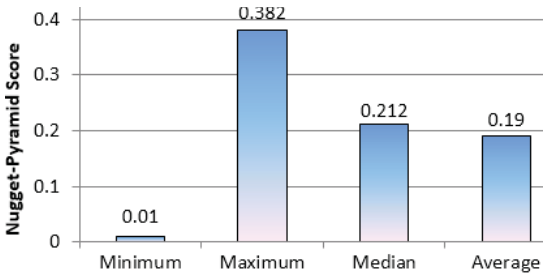

Figure 2: The *Nugget-Pyramid* scores achieved by our summarizer using TAC-08

where A is the set of (relevant) reference nuggets that are included in S, V is the set of all reference nuggets (as determined in TAC-08), and w_m (w_n, respectively) is the score (between 0 and 1, inclusively) of nugget m (n, respectively), which is determined by (*human*) *assessors*.

We have validated our approach in creating sentiment summaries that satisfy the information needs expressed in user queries using the Nugget-Pyramid score. Figure 2 shows the *minimum*, *maximum*, *median*, and *average* Nugget-Pyramid scores of our summarization approach based on TAC-08. In comparing the (average) Nugget-Pyramid scores achieved by 19[17] query-based multi-document summarizers of TAC-08, our summarization approach ranks *fourth*. Note that 36 multi-document summarizers originally took part at the 2008 TAC. However, 17 of the summarizers relied on *snippets*[18] of information provided by TAC in creating the summaries. To perform unbiased comparisons, we only consider the 19 summarizers that operate in a manner similar to our approach, i.e., they do not rely on external information, such as snippets, in creating a summary. In addition, we have evaluated 22 summaries, as opposed to the 87 summaries created using the questions and documents in TAC-08. The choice follows the evaluation premises defined by TAC, which provides assessment, in terms of using the computed Nugget-Pyramid scores, for only 22 summaries generated by each of the 19 automatic multi-document summarizers. The three summarizers that are ranked higher than our summarizer have achieved a Nugget-Pyramid score of 0.251, 0.223, and 0.201, respectively, which are close to the average score, 0.190, of our summarizer. This results have verified the effectiveness of our summarizer in generating summaries that satisfy users' information needs.

4.3 Quality Measures of Summaries

To evaluate the quality of the multi-document summaries created by our summarizer, we apply the *quality measures* defined by TAC-08 that assess the quality of summaries generated by summarizers participated at the 2008 Text Analysis Conference[19]. These quality measures, which are in the range of 1 to 10 (with 1 being the worse and 10 the best), reflect the overall quality of a generated summary

[17] The Nugget-Pyramid score for the 19 summarizers are available at The TAC-08 site www.nist.gov/tac/protected/past-blog06/2008/QA2008_runs.tar.gz.
[18] Snippets are answers to queries in TAC-08 generated by existing question-answering systems.
[19] www.nist.gov/tac/data/past-blog06/2008/OpSummQA08.html#OpSumm

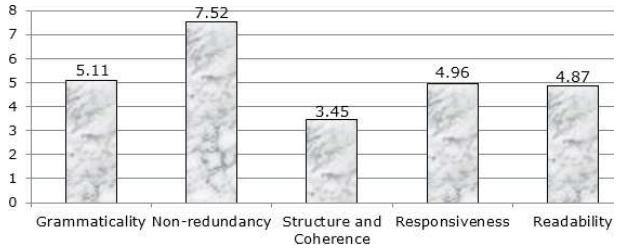

Figure 3: *Quality measures* of generated summaries

in terms of its *grammaticality*, *non-redundancy*, *structure and coherency*, *responsiveness*, and *readability*. Each of the five different measures is computed for each multi-document summary created by any summarizer to be evaluated.

- *Grammaticality*: A (high-quality) summary should not exhibit any *system-internal formatting* (e.g., html formatting tags), *capitalization errors*, or *grammatical mistakes* in sentences (e.g., fragments and missing components) that cause the text to be difficult to read.
- *Non-redundancy*: A summary should avoid unnecessary repetition, such as complete sentences that are repeatedly shown in a text, replicated facts, and repeated use of noun phrases.
- *Structure and Coherence*: A summary should be well-structured and organized, which should not be a heap of related information, but constructed from sentences that yield a *coherent* body of information on a specific topic. It should be easy to identify to whom or what each pronoun/noun phrase refers. If a product is mentioned, its role should be clear. A reference is *unclear* if a product is referred but its identity or relation to the remaining content is *unknown*.
- *Responsiveness*: A summary S on a topic C should include the information required to answer certain pre-defined questions on C[20]. The *responsiveness* score of S on C reflects the overall quality/usefulness of S in terms of satisfying the information needs expressed in the pre-defined questions on C.
- *Readability*: A summary should be *easy to read* and its content should be *easy to understand*. The *readability* score of a summary S is a score determined by the grammaticality, non-redundancy, structure, coherence, and referential clarity of the text in S.

4.4 Evaluating the Quality of Summaries Generated by Our Summarizer

We assess the quality of summaries generated by our summarizer using each of the five quality measures listed in Section 4.3. Following the evaluation methodology established by TAC[21], we relied on the automated appraiser of TAC-08 that evaluated the *grammaticality*, *non-redundancy*, *structure and coherency*, *responsiveness*, and *readability* of the summaries created by our summarizer using

[20] A set of pre-defined questions on each specified topic is provided as part of the TAC-08 dataset.

[21] www.nist.gov/tac/ protected/past-blog06/2008/README.QA.2008

1. Honda Odyssey touring
2. Minivan car-like driving
3. Transmission problem
4. Pax Tire System
5. Gas mileage
6. Usable Space
7. Easier Entry Exit
8. Driver Armrest
9. Combined City Highway
10. Stone Dead battery

> We recently bought an Odyssey EX '07 which was replacing a 2004 Tahoe and we're so happy with our new purchase. It takes the dealer about 4-5 hours to repair the tires purely outrageous. We have not been as happy with the 2007 as we were with the 2002. We took delivery of our 07 Odyssey only to be greeted by a dead battery the next day. I am sick to my stomach but thankful for the Certified warranty. No complains yet except for the poor gas mileage but I was told odyssey improves with more miles. …

Figure 4: The 10 different cluster labels and a portion of the *Facet-Sentiment-Specific* summary created for the query "The pros and cons of 2007 Honda Odyssey tire and battery"

TAC-08 dataset (as introduced in Section 4.1). The quality scores of summaries generated by our summarizer are shown in Figure 3.

To establish a baseline measure on the *quality scores* achieved by our summaries, we compare the performance, based on the quality scores, of the 19 automatic multi-document summarizers with our summarizer using the 22 summaries created by TAC-08. As shown in Table 2, our summarizer is *significantly outperformed* (with 95% confidence) by at most five (out of 19) summarizers in terms of creating summaries that are either higher in grammaticality, non-redundancy, structure and coherence, responsiveness, or readability. However, there is *not* a single system examined that achieves higher scores than our summarizer in all of the five quality measures. Our summarizer achieves the highest *non-redundant* score among all the 19 summarizers and is only outperformed by one of the 19 summarizers in creating summaries that are *responsiveness*.

The summarization systems that outperform our summarizer on either the *grammar*, *structure* and *coherence*, or *readability* quality measures employ natural language processing techniques to touch up the summaries shown to users as the final products. Our summarization approach, on the other hand, is purely *extractive*, i.e., it solely extracts sentences in the original document(s) without refinement to create a summary. The summarizer which achieves a better *responsiveness* quality measure than ours relies on word order and part-of-speech tags to determine usefulness of sentences, neither of which is employed by our summarizer for simplicity.

Figure 4 depicts the ten cluster labels and the *Facet-Sentiment-Specific* summary created by using our summarizer for the query, "The pros and cons of 2007 Honda Odyssey tire and battery." The summary achieves very high *non-redundant* and *responsiveness* ratings, in addition to the high *grammar*, *structure* and *coherence*, and *readability* quality scores according to the quality measures.

5 CONCLUSION

To facilitate the task of synthesizing opinions expressed in user reviews on a particular product specified in a user query/question, we have proposed a multi-document sentiment summarization system that is unique in terms of its design. Most prominently, our design has an effective heuristic-based sentence selection process which *retains* sentiment polarity and essential facets in the resultant summary while minimizes *redundancy* and maximizes the *coverage of information* from various information sources. The relatively straightforward approach of our proposed summarizer enhances the current design on summarization with its *effectiveness*

Table 2: Comparisons between 19 summarizers participated at TAC-08 where numbers in parentheses are System IDs

Our Summarizer	Out-Performed by	Outperforms	Significantly Out-Performed by	Significantly Outperforms
Grammaticality	5 (9, 10, 22, 23, 35)	14	4 (9, 10, 23, 35)	15
Non-redundancy	0	19	0	19
Structure and Coherence	6 (10, 13, 20, 22, 30, 35)	13	5 (10, 13, 22, 30, 35)	14
Readability	3 (8, 10, 23)	10	2 (8, 10)	17
Responsiveness	2 (9, 27)	13	1 (9)	18

and *simplicity*. The effectiveness allows its users to extract desired information without missing essential ideas captured in user reviews. The simplicity of our summarization approach (i) avoids the application of complex natural language processing and machine learning algorithms that require comprehensive training for performing the summarization task, and (ii) can be generalized to any documents, including news articles. Conducted empirical study on TAC-08 also verifies that our summarizer ranks near the top among the state-of-the-art approaches in generating query-based, high-quality, sentiment summaries that satisfy the information needs specified in users' queries.

REFERENCES

[1] A. Aker, M. Paramita, E. Kurtic, A. Funk, E. Barker, M. Hepple, and R. Gaizauskas. 2016. *Automatic Label Generation for News Comment Clusters.* Association for Computational Linguistics.
[2] S. Bahrainian and A. Dengel. 2013. Sentiment Analysis and Summarization of Twitter Data. In *IEEE CSE*. 227–234.
[3] S. Chevalier. 2021. *U.S. Online Shopper Online Reviews Concumption by Product Type.* Consumer Research Report.
[4] W. Croft, D. Metzler, and T. Strohman. 2010. *Search Engines: Information Retrieval in Practice.* Addison Wesley.
[5] J. Devlin, M. Chang, K. Lee, and K. Toutanova. 2018. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.* arXiv:1810.04805.
[6] D. Dunlavy, D. O'Leary, J. Conroy, and J. Schlesinger. 2007. QCS: A System for Querying, Clustering and Summarizing Documents. *IPM* 43, 6 (2007), 1588–1605.
[7] U. Farooq, H. Mansoor, A. Nongaillard, Y. Ouzrout, and M. Qadir. 2016. Negation Handling in Sentiment Analysis at Sentence Level. *Computer* 12, 5 (2016), 470–478.
[8] K. Ganesan and C. Zhai. 2012. Opinion-Based Entity Ranking. *Information Retrieval* 15 (2012), 116–150.
[9] J. He, P. Duboue, and J. Nie. 2012. Bridging the Gap between Intrinsic and Perceived Relevance in Snippet Generation. In *COLING*. 1129–1146.
[10] Y. Huang, Z. Liu, and Y. Chen. 2008. Query Biased Snippet Generation in XML Search. In *ACM SIGMOD*. 315–326.
[11] H. Jeong, Y. Ko, and J. Seo. 2015. Efficient Keyword Extraction and Text Summarization For Reading Articles on Smart Phone. *Computing & Informatics* 34, 4 (2015), 779–794.
[12] P. Judea. 1988. *Probabilistic Reasoning in the Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann.
[13] J. Lin and D. Demner-Fushman. 2006. Will Pyramids Built of Nuggets Topple Over?. In *HLT/NAACL*. 383–390.
[14] Y. Liu and Y. Zheng. 2005. One-Against-All Multi-Class SVM Classification Using Reliability Measures. In *IJCNN*. 849–854.
[15] G. Luger. 2009. *Artificial Intelligence: Structures & Strategies for Complex Problem Solving, 6^{th} Ed.* Addison Wesley.
[16] R. Nallapati, F. Zhai, and B. Zhou. 2017. SummaRuNNer: A Recurrent Neural Network Based Sequence Model for Extractive Summarization of Documents. In *AAAI*. 3075–3081.
[17] J. Ni, J. Li, and J. McAuley. 2019. Justifying Recommendations Using Distantly-Labeled Reviews and Fine-Grained Aspects. In *EMNLP-IJCNLP*. 188–197.
[18] S. Osinski. 2006. Improving Quality of Search Results Clustering with Approximate Matrix Factorisations. In *ECIR*. 167–178.
[19] B. Schiffman, A. Nenkova, and K. McKeown. 2002. Experiments in Multidocument Summarization. In *HLT*. 52–58.
[20] S. Trauzettel-Klosinski and K. Dietz. 2012. Standardized Assessment of Reading Performance: The New International Reading Speed Texts IreST. *Investigative Ophthalmology & Visual Science* 53 (2012), 5452–5461.
[21] M. Woolfe. 2014. *A Statistical Analysis of 1.2 Million Amazon Reviews.* Minimaxir.com.