Query-based summarization using MDL principle

Natalia Vanetik
Shamoon College of Engineering
Beer Sheva
Israel
natalyav@sce.ac.il

Marina Litvak
Shamoon College of Engineering
Beer Sheva
Israel
marinal@sce.ac.il

Abstract

Query-based text summarization is aimed at extracting essential information that answers the query from original text. The answer is presented in a minimal, often predefined, number of words. In this paper we introduce a new unsupervised approach for query-based extractive summarization, based on the minimum description length (MDL) principle that employs Krimp compression algorithm (Vreeken et al., 2011). The key idea of our approach is to select frequent word sets related to a given query that compress document sentences better and therefore describe the document better. A summary is extracted by selecting sentences that best cover query-related frequent word sets. The approach is evaluated based on the DUC 2005 and DUC 2006 datasets which are specifically designed for query-based summarization (DUC, 2005 2006). It competes with the best results.

1 Introduction

Query-based summarization (QS) is directed toward generating a summary most relevant to a given query. It can relate to a single-document or to a multi-document input. Our approach for QS is based on the MDL principle, defining the best summary as the one that leads to the *best compression* of the text *with query-related information* by providing its *shortest and most concise description*. The MDL principle is widely useful in compression techniques of non-textual data, such as summarization of query results for online analytical processing (OLAP) applications (Lakshmanan et al., 2002; Bu et al., 2005). However, only a few works about text summarization using MDL can be found in the literature. Nomoto and Matsumoto (2001) used K-means clustering extended with the MDL principle, to find diverse topics in the summarized text. Nomoto (2004) also extended the C4.5 classifier with MDL for learning rhetorical relations. In (Nguyen et al., 2015) the problem of micro-review summarization is formulated within the MDL framework, where the authors view the tips as being encoded by snippets, and seek to find a collection of snippets that produces the encoding with the minimum number of bits.

This work proposes a MDL approach where the sentences that are best described by the query-related word sequences are selected to a summary. It is principally different from the mentioned works by (1) using frequent itemsets and not single words in the description model, (2) compressing entire documents instead of summaries, (3) ranking method for sentences, and (4) the description model itself. We tested our approach on DUC 2005 and DUC 2006 data for English query-based summarization.

2 Related Work

Multiple works about QS have been published in recent years. Daumé III and Marcu (2006) presented BayeSum, a model for sentence extraction in QS. BayeSum is based on the concepts of three models: language model, Bayesian statistical model, and graphical model. Mohamed and Rajasekaran (2006) proposed an approach for QS based on document graphs, which are directed graphs of concepts or entity nodes and relations between them. The work in (Bosma, 2005) introduced a graph search algorithm that looks for relevant sentences in the discourse structure represented as a graph. The author used Rhetorical Structure Theory for creating a graph representation of a text document - a weighted graph with nodes standing for sentences and weighted edges representing a distance between sentences. Conroy et al. (2005) presented the CLASSY summarizer that used a hidden Markov model based on signature terms and query terms for sentence selection within a document, and a pivoted question answering algorithm for redundancy removal. Liu et al. (2012) proposed QS with multi-document input using unsupervised deep learning. Schilder and Kondadadi (2008) presented FastSum - a fast query-based multi-document summarizer based solely on word-frequency features of clusters, documents, and topics, where summary sentences are ranked by a regression support vector machine. Tang et al. (2009) proposed two strategies to incorporate the query information into a probabilistic model. Park et al. (2006) introduced a method that uses non-negative matrix factorization to extract query-relevant sentences. Some works deal with domain-specific data (Chen and Verma, 2006) and use domain-specific terms when measuring the distance between sentences and a query. Zhou et al. (2006) describes a query-based multi-document summarizer based on basic elements, a head-modifier-relation triple representation of document content. Recently, many works integrate topic modeling into their summarization models. For example, Li and Li (2014) extend the standard graph ranking algorithm by proposing a two-layer (sentence layer and topic layer) graph-based semi-supervised learning approach based on topic modeling techniques. Wang et al. (2014) present a submodular function-based framework for query-focused opinion summarization. Within their framework, relevance ordering produced by a statistical ranker, and information coverage with respect to topic distribution and diverse viewpoints are both encoded as submodular functions. Some works (Li et al., 2015) use external resources with the goal to better represent the importance of a text unit and its semantic similarity with the given query. Otterbacher et al. (2009) present Biased LexRank method, which represents a text as a graph of passages linked based on their pairwise lexical similarity, identifies passages that are likely to be relevant to a users natural language question and then perform a random walk on the lexical similarity graph in order to recursively retrieve additional passages that are similar relevant passages. Williams et al. (2014) provides a task-based evaluation of multiple query biased summarization methods for cross-language information retrieval using relevance prediction. In (Litvak et al., 2015) we applied the MDL principle to generic summarization, where we considered frequent word sets as the means for encoding text. The results demonstrated superiority of the proposed method over other methods on DUC data. This paper continues the above work by constructing a model where frequent word sets depend on the query.

3 Methodology

3.1 Overview

Our approach consists of the following steps: (1) text preprocessing, (2) query-related frequent itemset mining, (3) finding the best MDL model, and (4) sentence ranking for the summary construction. The general scheme of our approach is depicted in Figure 1. Details of every step are given in sections below. Section 3.8 contains an example of intermediate and the final (the summary) outputs for one of the document clusters from DUC 2005 dataset.

3.2 Query-based MDL principle

The Minimum Description Length (MDL) Principle is based on the idea that a regularity in the data can be used to compress the data, and this compression should use fewer symbols than the data itself. Intuitively, a *model* is a partial function from data subsets to codes, where the codes size has logarithmic growth.

In general, given a set of models $\mathcal{M}$, a model $M \in \mathcal{M}$ is considered the *best* if it minimizes $L(M) + L(D|M)$, where $L(M)$ is the bit length of description of M and $L(D|M)$ is the bit length of the dataset D encoded with M. As such, the frequency and the length of the codes that replace data subsets are the most important features of the MDL model. Because we aim at query-based summarization, in our approach we seek a query-dependent model M_Q that minimizes $L(M_Q) + L(D|M_Q)$; it may not be the best model overall but it has to be the best among models for the query Q. Note that the MDL approach does not use the actual codes but only takes into account their bit size.

3.3 Query-based data setup

In our case both text and query undergo preprocessing that includes sentence splitting, tokeniza-

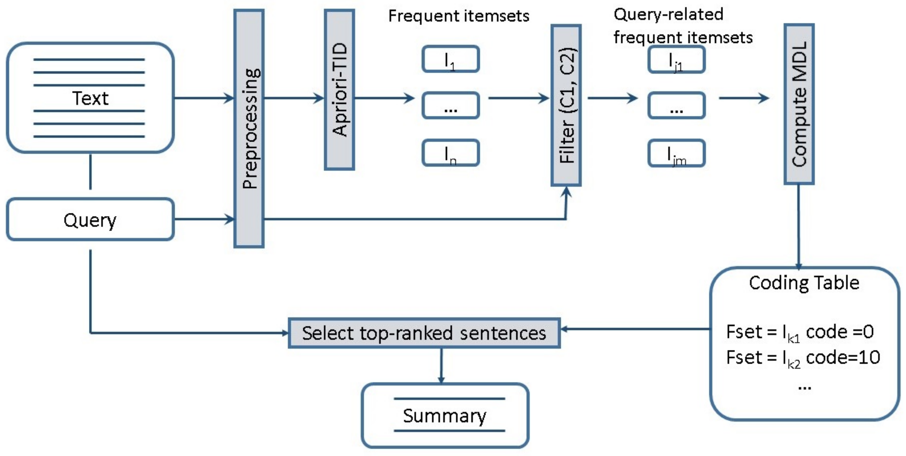

Figure 1: Query-based MDL summarization

tion, stemming, and stop-words removal. Additionally, sentences that are too long (over 40 words), too short (less than 5 words), or consist primarily of direct speech (main portion of a sentence is contained within quotes), are omitted. The number of these sentences is small and we noted that their inclusion in summaries, although rare, decreases summary quality. No deep linguistic analysis is performed, and therefore this method is suitable for any language with basic tools.

A query is considered to be a set of stemmed tokens, e.g., terms, even if it contains more than one sentence. A document or a document set (in case of multi-document summarization) D is treated as a dataset where each sentence is a *transaction* that is a set of stemmed tokens. The order of words in a sentence is ignored in our model, because we consider the relation of a sentence to a query to be more important than the order in which a sentence utilizes query tokens. Formally, we have sentences $S_1, \ldots, S_n$ of a document set where every sentence is a subset of unique terms (stemmed tokens), denoted by $T_1, \ldots, T_m$. A query Q is a subset of unique terms as well.

3.4 Frequent itemsets

In our approach, we refer to text as a transactional dataset, where each sentence is considered to be a single *transaction* consisting of *items*. An item in our case is a term, i.e. stemmed word. Therefore, a sentence is viewed as a set of terms contained in it. A set of items, called an *itemset*, is *frequent* if it is contained (as a set) in S sentences, where $S \geq 1$ is user-defined parameter.

The paper (Agrawal and Srikant, 1994) has proposed two algorithms–Apriori and Apriori-TID–for processing large databases and mining frequent itemsets in efficient time. The Apriori algorithm makes multiple passes over the database while Apriori-TID algorithm uses the database only once, in the first pass. In this work, we use the Apriori-TID algorithm for frequent itemset mining. While multitude of algorithms performing the same task exist, Apriori-TID is sufficient for our purposes because texts, treated as transactional datasets, are rarely dense, and therefore the number of frequent itemsets found in texts is usually not very large.

3.5 Data encoding

In general MDL approach, a *Coding Table* is a collection CT of sets from D that are used as a model of our dataset. According to the MDL principle, CT is considered to be the *best* when it minimizes encoded dataset size $size(D, CT) = L(CT) + L(D|CT)$. In general MDL approach, a *Coding Table* is a collection CT of sets from D that are used as a model of our dataset. According to the MDL principle, CT is considered to be the *best* when it minimizes encoded dataset size $size(D, CT) = L(CT) + L(D|CT)$.

In our approach, sets included in the Coding Table come from the set F of all frequent word sets in our text. Moreover, we only keep a frequent set in F if it is query-related, and therefore all the sets in CT are query-related as well. Every member $I \in CT$ is associated with its $code(I)$ of logarithmic growth (for instance, prefix codes may be used). In our case, the choice of a specific code is not important as we only use its size $|code(I)|$ when computing $size(D, CT)$. We use Huffman Coding in the current version, where $|code(I)| = \log |I|$. General approach of using frequent sets for dataset MDL representation first appeared in (Vreeken et al., 2011); here, we apply it to text and only care about word sets related to a query.

There are two main orders to consider: Standard Candidate Order in which F is kept, whose purpose is to build the coding table faster. Itemsets in F are first sorted by increasing support, then by decreasing sequence length, then lexicographically. The Standard Cover Order of CT keeps its members sorted by first by decreasing sequence length, then by decreasing support, and finally, in lexicographical order. Using this order ensures that encoding of the dataset with CT indeed produces minimal encoding length $L(D|CT)$ for fixed CT. Validity of this approach is proven in (Vreeken et al., 2011).

3.6 Sentence ranking and summary construction

We are interested in the dataset $D|CT$ after it is compressed in the best possible way with the best compressing query-related set CT. The dataset $D|CT$ is obtained by replacing in D every itemset in CT by its code, and shorter codes are selected first. We use an upper bound on the size of CT, in order to limit document compression, and select it to be *equal to the target summary size*, which is denoted by *SummarySize*. The ideal compression in this case will compress only words most relevant to the summary and will ignore everything else; additionally, this limitation speeds up computation.

The summary is constructed by iteratively selecting the sentences according to the coverage of CT. At each step, a sentence that covers the most important uncovered itemset in CT is added to a summary. Importance of itemsets in CT is determined by their order – higher itemsets in CT (those with shorter codes) have higher importance.

3.7 Query-based frequent itemsets and data encoding

In order to direct the summarization process towards the given query, we compute and use for encoding only frequent itemsets that are related to the given query. We tested two different types of constraints on frequent itemsets:

- **(C1)** All terms in a frequent itemset I must be contained in the query: $I \subseteq Q$.
  With this approach, a set of words in a sentence is encoded only if these words appear in the query.

- **(C2)** Every frequent itemset and the query must have a common term: $I \cap Q \neq \emptyset$.
  Here, a set of words in a sentence is encoded if at least one word in the set appears in the query.

Both methods ensure that only terms related to the query are taken into account. Therefore, instead of all frequent itemsets F, we only use its subset $F_{Ci}, i = 1, 2$. The general Standard Candidate Order is used, and members of F_{Ci} are sorted by by increasing support, then by decreasing sequence length, then lexicographically.

Because the coding table CT can now contain members of F_{Ci} only, we modify the Standard Cover Order of CT accordingly in order to compress query-related terms first. The order is modified as follows:

- **(C1)** Because every member of CT contains *only* terms used in the query, we sort it first by decreasing sequence length, then by increasing support, and then lexicographically. Here, sequence length is precisely the number of query terms contained in a frequent itemset, and itemsets containing more query terms get higher priority.

- **(C2)** Every member of CT has *some* common terms with the query, we sort CT first by the number of terms common to an itemset and the query, then by decreasing sequence length, then by increasing support, and then lexicographically. Here, a precedence is given to itemsets that have more in common to the query. Note that we tested other measures for distance between itemsets and the query (Jaccard similarity, cosine similarity), but this method provided better results.

Detailed description of our query-based summarization method Qump (Query-based Krimp) is given in Algorithm 1.

3.8 Example

Here we demonstrate intermediate and the final (summary) outputs for the document cluster

Algorithm 1: Qump: Query-Based Krimp

```
Input:
    (1) a document or a document set $D$,
        preprocessed as in Section 3.3,
    (2) a query $Q$ preprocessed
        as in Section 3.3,
    (3) target summary word limit SummarySize,
    (4) support bound S,
    (5) constraint Cx on frequent itemsets
        as described in Section 3.7.
Output: Extractive summary $Summary$
/* STEP 1: Query-related frequent set mining */
(1a) $F \leftarrow$ frequent sets of terms from $\{T_1, \ldots, T_m\}$
  appearing in at least Supp fraction of sentences that
  satisfy constraint Cx;
(1b) Sort $F$ according to Standard Candidate Order;
/* STEP 2: Initialize the coding table */
(2a) Add all terms $T_1, \ldots, T_m$ and their support to
  $CT$;
(2b) Keep $CT$ always sorted according to Standard
  Cover Order;
(2c) Initialize prefix codes according to the order of
  sets in $CT$;
/* STEP 3: Find the best encoding */
(3a) $EncodedData \leftarrow$
  $PrefixEncoding(\{S_1, \ldots, S_n\}, CT)$;
(3b) $CodeCount \leftarrow 0$;
while $CodeCount < SummarySize$ and $F \neq \emptyset$ do
    $BestCode \leftarrow \arg\min_{c \in F} L(CT \cup \{c\}) +$
      $L(PrefixEncoding(\{S_1, \ldots, S_n\}, CT \cup \{c\}))$;
    $CT \leftarrow CT \cup \{BestCode\}$;
    $F \leftarrow F \setminus \{BestCode\}$;
    /* If a code is used, its supercodes cannot
    appear in the data */
    $F \leftarrow F \setminus \{d \in F | BestCode \subset d\}$;
    $CodeCount$++;
end
/* STEP 4: Build the summary */
$Summary \leftarrow \emptyset$;
for codes $c \in CT$ do
    $importance(c)$ := serial number of $c$ in $CT$
end
while $|Summary| < SummarySize$ do
    for all unselected sentences $S$ do
        $nCov(S) \leftarrow \sum_{c \in CT} importance(c)/|S|$
    end
    $S \leftarrow \arg\max_S nCov(S)$;
    $Summary \leftarrow Summary \cup \{S\}$;
    $CT := CT \setminus \{c\}$
    for $d \subset c$ do
        $CT := CT \setminus \{d\}$
    end
end
return $Summary$
```

D301I and the query *"International Organized Crime Identify and describe types of organized crime that crosses borders or involves more than one country. Name the countries involved. Also identify the perpetrators involved with each type of crime, including both individuals and organizations if possible."* from the DUC 2005 dataset.

- The coding table CT (only its top 8 itemsets with the shortest codes, sorted from the most important to the least) is given in Table 1.

Code #	Itemset
0	cross border
1	countri includ
2	crime countri
3	border cross
4	countri identifi
5	drug
6	cocain
7	offici
...	...

Table 1: CT example, top records.

- The sentence *"The drugs organisation used intricate methods - including bank accounts, couriers and ships as well as dummy and real companies in many countries - to smuggle cocaine from South America to Europe."* after encoding (replacement of phrases by codes) looks like this: *"code#5 code#24 intric method includ code#19 account courier ship dummi real compani mani code#16 code#23 code#6 south america code#40"*, and it covers 7 out of 50 codes from the CT. Normalized by the sentence length, its coverage is the largest among all sentences, and therefore it is selected to the summary.

- The summary contains 8 following sentences with the highest CT coverage, ordered by their appearance in the summarized documents:
"The drugs organization used intricate methods - including bank accounts, couriers and ships as well as dummy and real companies in many countries - to smuggle cocaine from South America to Europe. But this week the New York Times gave extensive coverage to a report from a US intelligence officer that warned Mexican drug-traffickers were planning to take advantage of lax border controls. The measures announced yesterday include a CDollars 5 tax cut per carton of cigarettes, bringing federal taxes down to CDollars 11 a carton. Mr Louis Freeh, the FBI director who arrived in Moscow yesterday as part of a central and East European tour, said the mounting crime wave in Russia posed 'common threats' to all. Crime Without Frontiers

is the story of how western and eastern criminal syndicates secured the former Soviet safehouse, the last piece in constructing a global pax mafiosa. In the pax mafiosa, business is business - the Chinese Triads are partners in crime with the American Mafia; the Italians use the Russians to launder for the Colombian cartels, the Japanese Yakuza work hand in hand with the Italians. Last January, after the ouster of Panamanian strongman Manuel A. Noriega, the new government of Panama agreed to U.S. requests for records of bank accounts identified as having been used by cartel money launderers. Cuba's interior minister, the Cabinet officer in charge of domestic law enforcement, was fired Thursday as that nation's drug purge continued, but the crackdown has failed to touch other leaders who U.S. officials say are involved in trafficking."

4 Experiments

4.1 Dataset

We selected two English corpora of the Document Understanding Conference (DUC): DUC 2006 and DUC 2005 (DUC, 2005 2006) for our experiments, which are standard datasets used for query-based summarization methods evaluation.

The DUC 2005 dataset contains 50 documents sets of 25-50 related documents each. Average number of words in a document set is 20185. For every document set a short (1-3 sentences) query is supplied. From 4 to 9 gold standard summaries are supplied for every document set, and the target summary size is 250 words.

The DUC 2006 dataset contains 50 documents sets of 25 related document each. Average number of words in a document set is 15293. For every document set a short (1-3 sentences) query is supplied. Four gold standard summaries are supplied for every document set, and the target summary size is 250 words.

4.2 Experiment setup

We generated summaries for each set of related documents (by considering each set of documents as one meta-document) in the DUC 2005 and DUC 2006 corpora. The summarization quality was measured by the ROUGE-1 (Lin, 2004) recall scores[1] , with the word limit set to 250, without stemming and stopword removal. We limited the size of the coding table by 250, as described in Section 3.4, and set support count $\mathbf{S} = 2$ in order to take into account all terms repeated twice or more in the text.

4.3 Results

Table 2 shows the ROUGE-1 scores of our algorithm comparative to the scores of 32 systems that participated in the DUC 2005 competition. Two options of our algorithm that correspond to constraints described in Section 3.7 appear in the "Systems" column of Table 2, denoted by Qump(C1-C2). Qump(C2) places third on the ROUGE-1 recall and f-measure, and the difference between the top systems (ID=15 and ID=4) and our algorithm is statistically insignificant.

System 15 stands for the NUS summarizer from the National University of Singapore. This summarizer is based on the concept link approach (Ye et al., 2005). NUS method uses two features: sentence semantic similarity and redundancy minimization based on Maximal Marginal Relevance (MMR). The first one is computed as an overall similarity score between each sentence and the remainder of the document cluster. This overall similarity score reflects the strength of representative power of the sentence in regard to the rest of the document cluster and is used as the primary sentence ranking metric while forming the summary. Then, a module similar to MMR is employed to build the summary incrementally, minimizing redundancy and maintaining the summarys relevance to the query's topic. In order to reduce the run-time computational cost (required for a scan through all possible pairs of senses for all pairs of concepts), authors pre-computed the semantic similarity between all possible pairs of WordNet entries offline.

System 4 represents the Columbia summarizer from the Columbia University (Blair-Goldensohn, 2005). This is an adaptation of the DefScriber question answering (QA) system (Blair-Goldensohn et al., 2004). DefScriber (1) identifies relevant sentences which contain information pertinent to the target individual or term (i.e. the X in the "Who/What is X?" question); (2) incrementally clusters extracted sentences using a cosine

[1]we used ROUGE-1.5.5 version and the command line "-a -l 250 -n 2 -2 4 -u"

distance metric; then (3) selects sentences for output summary using a fitness function which maximizes inclusion of core definitional predicates, coverage of the highest ranking clusters, and answer cohesion; and finally (4) applies reference rewriting techniques to extracted sentences to improve readability of summary, using an auxiliary system (Nenkova and McKeown, 2003). The key adaptations made for the DUC 2005 task were in relevant-passage selection (step 1) by combining the following techniques:

1. Term frequency-based weighting for filtering the less relevant terms from the topic statement ("topic terms"), based on IDF calculated over a large news corpus;

2. Topic structure for adjustment the topic term weights with the simple heuristic of giving terms in the title double the weight of terms in the extended question/topic body;

3. Stemming for maximize coverage of relevant terms when measuring overlap of topic terms and document sentences;

4. Including content of the nearby-sentences in the determination of a given sentences relevance.

Using these techniques, the algorithm made two passes over each document. In the first pass assigning relevance scores to each sentence based on overlap with topic terms. In the second pass, these scores were adjusted using the first-pass scores of nearby sentences. Finally, the sentences scored above a certain cutoff were selected.

In comparison to the two top systems, our approach does not require any pre-computed data from the external resources (like NUS does), and has a very simple pipeline with a few stages (unlike the Columbia summarizer) which do not involve external tools and have a low computational cost.

The difference of Qump(C2) from system with ID=17 was statistically insignificant, and the difference from system with ID=11 was statistically significant.

Table 3 shows how the ROUGE-1 scores of our algorithm compare to the scores of 35 systems that participated in the DUC 2006 competition. Two options of our algorithm that correspond to constraints described in Section 3.7 appear in the table, denoted by Qump(C1-C2). Qump(C2) places second on the ROUGE-1 recall and f-measure, and the difference between the top system with ID=24 and our algorithm is statistically insignificant.

System ID	Recall	Precision	F-measure
15	0.3446	0.3436	0.3440
4	0.3424	0.3355	0.3388
Qump(C2)	0.3416	0.3334	0.3374
17	0.3400	0.3329	0.3363
11	0.3336	0.3134	0.3231
6	0.3310	0.3256	0.3282
19	0.3305	0.3249	0.3276
10	0.3304	0.3225	0.3263
7	0.3300	0.3211	0.3254
8	0.3292	0.3314	0.3301
5	0.3281	0.3406	0.3339
Qump(C1)	0.3276	0.3194	0.3233
25	0.3264	0.3197	0.3229
24	0.3223	0.3253	0.3237
9	0.3222	0.3138	0.3179
16	0.3209	0.3203	0.3205
3	0.3177	0.3179	0.3177
14	0.3172	0.3325	0.3235
12	0.3115	0.3043	0.3078
21	0.3107	0.3095	0.3100
29	0.3107	0.3159	0.3131
27	0.3069	0.2976	0.3021
28	0.3047	0.3074	0.3059
13	0.3039	0.3186	0.3109
18	0.3003	0.3350	0.3161
32	0.2977	0.3056	0.3014
30	0.2931	0.2900	0.2914
26	0.2824	0.3088	0.2949
2	0.2801	0.3052	0.2914
22	0.2795	0.3160	0.2878
31	0.2719	0.3062	0.2797
20	0.2552	0.3554	0.2930
1	0.2532	0.3104	0.2644
23	0.1647	0.3708	0.2196

Table 2: DUC 2005. ROUGE-1 scores.

ID 24 represents the IIITH-Sum system from the International Institute of Information Technology (Jagarlamudi et al., 2006). IIITH-Sum used two features to score the sentences, and then picked the top-scored ones to a summary in the greedy manner. The first feature is a query dependent adaptation of the HAL (Jagadeesh et al., 2005) feature, where an additional importance is given to a word/phrase of a query. The second feature calculates query-independent sentence importance, using external resources in the web. First, the Yahoo search engine was used to get a ranked list of retrieved documents, and a unigram language model was learned on a text content extracted from them. Then, Information Measure (IM), using entropy to compute the information content of a sentence based on the learned unigram model, was used for scoring a sentence. The final sentence ranks were computed as a weighted linear combination of modified HAL feature and IM.

In contrast to the IIIHT-Sum, our approach does not require any external resources and is strictly based on the internal content of the analyzed corpus. As results, it is also consumes less run-time.

The difference of Qump(C2) from system with

System ID	Recall	Precision	F-measure
24	0.3797	0.3781	0.3789
Qump(C2)	0.3745	0.3732	0.3745
12	0.3736	0.3734	0.3734
31	0.3675	0.3730	0.3702
10	0.3720	0.3680	0.3699
33	0.3700	0.3698	0.3699
15	0.3717	0.3675	0.3696
23	0.3726	0.3661	0.3692
28	0.3677	0.3707	0.3691
8	0.3702	0.3665	0.3683
27	0.3574	0.3707	0.3637
5	0.3665	0.3607	0.3635
Qump(C1)	0.3647	0.3623	0.3635
13	0.3553	0.3713	0.3629
3	0.3539	0.3650	0.3593
2	0.3587	0.3580	0.3583
6	0.3543	0.3567	0.3555
19	0.3552	0.3534	0.3542
4	0.3518	0.3567	0.3542
22	0.3518	0.3554	0.3536
29	0.3432	0.3598	0.3512
9	0.3373	0.3641	0.3492
32	0.3519	0.3466	0.3492
14	0.3501	0.3481	0.3490
30	0.3317	0.3575	0.3439
25	0.3374	0.3478	0.3425
20	0.3413	0.3412	0.3412
7	0.3417	0.3368	0.3392
16	0.3384	0.3377	0.3380
18	0.3335	0.3423	0.3377
17	0.3105	0.3647	0.3351
21	0.3244	0.3541	0.3344
34	0.3320	0.3300	0.3310
35	0.3058	0.3488	0.3242
26	0.3023	0.3399	0.3199
1	0.2789	0.3231	0.2962
11	0.1965	0.3014	0.2366

Table 3: DUC 2006. ROUGE-1 scores.

ID=12 was statistically insignificant, and the difference from system with ID=31 was statistically significant. It is not surprising that method C2 performed better that C1 as limiting frequent word sets to words appearing in a query only decreases the overall number of frequent word sets. In this case, many repetitive word sets that are related to the query are missed.

The actual running time of our Qump (both versions) was around 1-3 seconds per document sets. We also learned that long sentences do not affect computation cost of our approach.

5 Conclusions

This work introduces Qump, a system following a new MDL-based approach to a query-oriented summarization. Qump extracts sentences that best describe the query-related frequent set of words. The evaluation results show that Qump has an excellent performance. In absolute ranking, it outperforms all but two of participated systems in DUC 2005 competition and all but one of competing systems in DUC 2006 contest. According to significance test, Qump has the same performance as leading systems in both competitions (ID=15,4,7 in DUC-2005 and ID=24,12 in DUC-2006). In addition, Qump is an efficient algorithm having polynomial complexity. Qump's runtime is limited by Apriori that is known as a PSPACE-complete problem. However, because it is a rare occasion to have a set of words repeated in more than 4-5 different sentences in the entire document set, we have $O(n^5)$ frequent itemsets at most where n is the number of terms. The encoding process is bound by a number of frequent sets times a number of sentences (m) times a number of words (k). Therefore, we can say that Qump's runtime is polynomial in the number of terms and is bound by $O(m \times k \times n^5)$. In conclusion, the presented technique has the following advantages over other techniques: (1) It is unsupervised and does not require any external resources (many of the top-rated systems from the DUC competitions are supervised or use external data); (2) It has efficient time complexity (polynomial in the number of terms); (3) It is language-independent and can be applied on any language, as far as we have a tokenizer for this language (for example, we got excellent results with generic summarization in Chinese with this approach[2]); (4) Despite its robustness (independence on annotated data and language, and its efficiency), its performance is comparable to the one of the top systems.

In future, we intend to enrich this approach with word vectors for better match between a query and a sentence. Also, we plan to integrate it with a novel summarization technique using OLAP representation, where frequent itemsets of words will represent an additional dimension.

References

Rakesh Agrawal and Ramakrishnan Srikant. 1994. Fast algorithms for mining association rules. In *20th International Conference on Very Large Databases*, pages 487–499.

Sasha Blair-Goldensohn, Kathleen McKeown, and Andrew H. Schlaikjer, 2004. *Answering definitional questions: A hybrid approach*, chapter 4. AAAI Press.

Sasha Blair-Goldensohn. 2005. From definitions to complex topics: Columbia university at DUC 2005. In *Document Understanding Conference*.

Wauter Bosma, 2005. *Query-Based Summarization using Rhetorical Structure Theory*, pages 29–44. LOT.

[2]Litvak, M. and Vanetik, N. and Li, L., 2017, *Summarizing Weibo with Topics Compression*, unpublished manuscript.

Shaofeng Bu, Laks V. S. Lakshmanan, and Raymond T. Ng. 2005. Mdl summarization with holes. In *Proceedings of the 31st International Conference on Very Large Data Bases*, VLDB '05, pages 433–444.

Ping Chen and Rakesh Verma. 2006. A query-based medical information summarization system using ontology knowledge. In *Proceedings of the 19th IEEE Symposium on Computer-Based Medical Systems (CBMS06)*, pages 37–42.

John M. Conroy, Judith D. Schlesinger, and Jade Goldstein Stewart. 2005. CLASSY: Query-based multi-document summarization. In *DUC 05 Conference Proceedings*.

Hal Daumé III and Daniel Marcu. 2006. Bayesian query-focused summarization. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 305–312, Sydney, Australia, July. Association for Computational Linguistics.

DUC. 2005-2006. Document Understanding Conference. `http://duc.nist.gov`.

Jagarlamudi Jagadeesh, Prasad Pingali, and Vasudeva Varma. 2005. A relevance-based language modeling approach to DUC 2005. In *Document Understanding Conferences (along with HLT-EMNLP 2005)*.

Jagadeesh Jagarlamudi, Prasad Pingali, and Vasudeva Varma. 2006. Query independent sentence scoring approach to DUC 2006.

Laks V. S. Lakshmanan, Raymond T. Ng, Christine Xing Wang, Xiaodong Zhou, and Theodore J. Johnson. 2002. The generalized mdl approach for summarization. In *Proceedings of the 28th International Conference on Very Large Data Bases*, VLDB '02, pages 766–777.

Yanran Li and Sujian Li. 2014. Query-focused multi-document summarization: Combining a topic model with graph-based semi-supervised learning. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1197–1207, Dublin, Ireland, August. Dublin City University and Association for Computational Linguistics.

Chen Li, Yang Liu, and Lin Zhao. 2015. Using external resources and joint learning for bigram weighting in ilp-based multi-document summarization. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 778–787, Denver, Colorado, May–June. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of summaries. In *Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004)*, pages 25–26.

Marina Litvak, Mark Last, and Natalia Vanetik. 2015. Krimping texts for better summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1931–1935, Lisbon, Portugal, September. Association for Computational Linguistics.

Yan Liu, Sheng hua Zhong, and Wenjie Li. 2012. Query-oriented multi-document summarization via unsupervised deep learning. In *Proceedings of the Twenty-Sixth AAAI conference on Artificial Intelligence*, pages 1699–1705.

Ahmed A. Mohamed and Sanguthevar Rajasekaran. 2006. Query-based summarization based on document graphs. In *Proceedings of IEEE International Symposium on Signal Processing and Information Technology*, pages 408–410.

Ani Nenkova and Kathleen McKeown. 2003. References to named entities: a corpus study. In *NAACL-HLT 2003*.

Thanh-Son Nguyen, Hady W. Lauw, and Panayiotis Tsaparas. 2015. Review synthesis for micro-review summarization. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, WSDM'15, pages 169–178.

Tadashi Nomoto and Yuji Matsumoto. 2001. A new approach to unsupervised text summarization. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, pages 26–34.

Tadashi Nomoto. 2004. *Machine Learning Approaches to Rhetorical Parsing and Open-Domain Text Summarization*. Ph.D. thesis, Nara Institute of Science and Technology.

Jahna Otterbacher, Gunes Erkan, and Dragomir R. Radev. 2009. Biased lexrank: Passage retrieval using random walks with question-based priors. *Information Processing & Management*, 45(1):42–54.

Sun Park, Ju-Hong Lee, Chan-Min Ahn, Jun Sik Hong, and Seok-Ju Chun, 2006. *Query Based Summarization Using Non-negative Matrix Factorization*, volume 4253 of *Lecture Notes in Computer Science*, pages 84–89.

Frank Schilder and Ravikumar Kondadadi. 2008. Fastsum: Fast and accurate query-based multi-document summarization. In *Proceedings of ACL-08: HLT, Short Papers*, pages 205–208, Columbus, Ohio, June. Association for Computational Linguistics.

Jie Tang, Limin Yao, and Dewei Chen. 2009. Multi-topic based query-oriented summarization. In *SIAM International Conference Data Mining*.

Jilles Vreeken, Matthijs Leeuwen, and Arno Siebes. 2011. Krimp: Mining itemsets that compress. *Data Min. Knowl. Discov.*, 23(1):169–214, July.

Lu Wang, Hema Raghavan, Claire Cardie, and Vittorio Castelli. 2014. Query-focused opinion summarization for user-generated content. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1660–1669, Dublin, Ireland, August. Dublin City University and Association for Computational Linguistics.

Jennifer Williams, Sharon Tam, and Wade Shen. 2014. Finding good enough: A task-based evaluation of query biased summarization for cross-language information retrieval. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 657–669, Doha, Qatar, October. Association for Computational Linguistics.

Shiren Ye, Long Qiu, Tat-Seng Chua, and Min-Yen Kan. 2005. NUS at DUC 2005: understanding documents via concept links. In *Document Understanding Conference*.

Liang Zhou, Chin-Yew, and Eduard Hovy. 2006. Summarizing answers for complicated questions. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC-2006)*.