Krimping texts for better summarization

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

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

Mark Last
Ben-Gurion University of the Negev,
Beer Sheva, Israel
mlast@bgu.ac.il

Abstract

Automated text summarization is aimed at extracting essential information from original text and presenting it in a minimal, often predefined, number of words. In this paper, we introduce a new approach for unsupervised extractive summarization, based on the Minimum Description Length (MDL) principle, using the Krimp dataset compression algorithm (Vreeken et al., 2011). Our approach represents a text as a transactional dataset, with sentences as transactions, and then describes it by itemsets that stand for frequent sequences of words. The summary is then compiled from sentences that compress (and as such, best describe) the document. The problem of summarization is reduced to the maximal coverage, following the assumption that a summary that best describes the original text, should cover most of the word sequences describing the document. We solve it by a greedy algorithm and present the evaluation results.

1 Introduction

Many unsupervised approaches for extractive text summarization follow the maximal coverage principle (Takamura and Okumura, 2009; Gillick and Favre, 2009), where the extract that maximally covers the information contained in the source text, is selected. Since the exhaustive solution demands an exponential number of tests, approximation techniques, such as a greedy approach or a global optimization of a target function, are utilized. It is quite common to measure text informativeness by the frequency of its components– words, phrases, concepts, and so on.

A different approach that received much less attention is based on the Minimum Description Length (MDL) principle, defining the best summary as the one that leads to the *best compression* of the text 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 OLAP applications. (Lakshmanan et al., 2002; Bu et al., 2005) However, only a few works on text summarization using MDL can be found in the literature. Authors of (Nomoto and Matsumoto, 2001) used K-means clustering extended with MDL principle for finding diverse topics in the summarized text. Nomoto in (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 produce the encoding with the minimum number of bits.

This paper introduces a new MDL-based approach for extracting relevant sentences into a summary. The approach represents documents as a sequential transactional dataset and then compresses it by replacing frequent sequences of words by codes. The summary is then compiled from sentences that best compress (or describe) the document content. The intuition behind this approach says that a summary that best describes the original text should cover its most frequent word sequences. As such, the problem of summarization is very naturally reduced to the maximal coverage problem. We solve it by the greedy method which ranks sentences by their coverage of best compressing frequent word sequences and selects the top-ranked sentences to a summary. There are a few works that applied the common data mining techniques for calculating frequent itemsets from transactional data to the text summarization task (Baralis et al., 2012; Agarwal et al., 2011; Dalal and Zaveri, 2013), but none of them followed the MDL principle. The comparative results on three different corpora

Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 1931–1935,
Lisbon, Portugal, 17-21 September 2015.

show that our approach outperforms other unsupervised state-of-the-art summarizers.

2 Methodology

The proposed summarization methodology, denoted by Gamp[1], is based on the MDL principle that is defined formally as follows (Mitchell, 1997). 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 the description of M and $L(D|M)$ is the bit length of the dataset D encoded with M.

In our approach, we first represent an input text as a transactional dataset. Then, using the Krimp dataset compression algorithm (Vreeken et al., 2011), we build the MDL for this dataset using its frequent sequential itemsets (word sequences). The sentences that cover most frequent word sequences are chosen to a summary. The following subsections describe our methodology in detail.

2.1 Problem statement

We are given a single text or a collection of texts about the same topic, composed of a set of sentences $S_1, \ldots, S_n$ over terms $t_1, \ldots, t_m$. The word limit W is defined for the final summary.

We represent a text as a *sequential transactional dataset*. Such a dataset consists of *transactions* (sentences), denoted by $T_1, \ldots, T_n$, and unique items (terms[2]) $I_1, \ldots, I_m$. Items are unique across the entire dataset. The number n of transactions is called the *size* of a dataset. Transaction T_i is a sequence of items from $I_1, \ldots, I_m$, denoted by $T_i = (I_{i_1}, \ldots, I_{i_k})$; the same item may appear in different places within the same transaction. *Support* of an item sequence s in the dataset is the ratio of transactions containing it as a subsequence to the dataset size n, i.e., $supp(s) = \frac{|\{T \in D | s \subseteq T\}|}{n}$. Given a support bound $Supp \in [0, 1]$, a sequence s is called *frequent* if $supp(s) \geq Supp$.

According to the MDL principle, we are interested in the minimal size of a compressed dataset $D|CT$ after frequent sequences in D are encoded with the compressing set–codes from the *Coding Table* CT, where shorter codes are assigned to more frequent sequences. The description length of non-encoded terms is assumed proportional to their length (number of characters). We rank sentences by their coverage of the best compressing set, which is the number of CT members in the sentence. The sentences with the highest coverage score are added to the summary as long as its length does not exceed W.

2.2 Krimping text

The purpose of the Krimp algorithm (Vreeken et al., 2011) is to use frequent sets (or sequences) to compress a transactional database in order to achieve MDL for that database. Let $FreqSeq$ be the set of all frequent sequences in the database. A collection CT of sequences from $FreqSeq$ (called the *Coding Table*) is called *best* when it minimizes $L(CT) + L(D|CT)$. We are interested in both *exact* and *inexact* sequences, allowing a sequence to have gaps inside it as long as the ratio of sequence length to sequence length with gaps does not exceed a pre-set parameter $Gap \in (0, 1]$. Sequences with gaps make sense in text data, as phrasing of the same fact or entity in different sentences may differ. In order to encode the database, every member $s \in CT$ is associated with its binary *prefix code* c (such as Huffman codes for 4 members: 0, 10, 110, 111), so that the most frequent code has the shortest length. We use an upper bound C on the number of encoded sequences in the coding table CT, in order to limit document compression. Krimp-based extractive summarization (see Algorithm 1) is given a document D with n sentences and m unique terms. The algorithm parameters are described in Table 1:

#	*note*	*description*	*affects*
1	**W**	summary words limit	summary length
2	**Supp**	minimal support bound – minimal fraction of sentences containing a frequent sequence	number of frequent word sequences $\|FreqSeq\|$, compression rate
3	**C**	maximal number of codes	as in 2
4	**Gap**	maximal allowed sequence gap ratio	as in 2

Table 1: Parameters of Algorithm 1.

The algorithm consists of the following steps.

1. We find all frequent term sequences in the document using Apriori-TID algorithm (R and R, 1994) for the given $Supp$ and Gap and store them in set $FreqSeq$, which is kept in Standard Candidate Order[3]. The coding table CT is initialized to contain all single normalized terms and their frequencies. CT is always kept in Standard Cover Order[4] (Steps

[1] abbreviation of two words: GAp and kriMP

[2] normalized words following tokenization, stemming, and stop-word removal

[3] first, sorted by increasing support, then by decreasing sequence length, then lexicographically

[4] first, sorted by decreasing sequence length, then by decreasing support, and finally, in lexicographical order

1 and 2 in Algorithm 1).

2. We repeatedly choose frequent sequences from the set $FreqSeq$ so that the size of the encoded dataset is minimal, with every selected sequence replaced by its code. Selection is done by computing the decrease in the size of the encoding when each one of the sequences is considered to be a candidate to be added to CT (Step 3 in Algorithm 1).

3. The summary is constructed by incrementally adding sentences with the highest coverage of encoded term sequences (Step 4 in Algorithm 1) that are not covered by previously selected sentences. The sentences are selected in the greedy manner as long as the word limit W is not exceeded.

Algorithm 1: Gamp: Krimp-based extractive summarization with gaps

Input: (1) A document, containing sentences $S_1, \ldots, S_n$ after tokenization, stemming and stop-word removal;
(2) normalized terms $T_1, \ldots, T_m$;
(3) summary size W
(3) limit C on the number of codes to use;
(4) minimal support bound $Supp$;
(5) maximal gap ratio Gap.
Output: Extractive summary $Summary$
/ STEP 1: Frequent sequence mining */*
$FreqSeq \leftarrow$ inexact frequent sequences of terms from $\{T_1, \ldots, T_m\}$ appearing in at least $Supp$ fraction of sentences and having a gap ratio of at least Gap;
Sort $FreqSeq$ according to Standard Candidate Order;
/ STEP 2: Initialize the coding table */*
Add all terms $T_1, \ldots, T_m$ and their support to CT;
Keep CT always sorted according to Standard Cover Order;
Initialize prefix codes according to the order of sequences in CT; */* STEP 3: Find the best encoding */*
$EncodedData \leftarrow$ *PrefixEncoding*$(\{S_1, \ldots, S_n\}, CT)$;
$CodeCount \leftarrow 0$;
while $CodeCount < B$ *and* $FreqSeq \neq \emptyset$ **do**
  $BestCode \leftarrow c \in FreqSeq$ such that $L(CT \cup \{c\}) + L($*PrefixEncoding*$(\{S_1, \ldots, S_n\}, CT \cup \{c\}))$ is minimal;
  $CT \leftarrow CT \cup \{BestCode\}$;
  $FreqSeq \leftarrow FreqSeq \setminus \{BestCode\}$;
  Remove supersets of $BestCode$ from $FreqSeq$;
  $CodeCount$++;
end
$Summary \leftarrow \emptyset$; */* STEP 4: Build the summary */*
while $\#words(Summary) < W$ **do**
  Find the sentence S_i that covers the largest set $\mathcal{T}$ of terms in CT and add it to $Summary$;
  Remove terms of $\mathcal{T}$ from CT;
end
return $Summary$

Example 2.1 Let dataset D contain following three sentences (taken from the "House of cards" TV show):

S_1 = *A hunter must stalk his prey until the hunter becomes the hunted.*
S_2 = *Then the prey becomes the predator.*
S_3 = *Then the predator and the hunter fight.*

After stemming, tokenization and stop-word removal we obtain unique (stemmed) terms:

t_1 = *hunter*, t_2 = *must*, t_3 = *stalk*, t_4 = *prei*,
t_5 = *becom*, t_6 = *hunt*, t_7 = *predat*, t_8 = *fight*.
$supp(t_1) = supp(t_4) = supp(t_5) = supp(t_7) = \frac{2}{3}$.
$supp(t_2) = supp(t_3) = supp(t_8) = \frac{1}{3}$.

Now we can view sentences as the following sequences of normalized terms.

$S_1 = (t_1, t_2, t_3, t_4, t_1, t_5, t_4)$
$S_2 = (t_4, t_5, t_7)$
$S_3 = (t_7, t_1, t_8)$

Initial coding table CT will contain all frequent single terms in Standard Cover Order: $t_5, t_1, t_7, t_4, t_8, t_2, t_3$. Let the minimal support bound be $\frac{2}{3}$, i.e., to be frequent a sequence must appear in at least 2 sentences, and let the gap ratio be $\frac{1}{2}$. Also, let the limit C be 4, meaning that only the first four entries of the coding table will be used for encoding. There exists a frequent sequence (t_4, t_5) that appears twice in the text, once in S_1 with a gap and once in S_2 without a gap. We add it to the coding table according to the Standard Cover Order, generate prefix codes for the first four entries, and get

$CT =$

seq	*supp*	*code*	*code len*
(**prei**, **becom**)	2/3	0	1
becom	2/3	10	2
hunter	2/3	110	3
predat	2/3	111	3
prei	2/3	–	–
...	...	–	–

Now S_1 covers 3 out of 4 entries in CT, while S_2 and S_3 cover 2 terms each. If our summary is to contain just one sentence, we select S_1.

3 Experimental settings and results

We performed experiments on three corpora from the Document Understanding Conference (DUC): 2002, 2004, and 2007 (duc, 2002 2007), summarized in Table 2. DUC 2002 and 2007 each contains a set of single documents, and DUC 2004 contains 50 sets of 10 related documents. We generated summaries per single documents in the DUC 2002 and 2007 corpora, and per each set of related documents (by considering each set of documents as one meta-document) in DUC 2004, following the corresponding length restrictions. The summarization quality was measured

corpus	# documents	# summaries	summ len	doc size (KB)	type
DUC'02	533	2-3	100	1-20	SD
DUC'04	50	4	100	20-89	MD (50x10)
DUC'07	23	4	250	28-131	SD

Table 2: Corpora statistics.

by the ROUGE-1 and ROUGE-2 (Lin, 2004) recall scores, with the word limit indicated in Table 2, without stemming and stopword removal.

The results of the introduced algorithm (Gamp) may be affected by several input parameters: minimum support ($Supp$), codes limit (C), and the maximal gap allowed between frequent words (Gap). In order to find the best algorithm settings for a general case, we performed experiments that explored the impact of these parameters on the summarization results. First, we experimented with different values of support count in the range of $[2, 10]$. The results show that we get the best summaries using the sequences that occur in at least four document sentences.

A limit on the number of codes is an additional parameter.We explored the impact of this parameter on the quality of generated summaries. As we could conclude from our experiments, the best summarization results are obtained if this parameter is set to the maximal number of words in the summary, W. Consequently, we used 100 codes for summarizing DUC 2002 and DUC 2004 documents and 250 codes for DUC 2007 documents.

The maximal gap ratio defines a pattern for generating the frequent sequences and has a direct effect on their structure and number. Our experiments showed that allowing a small gap between words of a sequence helps to improve slightly the ranking of sentences, but the improvement is not significant. Thus we used $Gap = 0.8$ in comparative experiments for all corpora. The resulting settings for each corpus are shown in Table 3.

Corpus	Supp	Max. codes	Gap
DUC 2002 and 2004	4	100	0.8
DUC 2007	4	250	0.8

Table 3: Best settings.

We compared the Gamp algorithm with the two known unsupervised state-of-the-art summarizers denoted by Gillick (Gillick and Favre, 2009) and McDonald (McDonald, 2007). As a baseline, we used a very simple approach that takes first sentences to a summary (denoted by TopK). Table 4 contains the results of comparative evaluations. The best scores are shown in bold. Gamp outperformed the other methods on all datasets (using ROUGE-1 score). The difference between the scores of Gamp and Gillick (second best system) on DUC 2007 is highly significant according to the Wilcoxon matched pairs test. Based on the same test, the difference of scores obtained on the DUC 2004 is not statistically significant. On the DUC 2002, Gamp is ranked first, with insignificant difference from the second best (McDonald's) scores. Based on this result, we can conclude that MDL-based summarization using frequent sequences works better on long documents or multi-document domain. Intuitively, it is a very logical conclusion, because single short documents do not contain a sufficient number of frequent sequences. It is noteworthy that, in addition to the greedy approach, we also evaluated the global optimization with maximizing coverage and minimizing redundancy using Linear Programming (LP). However, experimental results did not provide any improvement over the greedy approach. Therefore, we report only the results of the greedy solution.

	ROUGE-1 Recall			ROUGE-2 Recall		
Algorithm	DUC'02	DUC'04	DUC'07	DUC'02	DUC'04	DUC'07
Gamp	**0.4421**	**0.3440**	**0.3959**	0.1941	**0.0829**	**0.0942**
Gillick	0.4207	0.3314	0.3518	0.1773	0.0753	0.0650
McDonald	0.4391	0.2955	0.3500	**0.1981**	0.0556	0.0672
TopK	0.4322	0.2973	0.3525	0.1867	0.0606	0.0706

Table 4: Comparative results.

4 Conclusions

In this paper, we introduce a new approach for summarizing text documents based on their Minimal Description Length. We describe documents using frequent sequences of their words. The sentences with the highest coverage of the best compressing set are selected to a summary. The experimental results show that this approach outperforms other unsupervised state-of-the-art methods when summarizing long documents or sets of related documents. We would not recommend using our approach for summarizing single short documents which do not contain enough content for providing a high-quality description. In the future, we intend to apply the MDL method to keyword extraction, headline generation, and other related tasks.

Acknowledgments

This work was partially supported by the U.S. Department of the Navy, Office of Naval Research.

References

Nitin Agarwal, Kiran Gvr, Ravi Shankar Reddy, and Carolyn Penstein Ros. 2011. Scisumm: A multi-document summarization system for scientific articles. In *Proceedings of the ACL-HLT 2011 System Demonstrations*, pages 115–120.

Elena Baralis, Luca Cagliero, Saima Jabeen, and Alessandro Fiori. 2012. Multi-document summarization exploiting frequent itemsets. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, SAC '12, pages 782–786.

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.

Mita K. Dalal and Mukesh A. Zaveri. 2013. Semisupervised learning based opinion summarization and classification for online product reviews. *Applied Computational Intelligence and Soft Computing*, 2013.

2002–2007. Document Understanding Conference. `http://duc.nist.gov`.

Dan Gillick and Benoit Favre. 2009. A Scalable Global Model for Summarization. In *Proceedings of the NAACL HLT Workshop on Integer Linear Programming for Natural Language Processing*, pages 10–18.

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.

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.

R. McDonald. 2007. A study of global inference algorithms in multi-document summarization. In *Advances in Information Retrieval*, pages 557–564.

Thomas M. Mitchell. 1997. *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edition.

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.

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

Hiroya Takamura and Manabu Okumura. 2009. Text summarization model based on maximum coverage problem and its variant. In *EACL '09: Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 781–789.

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