Multilingual Summarization with Polytope Model

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

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

Abstract

The problem of extractive text summarization for a collection of documents is defined as the problem of selecting a small subset of sentences so that the contents and meaning of the original document set are preserved in the best possible way. In this paper we describe the linear programming-based global optimization model to rank and extract the most relevant sentences to a summary. We introduce three different objective functions being optimized. These functions define a relevance of a sentence that is being maximized, in different manners, such as: coverage of meaningful words of a document, coverage of its bigrams, or coverage of frequent sequences of words. We supply here an overview of our system's participation in the MultiLing contest of SIGDial 2015.

1 Introduction

Automated text summarization is an active field of research in various communities, including Information Retrieval, Natural Language Processing, and Text Mining.

Some authors reduce summarization to the maximum coverage problem (Takamura and Okumura, 2009; Gillick and Favre, 2009) which, despite positive results, is known as NP-hard (Khuller et al., 1999). Because linear programming (LP) helps to find an accurate approximated solution to this problem it has recently become very popular in the summarization field (Gillick and Favre, 2009; Woodsend and Lapata, 2010; Hitoshi Nishikawa and Kikui, 2010; Makino et al., 2011).

Trying to solve a trade-off between summary quality and time complexity, we propose a summarization model solving the approximated maximum coverage problem by linear programming in polynomial time. We measure information coverage by an objective function and strive to obtain a summary that preserves its optimal value as much as possible. Three objective functions considering different metrics of *information* are introduced and evaluated. The main achievement of our method is a text representation model expanding a classic vector space model (Salton et al., 1975) to hyperplane and half-spaces and making it possible to represent an exponential number of extracts without computing them explicitly. This model also enables us to find the optimal extract by simple optimizing an objective function in polynomial time, using linear programming over rationals. For the first time, the frequent sequence mining was integrated with the maximal coverage approach in order to obtain a summary that best describes the summarized document. One of the introduced objective functions implements this idea.

Our method ranks and extracts significant sentences into a summary, without any need in morphological text analysis. It was applied for both single-document (MSS) and multi-document (MMS) MultiLing 2015 summarization tasks, in three languages–English, Hebrew, and Arabic. In this paper we present experimental results in comparison with other systems that participated in the same tasks, using the same languages.

2 Preprocessing and definitions

We are given a document or a set of related documents in UTF-8 encoding. Documents are split into sentences $S_1, ..., S_n$. All sentences undergo tokenization, stop-word removal, and stemming. For some languages, stemming may be very basic or absent, and a list of stop-words may be unavailable. All these factors affect summarization quality.

Unique stemmed words are called *terms* and are denoted by $T_1, ..., T_m$. Every sentence is modeled as a sequence of terms from $T_1, ..., T_m$ where each

Proceedings of the SIGDIAL 2015 Conference, pages 227–231,
Prague, Czech Republic, 2-4 September 2015.

term may appear zero or more times in a sentence. We are also given the desired number of words for a summary, denoted by $MaxWords$.

The goal of extractive summarization is to find a subset of sentences $S_1, ..., S_n$ that has no more than $MaxWords$ words and conveys as much information as possible about the documents. Because it is difficult, or even impossible, to know what humans consider to be the best summary, we approximate the human decision process by optimizing certain *objective functions* over representation of input documents constructed according to our model. The number of words in a summary, sentences, and terms, are represented as *constraints* in our model.

3 Polytope model

3.1 Definitions

In the polytope model (Litvak and Vanetik, 2014) a document is viewed as an integer sentence-term matrix $A = (a_{ij})$, where a_{ij} denotes the number of appearances of term T_j in sentence S_i. A row i of matrix A is used to define a linear constraint for sentence S_i as follows:

$$\sum_{j=1}^{m} a_{ij}x_{ij} \leq \sum_{j=1}^{m} a_{ij} \tag{1}$$

Equation (1) also defines the lower half-space in $\mathbb{R}^{mn}$ corresponding to sentence S_i. Together with additional constraints, such as a bound $MaxWords$ on the number of words in the summary, we obtain a system of linear inequalities that describes the intersection of corresponding lower half-spaces of $\mathbb{R}^{mn}$, forming a closed convex polyhedron called a *polytope*:

$$\begin{cases} \sum_{j=1}^{m} a_{ij}x_{ij} \leq \sum_{j=1}^{m} a_{ij},\ \forall i = 1..n \\ 0 \leq x_{ij} \leq 1,\ \forall i = 1..n,\ j = 1..m \\ \sum_{i=1}^{n}\sum_{j=1}^{m} a_{ij}x_{ij} \leq MaxWords \end{cases} \tag{2}$$

All possible extractive summaries are represented by vertices of the polytope defined in (2).

It remains only to define an *objective function* which optimum on the polytope boundary will define the summary we seek. Because such an optimum may be achieved not on a polytope vertex but rather on one of polytope faces (because we use linear programming over rationals), we need only to locate the vertex of a polytope closest to the point of optimum. This task is done by finding distances from the optimum to every one of the sentence hyperplanes and selecting those with minimal distance to the point of optimum. If there are too many candidate sentences, we give preference to those closest to the beginning of the document.

The main advantage of this model is the relatively low number of constraints (comparable with the number of terms and sentences in a document) and both the theoretical and practical polynomial running times of LP over rationals (Karmarkar, 1984).

3.2 Objective functions

In this section, we describe the objective functions we used in our system. Humans identify good summaries immediately, but specifying summary quality as a linear function of terms, sentences, and their parameters is highly nontrivial. In most cases, additional parameters, variables, and constraints must be added to the model.

3.3 Maximal sentence relevance

The first objective function maximizes relevance of sentences chosen for a summary, while minimizing pairwise redundancy between them.

We define relevance $cosrel_i$ of a sentence S_i as a *cosine similarity* between the sentence, viewed as a weighted vector of its terms, and the document. Relevance values are completely determined by the text and are not affected by choice of a summary. Every sentence S_i is represented by a sentence variable:

$$s_i = \textstyle\sum_{j=1}^{m} a_{ij}x_{ij} / \sum_{j=1}^{m} a_{ij} \tag{3}$$

Formally, variable s_i represents the hyperplane bounding the lower half-space of $\mathbb{R}^{mn}$ related to sentence S_i and bounding the polytope. Clearly, s_i assumes values in range $[0, 1]$, where 0 means that the sentence is completely omitted from the summary and 1 means that the sentence is definitely chosen for the summary. Relevance of all sentences in the summary is described by the expression

$$\sum_{i=1}^{n} cosrel_i s_i \tag{4}$$

Redundancy needs to be modeled and computed for every pair of sentences separately. We use additional redundancy variables red_{ij} for every pair S_i, S_j of sentences where $i < j$. Every one of these variables is $0 - 1$ bounded and achieves a value of 1 only if both sentences are chosen for

the summary with the help of these constraints:

$$\begin{cases} 0 \leq red_{ij} \leq 1,\ 0 \leq i < j \leq n \\ red_{ij} \leq s_i,\ red_{ij} \leq s_j \\ s_i + s_j - red_{ij} \leq 1 \end{cases} \quad (5)$$

The numerical redundancy coefficient for sentences S_i and S_j is their cosine similarity as term vectors, which we compute directly from the text and denote by $cosred_{ij}$. The objective function we use to maximize relevance of the chosen sentences while minimizing redundancy is

$$\max \sum_{i=1}^{n} cosrel_i s_i - \sum_{i=1}^{n}\sum_{j=1}^{n} cosred_{ij} red_{ij} \quad (6)$$

3.4 Sum of bigrams

The second proposed objective function maximizes the weighted sum of bigrams (consecutive term pairs appearing in sentences), where the weight of a bigram denotes its importance.

The importance $count_{ij}$ of a bigram (T_i, T_j) is computed as the number of its appearances in the document. It is quite possible that this bigram appears twice in one sentence, and once in another, and $i = j$ is possible as well.

In order to represent bigrams, we introduce new bigram variables bg_{ij} for $i, j = 1..m$, covering all possible term pairs. An appearance of a bigram in sentence S_k is modeled by a $0-1$ bounded variable bg_{ij}^k, and c_{ij}^k denotes the number of times this bigram appears in sentence S_k. A bigram is represented by a *normalized sum of its appearances* in various sentences as follows:

$$\begin{cases} 0 \leq bg_{ij}^k \leq 1,\ \forall i, j, k \\ bg_{ij} = \sum_{k=1}^{n} c_{ij}^k bg_{ij}^k / \sum_{k=1}^{n} c_{ij}^k \end{cases} \quad (7)$$

Additionally, the appearance bg_{ij}^k of a bigram in sentence S_k is tied to terms T_i and T_j composing it, with the help of variables x_{ki} and x_{kj} denoting appearances of these terms in S_k:

$$\begin{cases} bg_{ij}^k \leq x_{ki} \\ bg_{ij}^k \leq x_{kj} \\ x_{ki} + x_{kj} - bg_{ij}^k \leq 1 \end{cases} \quad (8)$$

The constraints in (8) express the fact that a bigram cannot appear without the terms composing it, and appearance of both terms causes, in turn, the appearance of a bigram. Our objective function is:

$$\max : \sum_{i=1}^{m}\sum_{j=1}^{m} count_{ij} bg_{ij} \quad (9)$$

3.5 Maximal relevance with frequent itemsets

The third proposed objective function modifies the model so that only the most important terms are taken into account.

Let us view each sentence S_i as a sequence $(T_{i1}, \ldots, T_{in})$ of terms, and the order of terms preserves the original word order of a sentence. Source documents are viewed as a database of sentences. Database size is n. Let $s = (T_{i1}, \ldots, T_{ik})$ be a sequence of terms of size k. *Support* of s in the database is the ratio of sentences containing this sequence, to the database size n.

Given a user-defined support bound $S \in [0, 1]$, a term sequence s is *frequent* if $support(s) \geq S$. Frequent term sequences can be computed by a multitude of existing algorithms, such as Apriori (Agrawal et al., 1994), FreeSpan (Han et al., 2000), GSP (Zaki, 2001), etc.

In order to modify the generic model described in (2), we first find all frequent sequences in the documents and store them in set F. Then we sort F first by decreasing sequence size and then by decreasing support, and finally we keep only top B sequences for a user-defined boundary B.

We modify the general model (2) by representing sentences as sums of their frequent sequences from F. Let $F = \{f_1, \ldots, f_k\}$, sorted by decreasing size and then by decreasing support. A sentence S_i is said to *contain* f_j if it contains it as a term sequence and no part of f_j in S_i is covered by sequences $f_1, \ldots, f_{j-1}$.

Let $count_{ij}$ denote the number of times sentence S_i contains frequent term sequence f_j. Variables f_{ij} denote the appearance of sequence f_j in sentence S_i. We replace the polytope (2) by:

$$\begin{cases} \sum_{j=1}^{k} count_{ij} f_{ij} \leq \sum_{j=1}^{k} count_{ij},\ \forall i = 1..n \\ 0 \leq f_{ij} \leq 1,\ \forall i = 1..n,\ j = 1..k \end{cases} \quad (10)$$

We add variables describing the relevance of each sentence by introducing sentence variables:

$$s_i = \sum_{j=1}^{k} count_{ij} f_{ij} / \sum_{j=1}^{k} count_{ij} \quad (11)$$

Defining a boundary on the length of a summary now requires an additional constraint because frequent sequences do not contain all the terms in the sentences. Summary size is bounded as follows:

$$\sum_{i=1}^{n} length_i s_i \leq MaxWords \quad (12)$$

Here, $length_i$ is the exact word count of sentence S_i.

Relevance $freqrel_i$ of a sentence S_i is defined as a *cosine similarity* between the vector of terms in S_i covered by members of F, and the entire document. The difference between this approach and the one described in Section 3.3 is that only frequent terms are taken into account when computing sentence-document similarity. The resulting objective function maximizes relevance of chosen sentences while minimizing redundancy defined in (5):

$$\max \sum_{i=1}^{n} freqrel_i s_i - \sum_{i=1}^{n} \sum_{j=1}^{n} cosred_{ij} red_{ij} \quad (13)$$

4 Experiments

Tables 4, 4, and 1 contain the summarized results of automated evaluations for MultiLing 2015, single-document summarization (MSS) task for English, Hebrew, and Arabic corpora, respectively. The quality of the summaries is measured by ROUGE-1 (Recall, Precision, and F-measure).(Lin, 2004) We also demonstrate the absolute ranks of each submission–P-Rank, R-Rank, and F-Rank–when their scores are sorted by Precision, Recall, and F-measure, respectively. Only the best submissions (in terms of F-measure) for each participated system are presented and sorted in descending order of their F-measure scores. Two systems–Oracles and Lead–were used as top-line and baseline summarizers, respectively. Oracles compute summaries for each article using the combinatorial covering algorithm in (Davis et al., 2012)–sentences were selected from a text to maximally cover the tokens in the human summary, using as few sentences as possible until its size exceeded the human summary, at which point it was truncated. Because Oracles can actually "see" the human summaries, it is considered as the optimal algorithm and its scores are the best scores that extractive approaches can achieve. Lead simply extracts the leading substring of the body text of the articles having the same length as the human summary of the article.

Below we summarize the comparative results for our summarizer (denoted in the following tables by **Poly**) in both tasks, in terms of Rouge-1, F-measure. For comparisons, we consider the best result out of 3 functions: coverage of frequent sequences for English and coverage of meaningful words for Hebrew and Arabic. **English**: 4^{th} places out of 9 participants in both MSS and MMS tasks. **Hebrew**: 3^{rd} place out of 7 and out of 9 participants in MSS and MMS tasks, respectively; and the highest recall score in MMS task. **Arabic**: 5^{th} place out of 7 systems in MSS task, and 4^{th} place out of 9 participants and the highest recall score in MMS task. As can be seen, the best performance for our summarizer has been achieved on the dataset of Hebrew documents. For example, only the top-line Oracles and the supervised MUSE summarizers outperformed our system in MSS task. Poly also outperformed Gillick (2009) model using ILP. The average running time for Poly is 500 ms per document.

system	P score	R score	F score	P-rank	R-rank	F-rank
Oracles	0.601	0.619	0.610	1	1	1
BGU-SCE-MUSE	0.488	0.500	0.494	3	2	2
CCS	0.477	0.495	0.485	6	3	4
Poly	**0.475**	**0.494**	**0.484**	**8**	**5**	**5**
EXB	0.467	0.495	0.480	13	4	9
NTNU	0.470	0.456	0.462	12	17	13
LCS-IESI	0.461	0.456	0.458	15	18	15
UA-DLSI	0.457	0.456	0.456	18	16	17
Lead	0.425	0.434	0.429	24	20	20

system	P score	R score	F score	P-rank	R-rank	F-rank
CCS	0.202	0.213	0.207	1	1	1
BGU-SCE-MUSE	0.196	0.210	0.203	2	2	2
Poly	**0.189**	**0.203**	**0.196**	**4**	**6**	**4**
EXB	0.186	0.205	0.195	5	4	5
Oracles	0.182	0.204	0.192	6	5	6
Lead	0.168	0.178	0.173	13	12	12
LCS-IESI	0.181	0.170	0.172	7	14	13

system	P score	R score	F score	P-rank	R-rank	F-rank
Oracles	0.630	0.658	0.644	1	1	1
BGU-SCE-MUSE	0.562	0.569	0.565	2	4	2
CCS	0.554	0.571	0.562	4	3	3
EXB	0.546	0.571	0.558	8	2	7
Poly	**0.545**	**0.560**	**0.552**	**10**	**9**	**9**
LCS-IESI	0.540	0.527	0.531	11	13	12
Lead	0.524	0.535	0.529	13	12	13

Table 1: MSS task. Rouge-1. **English**, **Hebrew**, and **Arabic**, top-down.

5 Conclusions and Future Work

In this paper we present an extractive summarization system based on a linear programming model. We represent the document as a set of intersecting hyperplanes. Every possible summary of a document is represented as the intersection of two or more hyperlanes. We consider the summary to be the best if the optimal value of the objective function is achieved during summarization. We introduce multiple objective functions describing the relevance of a sentence in terms of information coverage. The results obtained by automatic evaluation show that the introduced approach performs quite well for Hebrew and English. Only top-line and supervised summarizers outperform Poly on the Hebrew corpus. It is worth noting that our system is unsupervised and does not require annotated data, and it has polynomial running time.

References

Rakesh Agrawal, Ramakrishnan Srikant, et al. 1994. Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB*, volume 1215, pages 487–499.

S.T. Davis, J.M. Conroy, and J.D. Schlesinger. 2012. OCCAMS – An Optimal Combinatorial Covering Algorithm for Multi-document Summarization. In *Proceedings of the IEEE 12th International Conference on Data Mining Workshops*, pages 454–463.

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.

Jiawei Han, Jian Pei, Behzad Mortazavi-Asl, Qiming Chen, Umeshwar Dayal, and Mei-Chun Hsu. 2000. Freespan: frequent pattern-projected sequential pattern mining. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 355–359. ACM.

Yoshihiro Matsuo Hitoshi Nishikawa, Takaaki Hasegawa and Genichiro Kikui. 2010. Opinion Summarization with Integer Linear Programming Formulation for Sentence Extraction and Ordering. In *Coling 2010: Poster Volume*, pages 910–918.

N. Karmarkar. 1984. New polynomial-time algorithm for linear programming. *Combinatorica*, 4:373–395.

Samir Khuller, Anna Moss, and Joseph (Seffi) Naor. 1999. The budgeted maximum coverage problem. *Information Precessing Letters*, 70(1):39–45.

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 and Natalia Vanetik. 2014. Efficient summarization with polytopes. *Innovative Document Summarization Techniques: Revolutionizing Knowledge Understanding: Revolutionizing Knowledge Understanding*, page 54.

Takuya Makino, Hiroya Takamura, and Manabu Okumura. 2011. Balanced coverage of aspects for text summarization. In *TAC '11: Proceedings of Text Analysis Conference*.

G. Salton, C. Yang, and A. Wong. 1975. A vector-space model for information retrieval. *Communications of the ACM*, 18.

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.

Kristian Woodsend and Mirella Lapata. 2010. Automatic Generation of Story Highlights. In *ACL '10: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 565–574.

Mohammed J Zaki. 2001. Spade: An efficient algorithm for mining frequent sequences. *Machine learning*, 42(1-2):31–60.