MUSEEC: A Multilingual Text Summarization Tool

Marina Litvak[1], Natalia Vanetik[1], Mark Last[2], and Elena Churkin[1]

[1]Department of Software Engineering
Shamoon College of Engineering, Beer Sheva, Israel
{*marinal,natalyav*}*@sce.ac.il, elenach@ac.sce.ac.il*
[2]Department of Information Systems Engineering
Ben Gurion University of the Negev, Beer Sheva, Israel
mlast@bgu.ac.il

Abstract

The MUSEEC (MUltilingual SEntence Extraction and Compression) summarization tool implements several extractive summarization techniques – at the level of complete and compressed sentences – that can be applied, with some minor adaptations, to documents in multiple languages.

The current version of MUSEEC provides the following summarization methods: (1) MUSE – a supervised summarizer, based on a genetic algorithm (GA), that ranks document sentences and extracts top–ranking sentences into a summary, (2) POLY – an unsupervised summarizer, based on linear programming (LP), that selects the best extract of document sentences, and (3) WECOM – an unsupervised extension of POLY that compiles a document summary from compressed sentences. In this paper, we provide an overview of MUSEEC methods and its architecture in general.

1 Introduction

High quality summaries can significantly reduce the information overload of many professionals in a variety of fields. Moreover, the publication of information on the Internet in an ever–increasing variety of languages dictates the importance of developing *multi–lingual* summarization tools that can be readily applied to documents in multiple languages.

There is a distinction between *extractive* summarization that is aimed at the selection of a subset of the most relevant fragments – mostly complete sentences – from a source text, and *abstractive* summarization that generates a summary as a reformulated synopsis expressing the main idea of the input documents.

Unlike the abstractive summarization methods, which require natural language processing operations, language-independent summarizers work in an extractive manner, usually via ranking fragments of a summarized text by a relevance score and selecting the top-ranked fragments (e.g., sentences) into a summary. Because sentence scoring methods, like MUSE (MUltilingual Sentence Extractor) (Last and Litvak, 2012), use a greedy approach, they cannot necessarily find the best extract out of all possible combinations of sentences.

Another approach, based on the maximum coverage principle (McDonald, 2007; Gillick and Favre, 2009), tries to find the best subset of extracted sentences. This problem is known as NP-hard (Khuller et al., 1999), but an approximate solution can be found by the POLY algorithm (Litvak and Vanetik, 2013) in polynomial time.

Given the tight length constraints, extractive systems that select entire sentences are quite limited in the quality of summaries they can produce. Compressive summarization seeks to overcome this limitation by compiling summaries from compressed sentences that are composed of strictly relevant information(Knight and Marcu, 2002). WECOM (Weighted COMpression) summarization approach (Vanetik et al., 2016) combines methods for term weighting and sentence compression into a *weighted compression* model. WECOM extends POLY by utilizing the choice of POLY's objective functions for the term-weighting model.

In this paper, we present MUSEEC, a multilingual text summarization platform, which currently implements three single-document summarization algorithms: MUSE (Last and Litvak, 2012), POLY algorithm (Litvak and Vanetik, 2013), and WECOM (Vanetik et al., 2016).

Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics—System Demonstrations, pages 73–78,
Berlin, Germany, August 7-12, 2016.

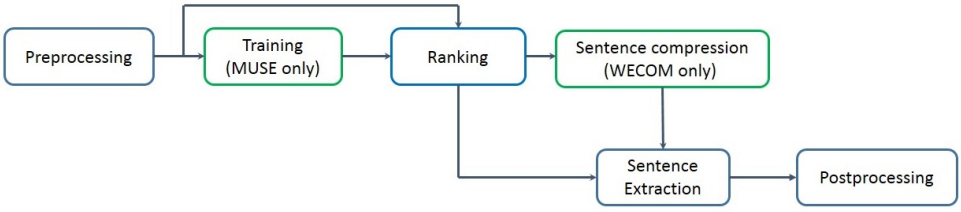

Figure 1: MUSEEC pipeline

2 MUSEEC: Overview

MUSEEC can be applied to documents in multiple languages. The current version was tested on nine languages: English, Hebrew, Arabic, Persian, Russian, Chinese, German, French, and Spanish, and its summarization quality was evaluated on three languages: English, Hebrew and Arabic.[1] The sections below provide brief descriptions of the system architecture and its main components.

2.1 MUSEEC Architecture

As shown in Figure 1, MUSEEC runs a pipeline that is composed of the following components:

1. **Preprocessing**. MUSEEC can work with documents written in any language by treating the text as a sequence of UTF-8 characters. It performs the following pre-processing operations: (1) sentence segmentation, (2) word segmentation, (3) stemming, and (4) stop-word removal. The last two operations are skipped if they are unavailable for a given language. Some optional, linguistic features require Part-of Speech (POS) tagging as a pre-processing step as well.
2. **Training**. This stage is optional and it is relevant only for the supervised MUSE algorithm. Given a set of training parameters, MUSE finds the best vector of weights for a linear combination of chosen sentence features. The resulting vector (trained model) can be saved and used for future summarization of documents in the same or any other language.
3. **Ranking**. At this stage, entire sentences or their parts (in case of compressive summarization) are ranked.
4. **Sentence Compression**. This stage is also optional and it is relevant only for compressive summarization performed by WECOM. Given ranked sentence parts, new, shorter sentences are compiled and ranked.
5. **Extraction**. Complete sentences are selected in the case of MUSE and POLY, and compressed sentences in the case of WECOM.
6. **Postprocessing**. The generated summaries can be post-processed by anaphora resolution (AR) and named entity (NE) tagging operations, if the corresponding tools are provided for a given language. MUSEEC utilizes Stanford CoreNLP package for English.
7. **Results Presentation**. Summaries are presented in two formats: sentences highlighted in the original document, selected by the user from a list of input documents, and a list of extracted sentences shown in their original order. The user can also sort sentences by their rank and see their scores.

MUSEEC allows the user to setup various summarization parameters, general and specific for a chosen algorithm, which are listed in Table 1. The table does not contain explicit WECOM settings because running WECOM equivalent to running POLY with "compressive" choice for the summary type.

2.2 MUltilingual Sentence Extractor (MUSE)

MUSE implements a supervised learning approach to extractive summarization, where the best set of weights for a linear combination of sentence scoring metrics is found by a GA trained on a collection of documents and their gold standard summaries. MUSE training can be performed from the MUSEEC tool. The obtained weighting vector is used for sentence scoring in future summarizations. Since most sentence scoring methods have a linear computational complexity, only the training phase of MUSE, which may be applied in advance, is time-consuming. In MUSEEC, one can use ROUGE-1 and ROUGE-2, Recall (Lin and Hovy, 2003) [2] as fitness functions for measuring summarization quality—similarity with gold stan-

[1] MUSEEC also participated in MultiLing 2011, 2013, and 2015 contests on English, Hebrew and Arabic, and demonstrated excellent results.

[2] We utilized the language-independent implementation of ROUGE that operates Unicode characters (Krapivin, 2014)

Parameter name	Description	Possible values	Default value
General			
Input path	Documents folder	Path name	
Output path	Summaries folder	Path name	
Summary type	Summarization approach	Compressive, Extractive	Extractive
Method	Summarization method	MUSE, POLY	MUSE (extr.), WECOM (comp.)
Limit by	Summary length unit	Words, Sentences, Ratio, Characters	Words
Limit	Summary length limit	Numeric value	Depends on unit
AR	Anaphora resolution	Check box	unchecked
NER	Named Entity tagging	Check box	unchecked
MUSE			
Mode	Train a new model, summarize documents evaluate summarizer	Train, Summarize, Evaluate	Summarize
Model	Model to save (training mode), or model to use (summarize mode)	Path name	
Sent. features	Sentence scoring features	31 basic metrics, 75 linguistic features	31 basic metrics
GA training			
Ratio split	Ratio of training data	[0..1]	1
Population	GA settings		500
Size	GA settings		100
Elite count	GA settings		5
Rouge	Rouge type as a fitness func.	1, 2	Rouge-1
POLY			
Objective function	Optimization function	8 functions, described in Section 2.3	Function 2 in Section 2.3

Table 1: MUSEEC general and method-specific parameters.

dard summaries, which should be *maximized* during the training. The reader is referred to (Litvak et al., 2010) for a detailed description of the optimization procedure implemented by MUSE.

The user can choose a subset of sentence metrics that will be included by MUSE in the linear combination. By default, MUSEEC will use the 31 language-independent metrics presented in (Last and Litvak, 2012). MUSEEC also allows the user to employ additional, linguistic features, which are currently available only for the English language. These features are based on lemmatization, multi-word expressions (MWE), NE recognition (NER), and POS tagging, all performed with Stanford CoreNLP package. The list of linguistic features is available in (Dlikman, 2015).

The training time of the GA is proportional to the number of GA iterations[3] multiplied by the number of individuals in a population, times the fitness (ROUGE) evaluation time. The summarization time (given a model) is linear in number of terms for all basic features.

[3]On average, in our experiments the GA performed $5-6$ iterations of selection and reproduction before reaching convergence.

2.3 POLYnomial summarization with POLYtopes (POLY)

Following the maximum coverage principle, the goal of POLY, which is an unsupervised summarizer, is to find the best subset of sentences that, under length constraints, can be presented as a summary. POLY uses an efficient text representation model with the purpose of representing all possible extracts[4] without computing them explicitly, that saves a great portion of computation time. Each sentence is represented by a hyperplane, and all sentences derived from a document form hyperplane intersections (polytope). Then, all possible extracts can be represented by subplanes of hyperplane intersections that are not located far from the boundary of the polytope. POLY is aimed at finding the extract that optimizes the chosen objective function.

MUSEEC provides the following categories of objective functions, described in detail in (Litvak and Vanetik, 2013).

1. **Maximal weighted term sum**, that maximizes the information coverage as a weighted term sum with following weight options supported:

   1. Term sum: all terms get weight 1;
   2. POS_F: terms appearing earlier in the text get higher weight;
   3. POS_L: terms appearing close to the end of the text get higher weight;
   4. POS_B: terms appearing closer to text boundaries (beginning or end) get higher weight;
   5. TF: weight of a term is set to its frequency in the document;
   6. TF_IDF: weight of a term is set to its tf*idf value;

2. **McDonald – maximal sentence coverage and minimal sentence overlap**, that maximizes the summary similarity to the text and minimizes the similarity between sentences in a summary, based on the Jaccard similarity measure (based on (McDonald, 2007));

3. **Gillick – maximal bigram sum and minimal sentence overlap**, that maximizes the information coverage as a bigram sum while minimizing the similarity between sentences (based on (Gillick

[4]exponential in the number of sentences

and Favre, 2009)).
All functions produce term weights in $[0, 1]$ that are then used for calculating the importance scores of each sentence.

Like in MUSE, the sentences with the highest score are added to the summary in a greedy manner. The overall complexity of POLY is polynomial in number of sentences. Further details about the POLY algorithm can be found in (Litvak and Vanetik, 2013).

2.4 WEighted Compression (WECOM)

In WECOM (Vanetik et al., 2016), we shorten sentences by iteratively removing Elementary Discourse Units (EDUs), which were defined as *grammatically independent parts of a sentence* in (Marcu, 1997). We preserve the important content by optimizing the weighting function that measures cumulative importance and preserve a valid syntax by following the syntactic structure of a sentence. The implemented approach consists of the following steps:
Term weight assignment. We apply a weighting model (using one of the options available for POLY) that assigns a non-negative weight to each occurrence of every term in all sentences of the document.
EDU selection and ranking. At this stage, we prepare a list of candidate EDUs for removal. First, we generate the list of EDUs from constituency-based syntax trees (Manning and Schütze, 1999) of sentences. Then, we omit from the list those EDUs that may create a grammatically incorrect sentence if they were to be removed. Finally, we compute weights for all remaining EDU candidates from term weights obtained in the first stage and sort them by increasing weight.
Budgeted sentence compression and selection. We define a summary cost as its length measured in words or characters[5]. We are given a *budget* for the summary cost, for example, the maximal number of words in a summary. The compressive part of WECOM is responsible for selecting EDUs in all sentences such that
(1) the weight to cost ratio of the summary is maximal; and
(2) the summary length does not exceed a given budget.

[5]depends on the user's choice of a summary maximal length

The compressed sentences are expected to be *more succinct* than the originals, to contain the important content from the originals, and to be grammatically correct. The compressed sentences are selected to a summary by the greedy manner. The overall complexity of WECOM is bound by $Nlog(N)$, where N is a number of terms in all sentences.

3 Experimental Results

Tables 2, 3, and 4 contain the summarized results of automated evaluations for the MultiLing 2015, single-document summarization (MSS) task. The quality of the summaries is measured by ROUGE-1 (Recall, Precision, and F-measure), (C.-Y, 2004). We also demonstrate the absolute ranks of each submission–P-Rank, R-Rank, and F-Rank–with their scores sorted by Precision, Recall, and F-measure, respectively. Only the best submissions (in terms of F-measure) for each participating 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. Since the Oracles system 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. The Lead system simply extracts the leading substring of the body text of the articles having the same length as the human summary of the article.

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

Table 2: MSS task. English.

As can be seen, MUSE outperformed all other participating systems except for CCS in Hebrew. CCS (the CCS-5 submission, to be precise) uses the document tree structure of sections, subsections, paragraphs, and sentences, and compiles a summary from the leading sentences of recursive

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

Table 3: MSS task. Hebrew.

system	P score	R score	F score	P-Rank	R-Rank	F-Rank
Oracles	0.630	0.658	0.644	1	1	1
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 4: MSS task. Arabic.

bottom-up interweaving of the node leading sentences, starting from leaves (usually, paragraphs in a section). POLY got very close scores, though it is an unsupervised approach and its comparison to a supervised summarizer is not fair.

MUSEEC also participated in the multi-document summarization (MMS) task, on English, Hebrew and Arabic. MUSE got first place on Hebrew, and 2^{nd} places on English and Arabic languages, out of 9 participants. POLY got third place on Hebrew, 4^{th} place on English, and 5^{th} place on Arabic, out of 9 participants. We explain the differences between scores in Hebrew and other languages by the lack of NLP tools for this language. For example, none of the competing systems performed stemming for Hebrew. Also, it is possible that the quality of the gold standard summaries or the level of agreement between annotators in Hebrew was lower than in other languages.

WECOM was evaluated in (Vanetik et al., 2016) on three different datasets (DUC 2002, DUC 2004, and DUC 2007) using automated and human experiments. Both automated and human scores have shown that compression significantly improves the quality of generated summaries. Table 5 contains results for POLY and WECOM summarizers on the DUC 2002 dataset. Statistical testing (using a paired T-test) showed that there is a significant improvement in ROUGE-1 recall between ILP concept-based extraction method of Gillick and Favre (2009) and WECOM with weights generated by Gillick and Favre's method. Another significant improvement is between ILP extraction method of McDonald (2007) and WECOM with weights generated by McDonald's method.

System	R-1 R	R-1 P	R-1 F	R-2 R	R-2 P	R-2 F
POLY + Gillick	0.401	0.407	0.401	0.160	0.162	0.160
WECOM + Gillick	**0.410***	0.413	0.409	0.166	0.166	0.165
POLY + McDonald	0.393	0.407	0.396	0.156	0.159	0.156
WECOM + McDonald	**0.401***	0.403	0.399	0.158	0.158	0.157
POLY + POS_F	0.448	0.453	0.447	0.213	0.214	0.212
WECOM + POS_F	0.450	0.450	0.447	0.211	0.210	0.210

Table 5: ROUGE-1 and -2 scores. DUC 2002.

Practical running times for MUSE (summarization) and POLY are tens of milliseconds per a text document of a few thousand words. WECOM running time is strictly dependent on the running time of dependency parsing performed by Stanford CoreNLP package, which takes $2-3$ seconds per sentence. Given pre-saved pre-processing results, WECOM takes tens of milliseconds per document as well.

4 Possible Extensions

MUSEEC functionality can be easily extended using its API. New algorithms can be added by implementing new ranking and/or compression modules of the pipeline. The pipeline is dynamically built before running a summarization algorithm, and it can be configured by a programmer[6]. The currently implemented algorithms can also be extended. For example, a new sentence feature for MUSE can be implemented by preparing one concrete class implementing a predefined interface. Using Java reflection, it does not require changes in any other code. New objective functions can be provided for POLY by implementation of one concrete class implementing the predefined interface and adding a few rows in the objective functions factory for creation instances of a new class (using factory method design pattern). Using dependency injections design pattern, MUSEEC can switch from Stanford CoreNLP package to any other tool for text preprocessing. MUSEEC is totally language-independent and works for any language with input texts provided in UTF-8 encoding. If no text processing tools for a given language are provided, MUSEEC skips the relevant stages in its pipeline (for example, it does not perform stemming for Chinese). Providing new NLP tools can improve MUSEEC summarization quality on additional languages. The subsequent stages in the MUSEEC pipeline (sentence

[6]Because building pipeline requires programming skills, this option cannot be applied from GUI.

ranking and compression) are totally language-independent and work with structured data generated during pre-processing. The optional capabilities of NE tagging and AR in the post-processing stage may be also extended with additional NLP tools for specific languages.

The programmer and user guidelines for extending and using MUSEEC can be provided upon request.

5 Final Remarks

In this paper, we present MUSEEC - a platform for summarizing documents in multiple languages. MUSEEC implements several variations of three single-document summarization methods: MUSE, POLY, and WECOM. The big advantage of MUSEEC is its multilinguality. The system has been successfully evaluated on benchmark document collections in three languages (English, Arabic, and Hebrew) and tested on six more languages. Also, MUSEEC has a flexible architecture and API, and it can be extended to other algorithms and languages.

However, MUSEEC has the following limitations: all its methods, especially compressive, are dependent on the pre-processing tools, in terms of summarization quality and performance. In order to improve coherency of the generated summaries, the MUSEEC user can apply AR as well as NE tagging to the generated summaries. More sophisticated post-processing operations performed on the extracted text in MUSEEC can further improve the user experience.

The MUSEEC tool, along with its code, is available under a BSD license on `https://bitbucket.org/elenach/onr_gui/wiki/Home`. In the future, we intend to prepare a Web application allowing users to apply MUSEEC online.

Acknowledgments

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

References

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

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.

A. Dlikman. 2015. Linguistic features and machine learning methods in single-document extractive summarization. Master's thesis, Ben-Gurion University of the Negev, Beer-Sheva, Israel. http://www.ise.bgu.ac.il/faculty/mlast/papers/Thesis-ver7.pdf.

D. Gillick and B. Favre. 2009. A scalable global model for summarization. In *Proceedings of the NAACL HLT Workshop on Integer Linear Programming for Natural Language Processing*.

S. Khuller, A. Moss, and J. Naor. 1999. The budgeted maximum coverage problem. *Information Precessing Letters*, 70(1):39–45.

K. Knight and D. Marcu. 2002. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, 139:91–107.

E. Krapivin. 2014. JRouge—Java ROUGE Implementation. https://bitbucket.org/nocgod/jrouge/wiki/Home.

M. Last and M. Litvak. 2012. Cross-lingual training of summarization systems using annotated corpora in a foreign language. *Information Retrieval*, pages 1–28, September.

C.-Y. Lin and E. Hovy. 2003. Automatic evaluation of summaries using N-gram co-occurrence statistics. In *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 71–78.

M. Litvak and N. Vanetik. 2013. Mining the gaps: Towards polynomial summarization. In *Proceedings of the International Joint Conference on Natural Language Processing*, pages 655–660.

M. Litvak, M. Last, and M. Friedman. 2010. A new approach to improving multilingual summarization using a Genetic Algorithm. In *ACL '10: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 927–936.

C. D. Manning and H. Schütze. 1999. *Foundations of statistical natural language processing*, volume 999. MIT Press.

D. Marcu. 1997. From discourse structures to text summaries. In *Proceedings of the ACL*, volume 97, pages 82–88.

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

N. Vanetik, M. Litvak, M. Last, and E. Churkin. 2016. An unsupervised constrained optimization approach to compressive summarization. Manuscript submitted for publication.