MUSE – A Multilingual Sentence Extractor

Marina Litvak†, Mark Last∗, Menahem Friedman∗ and Slava Kisilevich‡
†Sami Shamoon Academic College of Engineering
Beer-Sheva, Israel
Email: marinal@sce.ac.il
∗Ben Gurion University of Negev
Beer Sheva, Israel
Email: {litvakm, mlast, fmenahem}@bgu.ac.il
‡University of Konstantz
Konstantz, Germany
Email: slaks@dbvis.inf.uni-konstanz.de

Abstract—Multilingual Sentence Extractor (MUSE) is aimed at multilingual single-document summarization. MUSE implements the supervised language-independent summarization approach based on optimization of multiple statistical sentence ranking methods. The MUSE tool consists of two main modules: the training module activated in the offline mode, and the on-line summarization module. The training module can be provided with a corpus of summarized texts in any language. Then, it learns the best linear combination of user specified sentence ranking measures applying a Genetic Algorithm to the given training data. The summarization module performs real-time sentence extraction by computing sentence rankings according to the weighted model induced in the training phase. The main advantage of MUSE is its language-independency – it can be applied to any language given a gold standard summaries in that language. The performance of MUSE in our previous works was found to be significantly better than the best known state-of-the-art extractive summarization approaches and tools in the three different languages: English, Hebrew, and Arabic. Moreover, our experimental results in a cross-lingual domain suggest that MUSE does not need to be retrained on each new language, and the same weighting model can be used across several languages.

Index Terms—automated summarization, multi-lingual summarization, language-independent summarization, genetic algorithm, optimization, linear combination

I. INTRODUCTION

DOCUMENT tools should identify a minimum number of words and/or sentences to express a document’s main ideas. In this way, high quality summarization tools can significantly reduce the information overload faced daily by many professionals in a variety of fields.

The publication of information on the Internet in an ever-increasing variety of languages amplifies the importance of developing multilingual summarization tools. There is a particular need for language-independent statistical tools that can readily be applied to text in any language without depending on language-specific linguistic analysis. In the absence of such tools, the only alternative to language-independent summarization is the labor-intensive translation of the entire document into a common language.

Since a pure statistical method usually characterizes only one sentence feature, various attempts were made to use a combination of several methods as a ranking function [1], [2]. MUSE continues this effort by learning the best linear combination of 31 statistical language-independent sentence ranking features using a Genetic Algorithm (GA). With this approach, MUSE can be easily applied to multilingual extractive summarization. All sentence features comprising the linear combination are based on either a vector or a graph representation using a mere word and sentence segmentation of a document.

MUSE implements the multilingual summarization approach introduced in [4]. Evaluation of MUSE on three monolingual (English, Hebrew and Arabic) and one bilingual corpora of English and Hebrew documents has shown the following:
- MUSE performance is significantly better than TextRank [5] and Microsoft Word’s Autosummarize tool in all tested languages, as demonstrated in Table II.
- In English, MUSE outperforms such known summarization tools as MEAD [1] and SUMMA [2].
- MUSE does not need to be retrained on each language and the same model can be used across at least three–English, Hebrew, and Arabic–different languages.

Table I demonstrates the results of training and testing comprising the average ROUGE values obtained for English, Hebrew and bilingual corpora using 10-fold cross validation and reported in [4].

Table II shows the comparative results, also reported in [4], (ROUGE mean values) for three corpora, with the best summarizers on top. Results contain comparisons between: (1) a multilingual version of TextRank (denoted by

1Multilingual summarization is defined by [3] as “processing several languages, with summary in the same language as input”
2A web application of the MUSE-based summarizer will soon be made available at http://www.cs.bgu.ac.il/~litvakm/
3We used the corpus of summarized documents available at the Document Understanding Conference, 2002 [6] for English. This benchmark dataset contains 533 news articles, each accompanied by two by three human-generated abstracts of approximately 100 words each.
4For the Hebrew language we generated a corpus of 50 summarized news articles of 250 to 830 words each from the Website of the Haaretz newspaper (http://www.haaretz.co.il)
ML_TR (Mihalcea, 2005), (2) Microsoft Word’s Autosummarize function (denoted by MS_SUM), and (3) the best single scoring method in each corpus. As a baseline, we compiled summaries created from the initial sentences (denoted by POS_F). MUSE performed significantly better than TextRank in all three corpora and better than the best single methods COV_DEG in English and D_COV_I in Hebrew corpora respectively.

TABLE I
RESULTS OF 10-FOLD CROSS VALIDATION (LITVAK ET AL., 2010b)

<table>
<thead>
<tr>
<th></th>
<th>ENG</th>
<th>HEB</th>
<th>MULT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>0.4483</td>
<td>0.5993</td>
<td>0.5205</td>
</tr>
<tr>
<td>Test</td>
<td>0.4461</td>
<td>0.5936</td>
<td>0.5027</td>
</tr>
</tbody>
</table>

TABLE II
SUMMARIZATION PERFORMANCE: MEAN ROUGE-1 (LITVAK ET AL., 2010b)

<table>
<thead>
<tr>
<th>Metric</th>
<th>ENG</th>
<th>HEB</th>
<th>MULT</th>
</tr>
</thead>
<tbody>
<tr>
<td>MUSE</td>
<td>0.4461</td>
<td>0.5921</td>
<td>0.4633</td>
</tr>
<tr>
<td>COV_DEG</td>
<td>0.4363</td>
<td>0.5679</td>
<td>0.4588</td>
</tr>
<tr>
<td>D_COV_I</td>
<td>0.4251</td>
<td>0.5748</td>
<td>0.4512</td>
</tr>
<tr>
<td>POS_F</td>
<td>0.4190</td>
<td>0.5678</td>
<td>0.4440</td>
</tr>
<tr>
<td>ML_TR</td>
<td>0.4138</td>
<td>0.5190</td>
<td>0.4288</td>
</tr>
<tr>
<td>MS_SUM</td>
<td>0.3097</td>
<td>0.4114</td>
<td>0.3184</td>
</tr>
</tbody>
</table>

II. MULTILINGUAL SENTENCE EXTRACTOR (MUSE): OVERVIEW

A. Methodology

MUSE implements a supervised learning approach to language-independent extractive summarization where the best set of weights for a linear combination of sentence scoring methods is found by a genetic algorithm trained on a collection of document summaries. The weighting vector thus obtained is used for sentence scoring in future summarizations. Since most sentence scoring methods have a linear computational complexity, only the training phase of our approach is time-consuming.

Using MUSE, the user can choose the subset of totally 31 sentence metrics that will be included in the linear combination. All metrics are based on different text representation models and are language-independent since they do not rely on any language-specific knowledge. Figure 1 demonstrates the taxonomy of all 31 metrics. We divided them into three main categories—structure-, vector-, and graph-based—according to their text representation model, where each sub-category contains group of metrics using the same scoring method.

A detailed description of sentence metrics used by MUSE can be found in (Litvak et al., 2010b).

We found the best linear combination of the metrics depicted in Figure 1 using a Genetic Algorithm (GA). GAs are categorized as global search heuristics. Figure 2 shows a simplified GA flowchart. A typical genetic algorithm requires (1) a genetic representation of the solution domain, (2) a fitness function to evaluate the solution domain, and (3) some basic parameter settings like selection and reproduction rules.

We represent each solution as a vector of weights for a linear combination of sentence scoring metrics—real-valued numbers in the unlimited range normalized in such a way that they sum up to 1. The vector size is fixed and it equals the number of metrics used in the combination.

Defined over the genetic representation, the fitness function measures the quality of the represented solution. We use ROUGE-1 and ROUGE-2, Recall (Lin & Hovy, 2003) as a fitness functions for measuring summarization quality—similarity with gold standard summaries, which should be maximized during the training (optimization procedure). We used annotated corpus of summarized documents where each document is accompanied by several human-generated summaries—abstracts or extracts as a training set.

The reader is referred to (Litvak et al., 2010b) for a detailed description of the optimization procedure implemented by MUSE.

Algorithms 1 and 2 contain the pseudo-code for two independent phases of MUSE: training and summarization, respectively. Assuming efficient implementation, all metrics have a linear computational complexity relative to the total number of words in a document - \( O(n) \). As a result, summarization computation time, given a trained model, is also linear (at factor of the number of metrics in a combination). The training time is proportional to the number of GA iterations multiplied by the number of individuals in a population times the fitness evaluation (ROUGE) time. On average, in our experiments the GA performed 5 - 6 iterations—selection and reproduction—before reaching convergence.

B. Architecture

The current version of MUSE tool can be applied only to text documents or textual content of the HTML pages. It consists of two main modules: the training module activated in offline, and the real-time summarization module. Both modules utilize two different representations of documents described in (Litvak et al., 2010b): vector- and graph-based.
Algorithm 1 Step 1: Training

Input: Gold Standard - a corpus of summarized documents
Output: A weighted model \( W \) - vector of weights for each of \( N \) metrics

Step 1.1: Compute \( M \) - sentence-score matrix for all \( d \in D \)
Let \( R_1, R_2, \) and \( R_3 \) are \( d \) representations for all sentences \( s \in d \)
Calculate \( N \) metrics using \( R_1, R_2, \) and \( R_3 \)
Add metrics row for \( s \) into \( M \)
end for
end for

Step 1.2: Compute a vector \( W \) of metrics weights
Run a Genetic Algorithm on \( M \), given \( D \):
Initialize a population \( P \)
repeat
for all solution \( g \in P \) do
Generate a summary \( a \)
Evaluate \( a \) by ROUGE on summaries of \( D \)
end for
Select the best solutions \( G \)
\( P \) - a new population generated by \( G \)
until convergence - no better solutions are found
return a vector \( W \) of weights - output of a GA

Algorithm 2 Step 2: Summarizing a new document

Input: A document \( d \), maximal summary length \( L \), a trained weighted model \( W \)
Output: A set of \( n \) sentences, which were top-ranked by the algorithm as the most important.

Step 2.1: Compute a score of each sentence
Let \( R_1, R_2, \) and \( R_3 \) are \( d \) representations for all sentence \( s \in d \)
Calculate \( N \) metrics using \( R_1, R_2, \) and \( R_3 \)
Calculate a score as a linear combination according to \( W \)
end for
Step 2.2: Compile the document summary
Let \( S = \emptyset \) be a summary of \( d \)
repeat
get the top ranked sentence \( s_i \)
\( S = S \cup s_i \)
until \( S \) exceeds max length \( L \)
return \( S \)

The preprocessing module is responsible for constructing each representation, and it is embedded in both modules. The training module receives as input a corpus of documents, each accompanied by one or several gold-standard summaries—abstracts or extracts—compiled by human assessors. The set of documents may be either monolingual or multilingual and their summaries have to be in the same language as the original text. The training module applies a genetic algorithm to a document-feature matrix of precomputed sentence scores with the purpose of finding the best linear combination of features using any ROUGE metric (ROUGE-1 Recall as a default or specified by end-user) as a fitness function. The output/model of the training module is a vector of weights for user-specified sentence ranking features. In the current version of the tool, the user can choose from 31 vector-based and graph-based features. The recommendation for the best 10 features one can find in (Litvak et al., 2010a).

The summarization module performs summarization of input text/texts in real time. Each sentence of an input text obtains a relevance score according to the trained model, and the top ranked sentences are extracted to the summary in their original order. The length of resulting summaries is limited by a user-specified value (maximum number of words in the text extract or a ratio). Being activated in real-time, the summarization module is expected to use the model trained on the same language as input texts. However, if such model is not available (no annotated corpus in the text language),
the user can choose the following: (1) the model trained on some other language/corpus can be used (in (Litvak et al., 2010b) we show that the same model can be efficiently used across different languages), or (2) user-specified weights for each sentence feature (from 31 provided in the system) in the linear combination can be used for summarization.

The preprocessing module performs the following tasks: (1) sentence segmentation, (2) word segmentation, (3) vector space model construction using tf and/or tf-idf weights, (4) a word-based graph representation construction, (5) a sentence-based graph representation construction, and (6) document metadata construction, including such information like frequency (tf and tf-idf) for each unique term, its location inside the document, etc. The outputs of this submodule are: sentence segmented text (SST), vector space model (VSM), the document graphs, and the metadata stored in the xml files. Steps (1) and (2) are performed by the text processor submodule, which is implemented using Strategy Design Pattern (Freeman et al., 2004) and consists of three elements: filter, reader and sentence segmenter. The filter works on the Unicode character level and performs such operations like identification of characters, digits, punctuations and normalization (optional for some languages). The reader invokes the filter, constructs word chunks from the input stream and identifies the following states: words, special characters, white spaces, numbers, URL links and punctuation marks. The sentence segmenter invokes reader and divides the input space into sentences. By implementing different filters, the reader can work either with a specific language (taking into account its intricacies) or with documents written in arbitrary language. Figure 4 presents a small text example and its word-based graph representation.

Figure 3 shows the general architecture of the MUSE system.

![MUSE architecture](image-url)

C. Use Case

MUSE has five possible use cases demonstrated in Figure 9 and briefly described in Table III: Configure, Train, Summarize, Rouge, and Evaluate.

In the Configure use case, the user is required to specify the following parameters: folder paths to input documents being summarized, gold standard, output summaries, maximal length of a summary, preprocessing settings. The advanced user can change the default settings for a GA: population size, crossover and mutation probabilities, etc., training settings: splitting to the training and test data, preprocessing settings: maximal size of a graph representation, etc.. All specified settings can be used in the next user actions aka training and/or summarization or stored for the later use.

In the Train use case, the user trains genetic algorithm on the training corpus of annotated documents, where the optimal weights for the linear combination of sentence scoring metrics are determined. The weighted model can be used in the subsequent Summarize use case or stored for the later use.

The MUSE system can be evaluated on a new annotated dataset in the Evaluate use case.

In the Summarize use case, the user can get a summary for a single input document or set of summaries for a set of input documents. The pre-specified, new or modified settings can be used. The pre-trained or a new trained weighted model can be used.

In the fifth, Rouge use case, the user can apply ROUGE evaluation toolkit on existing—just produced or stored in advance—summaries. The ROUGE-1 Recall metric is applied by default, but it can be changed in the configuration use case.

D. Features

MUSE software has the following unique features:

- **Multilingual summarization.** The way in which MUSE processes a text is fully multilingual. All statistical metrics for sentence ranking used by MUSE do not require any language-specific analysis or knowledge, that allow MUSE to process texts in any language. Figures 6, 7, and 8 demonstrate the documents and their summaries—in a source language and translated to English—generated
Fig. 5. Input file (AP880228-0097 from DUC 2002 collection) and its extract by MUSE.

by MUSE for Arabic, Hebrew and English languages, respectively. Figures 6, 7 and 8 demonstrate extracts produced by MUSE for Arabic, Hebrew and English texts, respectively.

- **Flexible pre-processing.** User is allowed to add the following language-specific analysis to the MUSE processing: stopwords removal, stemming, sentence segmentation and POS tagging, by configuring the system and providing necessary tools or data. For example, the user can decide that he/she wants to remove stopwords by providing the stopwords list in the processed language and turning on the "remove stopwords" parameter. Also, many parameters for constructing a document representation are configurable.
A rich choice of statistical sentence features. User can choose from 31 sentence features provided by MUSE for summarization by configuring their weights in a linear combination. Our recommendation for 10 best metrics identified by cluster analysis of their performance on English and Hebrew corpora can be found in (Litvak et al., 2010a).

Easy to use. Text and HTML documents can be summarized with just one click. Figure 5 shows extract produced by MUSE for one of the DUC 2002 (DUC, 2002) documents.

Cross-lingual use of multiple ranking models. MUSE allows to use the same trained model across different languages. As result, no need in retraining on a new language.

Fig. 7. Hebrew document titled "Netanyahu and Abbas agreed to complete negotiations within a year" and its summary.

(a) Summarized document

(b) Original summary

(c) Translated summary

Fig. 8. English document titled "Images reveal Indonesian tsunami destruction" and its summary.

•
### TABLE III
USE CASE DESCRIPTION

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Goal</th>
<th>Precondition</th>
<th>Postcondition</th>
<th>Brief</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configure</td>
<td>Specify preprocessing and summarizer settings</td>
<td>None</td>
<td>Stored configuration file for later use</td>
<td>User specifies all necessary parameters as: path to the input document/s, summary length, gold standard folder, etc. User can store his settings for the later use.</td>
</tr>
<tr>
<td>Train</td>
<td>Train a model</td>
<td>Parameters settings, train and test data</td>
<td>Trained model - weights for linear combination</td>
<td>Train the genetic algorithm on the training document set. The trained model can be stored for the later use.</td>
</tr>
<tr>
<td>Evaluate</td>
<td>Evaluate the system</td>
<td>Parameters settings, gold standard summaries</td>
<td>Average train and test scores (ROUGE)</td>
<td>Evaluate the system on the given document set using 10-fold cross-validation. The average ROUGE score is presented to the user.</td>
</tr>
<tr>
<td>Summarize</td>
<td>Summarize the input document/s</td>
<td>Settings and a weighted model</td>
<td>Summary for each input document</td>
<td>User can get a summary for each input document and store it in the file system. Also, different output representations are provided to the user: sentence scores, highlighted and sorted sentences.</td>
</tr>
<tr>
<td>Rouge</td>
<td>Calculate ROUGE score for the input summaries</td>
<td>Settings, input and gold standard summaries</td>
<td>ROUGE score</td>
<td>User can get a ROUGE score for the input summaries given gold standard summaries for the document set.</td>
</tr>
</tbody>
</table>

**Fig. 9. Use case diagram of MUSE**

In this article we described MUSE—a supervised language-independent summarizer for the text documents based on sentence extraction.

We described and detailed the MUSE architecture, approach and use cases.

MUSE does not require any language-specific knowledge and can be applied to any language with a minimum amount of text pre-processing. Moreover, our experiments show that the same weighting model is applicable across multiple languages.

In general, we can conclude that combination of as many independent statistic features as possible can compensate the lack of linguistic analysis and knowledge for selecting the most informative sentences to a summary. More features can be added to our system, and/or another supervised model can be used for the learning and optimization of a linear combination. We believe that such an approach generally works when retrained on different genres and languages.

We can recommend the following: If a corpus in the target language exists, the best approach is to train MUSE on the target-language corpus, while periodically updating the trained model when new annotated data becomes available. If there is a corpus in any source language, but no high-quality target-language corpus is available, we would recommend to use the model trained on the source language corpus for summarizing documents in the target language.

In future work, MUSE may be evaluated on additional languages and language families, incorporate threshold values for threshold-based metrics into the GA-based optimization procedure, improve performance of similarity-based metrics in the multilingual domain, apply additional optimization techniques like Evolution Strategy (Beyer & Schwefel, 2002), which is known to perform well in a real-valued search space, and extend the search for the best summary to the problem of multi-objective optimization, combining several summary quality metrics.
ACKNOWLEDGMENTS
We are grateful to Nataly Alalouf for implementing the stand-alone and the web versions of MUSE.

REFERENCES