# Multilingual Single-Document Summarization with MUSE

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

**Mark Last**
Department of Information Systems
Engineering, Ben Gurion University
Beer Sheva, Israel
mlast@bgu.ac.il

## Abstract

MUltilingual Sentence Extractor (MUSE) is aimed at multilingual single-document summarization. MUSE implements a supervised language-independent summarization approach based on optimization of multiple sentence ranking methods using a Genetic Algorithm. The main advantage of MUSE is its language-independency – it is using statistical sentence features, which can be calculated for sentences in any language.

In our previous work, the performance of MUSE was found to be significantly better than the best known state-of-the-art extractive summarization approaches and tools in three different languages: English, Hebrew, and Arabic. Moreover, our experimental results in the cross-lingual domain suggest that MUSE does not need to be retrained on a summarization corpus in each new language, and the same weighting model can be used across several languages (Last and Litvak, 2012).

MUSE participated in the MultiLing 2013 single document summarization task on three languages: English, Hebrew and Arabic. Due to a very limited time that was given to the participants to run their systems on the MultiLing 2013 data, the results submitted to evaluation were obtained by summarizing the documents using models *pre-trained* on *different* corpora. As such, no training has been performed on the MultiLing 2013 corpus.

## 1 MUltilingual Sentence Extractor (MUSE): Overview

### 1.1 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 documents and their 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. The available metrics are based on various 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 subcategory contains group of metrics using the same scoring method.

A detailed description of sentence metrics used by MUSE can be found in (Last and Litvak, 2012).

The best linear combination of the metrics depicted in Figure 1 can be found 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 scor-

*Proceedings of the MultiLing 2013 Workshop on Multilingual Multi-document Summarization*, pages 77–81,
Sofia, Bulgaria, August 9 2013. ©2013 Association for Computational Linguistics

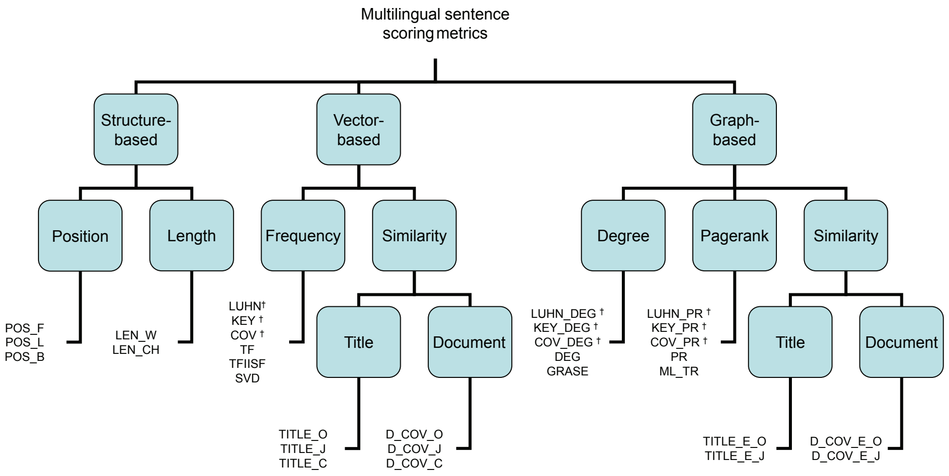

Figure 1: Taxonomy of language-independent sentence scoring metrics (Litvak et al., 2010b)

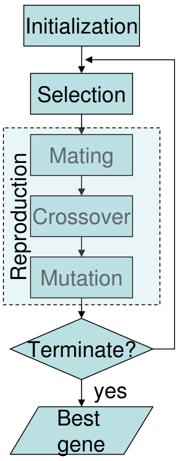

Figure 2: Simplified flowchart of GA

**Algorithm 1** Step 1: Training

```
Require: Gold Standard - a corpus of summarized documents $D$, $N$ chosen metrics
Ensure: A weighted model $W$ - vector of weights for each of $N$ metrics
  Step 1.1: Compute $M$ - sentence-score matrix
  for all $d \in D$ do
      Let $R_1$, $R_2$, and $R_3$ are $d$ representations
      for all sentences $s \in d$ do
          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
```

ing 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 to the number of metrics used in the combination.

Defined over the genetic representation, the fitness function measures the quality of the represented solution. We can use ROUGE-1 and ROUGE-2, Recall (Lin and 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 use an 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, the summary extraction time, given a trained model, is also linear (in 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—

**Algorithm 2** Step 2: Summarizing a new document

```
Require: A document $d$, maximal summary length $L$, a trained weighted model $W$
Ensure: 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 sentense $s \in d$ do
      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 $s_i$
      $S = S \bigcup s_i$
  until $S$ exceeds max length $L$
  return $S$
```

selection and reproduction—before reaching convergence.

### 1.2 Architecture

The current version of MUSE tool can be applied only to text documents or textual content of 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. 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 as a fitness function. ROUGE-1 Recall is used as a default unless specified otherwise by the end-user. 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 can be found 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, maximum number of sentences or a compression 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 one of the following options: (1) a model trained on some other language/corpus (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.

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 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 (in this case, a general filtering according to UTF-8 encoding is performed).

Figure 3 shows the general architecture of the MUSE system.

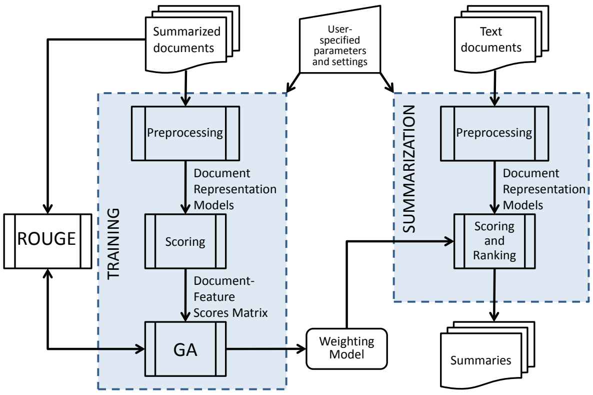

Figure 3: MUSE architecture

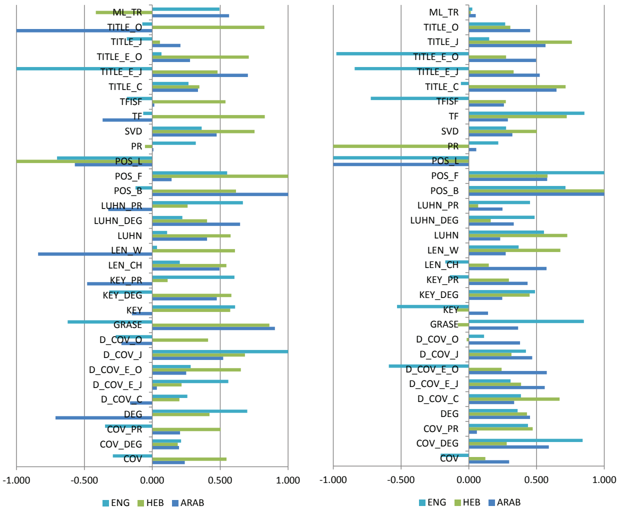

Figure 4: Models trained on monolingual corpora: ROUGE-1 (left) and ROUGE-2 (right)

## 2 Training of MUSE

Since a very limited time was given to participants to run their summarizers on the MultiLing 2013 dataset, we did not perform training on a new data. The models obtained from training MUSE on monolingual corpora of English, Hebrew, and Arabic texts in 2011 (Last and Litvak, 2012), have been used for summarization in three languages. Both ROUGE-1 and ROUGE-2 have beed used for building the models. In the current settings, ROUGE-1-based models were utilized.

The English text material used in the experiments comprised the corpus of summarized documents available for the summarization task at the Document Understanding Conference 2002 (DUC, 2002). This benchmark dataset contains 533 news articles, each accompanied by two to three human-generated *abstracts* of approximately 100 words each.

For the Arabic language, we used a corpus compiled from 90 news articles. Each article was summarized by three native Arabic speakers selecting the most important sentences into an *extractive* summary of approximately 100 words each.

For the Hebrew language, we used a corpus where 120 news articles of 250 to 830 words are summarized by five human assessors each.

The documents from all corpora have a title as the first sentence.

ROUGE-1 and ROUGE-2 metrics (Lin, 2004) have been used as a fitness function during the training of MUSE. The same metrics have been used for evaluation of generated summaries in three languages. In order to use the ROUGE toolkit on Hebrew and Arabic, it was adapted to these languages by specifying the regular expressions for a single “word” using Hebrew and Arabic characters.

Figure 4 present models learned by MUSE on different monolingual corpora using ROUGE-1 and ROUGE-2, respectively. The actual results in the trained models include some negative values.

The evaluation results of MUSE on three monolingual corpora using 10-fold cross validation showed its significant superiority over *TextRank* (Mihalcea, 2005), the best known language-independent unsupervised approach.

## 3 Experimental Results

According to the results of automated evaluation in MultiLing 2013 (N-gram graph methods: AutoSummENG, MeMoG, NPowER), MUSE took **fourth** place in English corpus (out of 7 systems), **third** place in Hebrew (out of 5 summarizers), and the **first** place in Arabic (out of 6 participants). We believe, that training MUSE on the original data and using correct titles[1] (by parsing xml documents) may significantly improve its results.

[1] Due to the time constraints of the single-document summarization task, we used a simple txt format of summarized documents in the published dataset, where the title is not separated from the first sentence by punctuation marks.

## References

DUC. 2002. Document Understanding Conference. http://duc.nist.gov.

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.

Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using N-gram cooccurrence 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.

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.

M. Litvak, S. Kisilevich, D. Keim, H. Lipman, A. Ben-Gur, and M. Last. 2010a. Towards language-independent summarization: A comparative analysis of sentence extraction methods on english and hebrew corpora. In *Proceedings of the CLIA/COLING 2010*.

Marina Litvak, Mark Last, and Menahem Friedman. 2010b. 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.

Rada Mihalcea. 2005. Language independent extractive summarization. In *AAAI'05: Proceedings of the 20th National Conference on Artificial Intelligence*, pages 1688–1689.