clstk: The Cross-lingual Summarization Tool-kit

by

Nisarg Jhaveri, Manish Gupta, Vasudeva Varma

in

12th ACM Conference on Web Search and Data Mining
(WSDM-2019)

Melbourne, Australia

Report No: IIIT/TR/2019/-1

Centre for Search and Information Extraction Lab
International Institute of Information Technology
Hyderabad - 500 032, INDIA
February 2019
c1stk: The Cross-lingual Summarization Tool-kit

Nisarg Jhaveri  
nisarg.jhaveri@research.iiit.ac.in  
IIIT-H, Hyderabad, India

Manish Gupta  
gmanish@microsoft.com  
Microsoft, Hyderabad, India

Vasudeva Varma  
vv@iiit.ac.in  
IIIT-H, Hyderabad, India

ABSTRACT
Cross-lingual summarization (CLS) aims to create summaries in a target language, from a document or document set given in a different, source language. Cross-lingual summarization can play a critical role in enabling cross-lingual information access for millions of people across the globe who do not speak or understand languages having large representation on the web. It can also make documents originally published in local languages quickly accessible to a large audience which does not understand those local languages. Though cross-lingual summarization has gathered some attention in the last decade, there has been no serious effort to publish rigorous software for this task. In this paper, we provide a design for an end-to-end CLS software called clstk. Besides implementing a number of methods proposed by different CLS researchers over years, the software integrates multiple components critical for CLS. We hope that this extremely modular tool-kit will help CLS researchers to contribute more effectively to the area.

CSC CONCEPTS
• Information systems → Summarization; Multilingual and cross-lingual retrieval;

KEYWORDS
Cross-lingual summarization; Document summarization tool-kit

ACM Reference Format:

1 INTRODUCTION

Although English is the most popular language on the web, many highly-populated countries like Egypt, China and India have other (non-English) languages like Arabic, Chinese, and Hindi respectively as the most spoken languages. In such countries, most content still gets published online in English first. While some of the content is later published in regional languages, a large amount does not appear in regional languages. Automatic cross-lingual summarization (CLS) systems help in summarizing the information contained in a “rich language” document to a “poor language”. Jhaveri et al. [5] argue that automatic CLS can make a high impact in the current scenario where automatic machine translation systems are not yet perfect.

In this paper, we describe a tool-kit for cross-lingual summarization, c1stk. The tool-kit is intended for both, developers and researchers working on CLS. End-users wanting to use CLS in real-world applications can also benefit from the tool-kit. The goals of the system are as follows.

• Help researchers quickly implement new models as well as compare with existing models for CLS.
• Provide a unified platform and API for researchers to publish their CLS models.
• Make different algorithms and models for CLS accessible for use in real-world end-user applications.

The proposed tool-kit contains a collection of several CLS methods, as well as bootstrap code to develop new methods for the problem. The tool contains summary evaluation module, which can be helpful in evaluating and experimenting with CLS easily. We also publish a new CLS evaluation dataset for English to Gujarati summarization. The dataset is prepared by translating summaries from DUC 2004 summarization dataset to Gujarati. We discuss more details on this dataset in Section 4.1.

Additionally, we demonstrate the use of the system by running several experiments on two CLS datasets, one each for English to Gujarati and for English to Hindi summarization. Our major contributions through this work are as follows.

• Python tool-kit for easy implementation and experimentation with cross-lingual summarizers.
• Manually translated summaries from DUC 2004 into Gujarati, that can be used to evaluate English to Gujarati CLS.
• Implementation of several cross-lingual summarizers which can be used as baselines in future work on CLS.

clstk, is freely available at https://github.com/nisargjhaveri/clstk with documentation at https://clstk.readthedocs.io which also includes dependencies, installation and development guide. Also, we have uploaded a demo video at https://youtu.be/SdGZ_Ns6Rqs. The rest of the paper is organized as follows. Section 2 describes some related work on CLS. Section 3 describes different components and modules of the system. Sections 4 and 5 describe the datasets and preliminary experiments demonstrating the usage of the tool-kit. Section 6 describes companion tools and clstk roadmap. Finally, we conclude with a brief summary in Section 7.

2 RELATED WORK

While summarization in general is a very well studied topic [10], we focus on a special type of summarization: CLS. Recently, there has been a lot of work on CLS. Wan et al. [15] extract multiple candidate summaries and then rank the summaries to get the best summary for the document set in the target language. Zhang et al.
Wan [13] proposed ranking framework which simultaneously uses Translation
Sentence Simplification
Translation Quality Estimation
Core
Summarizer
Evaluation
User Summaries
[13x59]User Summaries
[52x725]clstk

Next, we describe different logical components of clstk, which correspond to the code structure. Section 3.1 describes the core part of clstk which glues different components together. Section 3.2 describes the component responsible for evaluation of cross-lingual summaries. Sections 3.3, 3.4 and 3.5 describe automatic translation, sentence simplification and translation quality estimation modules respectively. At the end, Section 3.6 describes different summarization algorithms included in the system. We have uploaded a demo video of clstk at https://youtu.be/SdGZ_Ns6Rqs.

3 COMPONENTS

Next, we describe different logical components of clstk, which correspond to the code structure. Section 3.1 describes the core part of clstk which glues different components together. Section 3.2 describes the component responsible for evaluation of cross-lingual summaries. Sections 3.3, 3.4 and 3.5 describe automatic translation, sentence simplification and translation quality estimation modules respectively. At the end, Section 3.6 describes different summarization algorithms included in the system. We have uploaded a demo video of clstk at https://youtu.be/SdGZ_Ns6Rqs.

3.1 The Core

The core contains the bootstrap code for summarization needs. The core provides: (1) A common standard structure for documents and summaries to ensure interoperability between different components. (2) Utilities for loading document sets into the common structure. (3) Common utilities on document sets, documents and sentences, for example, sentence splitting, tokenization, etc. The core also provides command-line access to different summarizers included in the tool-kit.

3.2 Evaluation

This module contains different utilities for evaluation of generated summaries. ROUGE score is a widely used metric for automated evaluation of summaries. The tool contains a Python implementation of ROUGE. Additionally, the code integrates with the original ROUGE package [7]. We recommend a Unicode-aware version of the original ROUGE-1.5.5 package [8] for evaluation of cross-lingual summaries as the target language may need Unicode character set.

Additionally, a command-line script for evaluation is provided which runs a selected summarization method with given parameters for all document sets in a directory and reports the ROUGE score given the reference summaries. This is quite useful when trying out multiple methods to get comparative scores.

3.3 Translation

Translation is an important module when working with CLS. clstk includes a translation module to easily incorporate machine translation in the CLS process. The module is designed keeping in mind that different methods may use translation at different stages and in different contexts. For example, the module allows for translating documents before summarization or after summarization or selectively translating sentences while summarizing. There are a large number of machine translation (MT) systems available both commercially and non-commercially. We acknowledge the need of use of particular MT systems based on various reasons. Hence, the module is designed to allow easy integration with various third-party MT tools and APIs. Currently, clstk contains integration with the Google Translate API [9].

3.4 Sentence Simplification

Using sentence simplification can help in obtaining better translation [4, 12] for CLS. Hence, we include a sentence simplification module in clstk. Similar to the translation module, the sentence simplification module also allows integration with third-party tools and APIs. Currently, clstk contains integration with the Neural Text Simplification system [11].

3.5 Translation Quality Estimation

The use of Translation Quality as a measure while extracting sentences for CLS has been explored in the past [1, 14]. We include a Translation Quality Estimation (QE) module in clstk which can be used to experiment with QE scores in different contexts and at different stages while summarizing. Currently, clstk integrates with the QE system [10] published by Jhaveri et al. [6], which contains implementation of several state-of-the-art models for QE.

3.6 Summarizers

One of the major goals of the tool-kit is to make available multiple approaches and methods for CLS. The tool currently contains implementations of two models for CLS by Wan [13]. Additionally, the tool contains an implementation of the popular sub-modular function maximization based summarization algorithm [9], and adapts it for use in the cross-lingual setting. The major reasons for

2http://www.summarization.com/mead/
3http://www.taln.upf.edu/pages/summa.upf/documentation.html
4https://github.com/ceteri/pytextrank
5https://github.com/Mojololo/textteaser
6https://github.com/tensorflow/models/tree/master/research/textsum
7https://radimrehurek.com/gensim/summarization/summariser.html
8https://github.com/nisargjhaveri/ROUGE-1.5.5-unicode
9https://translate.google.com/
10https://github.com/nisargjhaveri/tqe
choosing these methods for implementation are that these systems are (1) not resource intensive, (2) popular, and (3) highly accurate.

3.6.1 SimFusion. This method, proposed by Wan [13], uses English-side information along with Chinese-side information for Chinese sentence ranking in a graph-based framework. All the sentences in the source documents are first translated automatically from source language (English in their case) to the target language (Chinese in their case). The sentence similarities are computed for sentence pairs in both the languages and are fused to get final similarity score for a sentence pair in target language. The intuition behind the method is that information from single side is not very reliable in a CLS setting. Finally, using the similarity scores for target sentence pairs, a graph is created where each node is a sentence. PageRank-kind computations on this graph are then used to extract summary sentences.

3.6.2 CoRank. Similar to SimFusion method, Wan [13] also proposed the CoRank method to leverage information from both languages. It assumes that a sentence would be salient if it is heavily linked with other salient sentences in the same language as well as if it is heavily linked with salient sentences from the other language. In this method, the source sentences and the automatically translated sentences in target language are ranked simultaneously using a unified graph-based algorithm.

3.6.3 LinBilmes. clstk also supports the summarization framework and sub-modular functions described by Lin and Bilmes [8] and Lin and Bilmes [9]. We adapt the system for CLS by adding different options for translation and simplification at different steps. Additionally, we implement a new sub-modular objective function for sentence-level translation quality given by the QE module along with the objective functions for coverage and diversity.

The three summarizers already implemented act as examples of the intended use of the system. Other summarizers can be easily integrated into the tool.

4 DATA

This section describes two datasets we use to demonstrate the usage of the system. Section 4.1 describes a new manually annotated CLS dataset, where the raw data is derived from the DUC 2004 dataset. Section 4.2 describes a dataset published as part of TAC 2011 language independent multi-document summarization task, which we use to evaluate CLS.

4.1 DUC 2004 Gujarati

Along with the system we publish a dataset for English to Gujarati CLS11. The dataset was created by manually translating all summaries of all 50 document sets from DUC 2004 multi-document summarization dataset to Gujarati using a custom annotation tool we designed for CLS [5].

The annotation tool was configured to allow annotators to edit translations only. Automatic translations from Google Translate were provided as a reference. The translators were told to make the translations of summaries as natural as possible in Gujarati and not to minimize edits. Five native Gujarati speakers translated the summaries to Gujarati.

The dataset now contains source documents in English from the original DUC 2004 corpus, and summaries in Gujarati. We use this dataset to demonstrate the usage of the system and show comparative results of some of the included algorithms and configurations.

4.2 TAC 2011 MultiLing Pilot Dataset

MultiLing Pilot 2011 dataset12 was published as part of TAC 2011 Summarization Track, for language independent or multi-lingual summarization task [3]. The data was prepared by sentence-by-sentence translation of document sets from English to six other languages: Arabic, Czech, French, Greek, Hebrew and Hindi. The model summaries were created by fluent speakers (generally, native speakers) of each corresponding language. As a result the dataset contains parallel documents in all seven languages and their respective summaries. The dataset contains ten document sets in seven languages, and three summaries for each document set.

We use the dataset in a CLS setting where summaries are generated in target language for the source documents in source language. The target language summaries are then evaluated using the summaries available in the dataset. Here, the source and target languages could be any two different languages selected from the set of seven languages for which the data is available.

5 EXPERIMENTS AND RESULTS

We demonstrate the use of the system for English to Gujarati summarization using the proposed DUC 2004 Gujarati dataset described in Section 4.1 and for English to Hindi summarization using the MultiLing Pilot 2011 dataset described in Section 4.2.

<table>
<thead>
<tr>
<th>Model</th>
<th>ROUGE-1</th>
<th>ROUGE-2</th>
<th>Perplexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoRank</td>
<td>21.52</td>
<td>21.17</td>
<td>3.92</td>
</tr>
<tr>
<td>CoRank.earlySimplify</td>
<td>21.35</td>
<td>20.96</td>
<td>3.77</td>
</tr>
<tr>
<td>SimFusion</td>
<td>22.92</td>
<td>22.08</td>
<td>4.06</td>
</tr>
<tr>
<td>SimFusion.earlySimplify</td>
<td>22.71</td>
<td>22.45</td>
<td>4.13</td>
</tr>
<tr>
<td>LinBilmes</td>
<td>21.17</td>
<td>21.41</td>
<td>3.58</td>
</tr>
<tr>
<td>LinBilmes.earlySimplify</td>
<td>20.67</td>
<td>20.74</td>
<td>3.37</td>
</tr>
<tr>
<td>SimFusion.earlyTranslate</td>
<td>22.11</td>
<td>21.32</td>
<td>3.90</td>
</tr>
<tr>
<td>SimFusion.earlySimplify</td>
<td>22.71</td>
<td>22.45</td>
<td>4.13</td>
</tr>
</tbody>
</table>

Table 1: Evaluation of Different CLS Methods on DUC 2004 Gujarati Dataset

<table>
<thead>
<tr>
<th>Model</th>
<th>ROUGE-1</th>
<th>ROUGE-2</th>
<th>Perplexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoRank</td>
<td>36.02</td>
<td>36.30</td>
<td>7.04</td>
</tr>
<tr>
<td>CoRank.earlySimplify</td>
<td>37.72</td>
<td>37.64</td>
<td>7.48</td>
</tr>
<tr>
<td>SimFusion</td>
<td>37.75</td>
<td>37.76</td>
<td>6.80</td>
</tr>
<tr>
<td>SimFusion.earlySimplify</td>
<td>38.40</td>
<td>38.28</td>
<td>7.55</td>
</tr>
<tr>
<td>LinBilmes</td>
<td>36.48</td>
<td>36.48</td>
<td>6.90</td>
</tr>
<tr>
<td>LinBilmes.earlySimplify</td>
<td>36.48</td>
<td>36.48</td>
<td>6.90</td>
</tr>
</tbody>
</table>

Table 2: Evaluation of Different CLS Methods on MultiLing Pilot 2011 Dataset for Hindi

Tables 1 and 2 show the ROUGE scores and the perplexity of the summaries for DUC 2004 Gujarati and MultiLing Pilot 2011 Hindi dataset respectively for different methods included in clstk.


namely CoRank, SimFusion, and LinBilmes. The perplexity was calculated using the bi-gram language models trained on FIRE 2011 monolingual datasets\textsuperscript{13} for Gujarati and Hindi using kenlm\textsuperscript{14}. We also ran all three algorithms with early-simplification, in which, the source documents are first simplified by the sentence simplification module and then the respective methods are applied to generate the summaries. Additionally, we also ran LinBilmes with early-translation, in which the documents are translated first then the summarizer is run on the translated documents. The early-translation configuration is not included for the other two methods as they already use information from both the source and automatically translated sentences.

We note that, for our datasets, SimFusion works best for both of our datasets, contrary to the trend shown by Wan [13], which proposed and compared SimFusion and CoRank for English to Chinese summarization.

We also note that using sentence simplification helps in terms of readability. For this we assume that better perplexity on a language model trained on large corpora leads to better readability of naturalness of the sentences in the generated summary. Table 1 and Table 2 show that for both the datasets, applying early-simplification leads to better perplexity for all the three algorithms. This also justifies the integration of sentence simplification module in clstk.

Further, in Table 3, we show the average CLS run-time of various algorithms for the two datasets. The small execution times imply that the system is clearly practically usable.

### Table 3: Average Run-time per Document Set of Different CLS Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>DU. 2004 Gujarati</th>
<th>MultiLing Pilot 2011 Hindi</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoRank</td>
<td>0.49s</td>
<td>0.44s</td>
</tr>
<tr>
<td>SimFusion</td>
<td>0.49s</td>
<td>0.44s</td>
</tr>
<tr>
<td>LinBilmes</td>
<td>2.90s</td>
<td>2.06s</td>
</tr>
</tbody>
</table>

### Table 4: Evaluation of Different CLS Methods on MultiLing Pilot 2011 Dataset for Different Languages

<table>
<thead>
<tr>
<th>Language</th>
<th>Model</th>
<th>ROUGE-1</th>
<th>F-score</th>
<th>ROUGE-2</th>
<th>F-score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arabic</td>
<td>CoRank</td>
<td>29.03</td>
<td>29.75</td>
<td>8.90</td>
<td>9.12</td>
</tr>
<tr>
<td></td>
<td>SimFusion</td>
<td>29.81</td>
<td>30.49</td>
<td>8.80</td>
<td>8.99</td>
</tr>
<tr>
<td></td>
<td>LinBilmes</td>
<td>27.12</td>
<td>27.64</td>
<td>6.95</td>
<td>7.08</td>
</tr>
<tr>
<td>Czech</td>
<td>CoRank</td>
<td>31.55</td>
<td>31.49</td>
<td>7.34</td>
<td>7.35</td>
</tr>
<tr>
<td></td>
<td>SimFusion</td>
<td>33.24</td>
<td>33.16</td>
<td>8.14</td>
<td>8.13</td>
</tr>
<tr>
<td></td>
<td>LinBilmes</td>
<td>30.27</td>
<td>30.29</td>
<td>6.57</td>
<td>6.57</td>
</tr>
<tr>
<td>French</td>
<td>CoRank</td>
<td>47.75</td>
<td>48.84</td>
<td>13.78</td>
<td>13.47</td>
</tr>
<tr>
<td></td>
<td>SimFusion</td>
<td>48.88</td>
<td>48.00</td>
<td>15.88</td>
<td>15.60</td>
</tr>
<tr>
<td></td>
<td>LinBilmes</td>
<td>47.23</td>
<td>47.36</td>
<td>14.42</td>
<td>14.17</td>
</tr>
<tr>
<td>Greek</td>
<td>CoRank</td>
<td>34.88</td>
<td>34.71</td>
<td>8.50</td>
<td>8.50</td>
</tr>
<tr>
<td></td>
<td>SimFusion</td>
<td>36.23</td>
<td>36.16</td>
<td>8.45</td>
<td>8.45</td>
</tr>
<tr>
<td></td>
<td>LinBilmes</td>
<td>34.61</td>
<td>34.59</td>
<td>7.62</td>
<td>7.62</td>
</tr>
<tr>
<td>Hebrew</td>
<td>CoRank</td>
<td>21.87</td>
<td>22.06</td>
<td>4.40</td>
<td>4.44</td>
</tr>
<tr>
<td></td>
<td>SimFusion</td>
<td>22.69</td>
<td>22.95</td>
<td>4.59</td>
<td>4.66</td>
</tr>
<tr>
<td></td>
<td>LinBilmes</td>
<td>20.65</td>
<td>21.00</td>
<td>4.00</td>
<td>4.08</td>
</tr>
</tbody>
</table>

Finally, we also experiment with the other five languages which are a part of the MultiLing 2011 Dataset. Table 4 shows results obtained using various methods in terms of ROUGE scores. Note that for most datasets, SimFusion works best, while CoRank performs better in a few cases.

### 6 COMPANION TOOLS AND ROADMAP

Our CLS workbench [5] can be used as a companion tool with this system in the development of new methods for the task. This workbench can be plugged into clstk, to help rapidly generate CLS data for new language pairs, and later the data can be used to improve or implement new methods in clstk.

In future, we plan to include more existing CLS methods in the tool and encourage the community to contribute and use the tool-kit. We plan to provide an easy web-interface for the implemented methods to enable visitors to try out different CLS methods easily.

### 7 CONCLUSIONS

We contribute the only available tool-kit for CLS, clstk, containing different existing methods as well as bootstrap code to easily develop and experiment with new methods for the same. We also propose a new dataset for cross-lingual summarization evaluation, along with an annotation tool custom designed for CLS. We show preliminary experiments and comparative results on two datasets for different methods and configurations for cross-lingual summarization.

### REFERENCES