International Journal of Applied Mathematics, Electronics and Computers 8(4): 179-184, 2020

IJAMEC

e-ISSN: 2147-8228

INTERNATIONAL JOURNAL OF APPLIED MATHEMATICS ELECTRONICS AND COMPUTERS

www.dergipark.org.tr/ijamec

International Open Access

Volume 08
Issue 04

December, 2020

*Research Article*

# Extractive Text Summarization System for News Texts

***Fahrettin Horasan*** [a,*], ***Burhan Bilen*** [a]

[a] *Kirikkale University, Computer Engineering, Yahsihan Campus, Kirikkale 71450, Turkey*

ARTICLE INFO

*Article history:*
Received 28 September 2020
Accepted 5 November 2020

*Keywords:*
Automatic text summarization
Data processing
Evaluation
Feature extraction
Sentence scoring

ABSTRACT

In today's conditions, it is difficult to obtain information quickly and efficiently due to the size of the data. There are various text documents on the internet and a good extraction algorithm is essential to have the most relevant information from them. Long texts can be boring sometimes. So, readers are eager to get the main idea of the text or any useful information. For this reason, the importance of automatic summarization systems is understood. Text summarization systems can be considered as abstractive summarization or extractive summarization. While abstractive systems produce a summary with new sentences, extractive systems make a selection of sentences from the text used and combine them and present them as a summary. Creating a successful summarization algorithm increases in direct proportion to the success of applying text mining techniques. Text summary systems provide a summary of the text to the user by scoring words and sentences in the main text using various methods and combining high ranked sentences as a result of the process. In this context, many scoring methods have been used. In our study, news data sets are used. The algorithm used is based on extraction and has been evaluated using a task-independent method. After evaluation, the two highest scores taken are ROUGE-1 with 0.68 score and ROUGE-S with 0.54 score. Through all evaluation steps, Precision, Recall and F-Measure values are also specified to see the steps clearly.

## 1. Introduction

Writing is one of the main ways of communication. There are many types of writing due to the large number of languages spoken around the world. Every day, thousands of texts are being written by people worldwide. To get to know the main idea and the basic information that those texts bring to the readers, it is a must to read them completely. Many people wonder how texts include the wanted or important information through sentences, or in short, how much of those sentences the texts have to deliver the main theme. Sometimes, in any text, the amount of unwanted sentences may get more than the amount of wanted information. In that way, just as we've said, readers need to read the entire text just to find out what the post is about or what is meant to be told, which makes readers waste lots of times. To have the main theme or wanted information from texts and solve these problems, the automatic text summarization topic has got much attention.

Automatic summarization systems let readers get the main theme of texts with ease. Thanks to that, they get rid of the redundant data. So, reading time is decreased, text importance is increased. Thus, information access is easier.

Automatic summarization systems can be represented as extractive or abstractive systems depending on the application ways and methods. This work has focused on information extraction, because abstractive summarization systems are still performing lower performance and often having less exact information than extractive summarization systems. This is because extractive systems choose informations directly from the main text, instead of trying to have new informations by itself. Even though the extractive summarization systems are not perfect, they give readers an opinion about original text [4]. Besides, there are many measurement

[*] Corresponding author. E-mail address: *fhorasan@kku.edu.tr*

DOI: 10.18100/ijamec.800905

methods used to evaluate the performance of automatic summarization systems on the datasets used. These evaluation methods can be examined under two titles as task-independent and task-based methods. Task-independent methods are based on an expert opinion summary (ideal/reference summary). Task-based evaluation methods do not analyse the sentences in the summary. Main goal is to analyse the possibility of a summary usage for a specific task. There are many approaches to task-based evaluation. The three most important tasks are categorization, information retrieval and question answering [4].

Additionally, automatic summarization systems can be used as single or multiple document summarization.

This work has evaluations based on multiple document summarization and task-independent method. The algorithm will be evaluated according to ROUGE automated evaluation metrics.

For testing the algorithm, five dataset categories of news were used and amount of documents in categories are:

| | |
|---|---|
| Business: | 510 |
| Entertainment: | 386 |
| Politics: | 417 |
| Sport: | 511 |
| Technology: | 401 |

One of the most important topics that will be needed in this and similar studies is text mining.

The dataset can be found here [17]. It also contains ideal summaries which are hard to obtain, to be able to evaluate the system easily.

### 1.1. Text Mining

Text mining is a data mining method and makes information exploration from raw texts possible. It's mostly used for finding documents related to each other and exploring relationships between concepts [12]. Text Mining is a data analysis method that makes it possible to obtain information from existing data with statistics, machine learning, database systems or similar subjects. As in this article, word or phrase extraction, feature extraction or data preprocessing are examples of text mining. It can be used to extract information from large data, summarize or similarity calculations. Text mining reduces the cost of time and resources. It generally consists of six steps. These are:

#### 1.1.1. Data Acquisition

The first stage of text or data mining is to obtain information [15].

Data sources suitable for the project can be obtained from an online or offline source. In addition, having an expert summary package will let researchers to evaluate the system.

#### 1.1.2. Preprocessing Phase

While the data is being obtained, they may often contain unwanted characters or have an incorrect data order. This situation can cause an unacceptable issue. For example, in a sentiment analysis study, when microtexts were replaced with their originals, a ~ 4% performance improvement was achieved [6]. Also, data can be preprocessed using methods such as lowercase conversion, space removal, punctuation mark deletion, character replacement, minimum word length elimination, ineffective word elimination, stems [14], lemmatization or more.

#### 1.1.3. Feature Extraction

This is the phase where the raw dataset is reduced to more controllable pieces to process well. In short, it is the main title of the methods that accurately and completely describe the original data set while reducing the amount of data that needs to be processed.

#### 1.1.4. Data Mining

It is the stage of translation of unprocessed data to useful information. It's based on data collection, storage and mathematical processing.

Data mining is an important phase of extraction of data patterns where various methods used. The aim is to find the relationship between the groups of knowledge that reveal the points and to make researchers able to discover new information that is difficult to obtain [15].

#### 1.1.5. Data Visualization

It is the presentation of the values obtained as a result of a series of processes to the user in a design way, such as a graphic.

#### 1.1.6. Evaluation

In data mining, the evaluation of the results is provided by precision, recall and the f-score which depends on precision and recall values. The formulas are as following [16]:

$$\text{Precision} = \frac{X}{Y} \tag{1}$$

$$\text{Recall} = \frac{X}{Z} \tag{2}$$

$$F - \text{Score} = \frac{2 * Precision * Recall}{Precision + Recall} \tag{3}$$

x = Number of matching sentences in the reference summary and the system summary

y = Amount of the sentences in the system summary

z = Amount of the sentences in the reference summary

*x, y and z value sources may vary according to the project.

## 2. Related Works

The first automatic summation system was developed by Luhn in 1958 based on term frequency. Automatic text summarization system in 1969 by Edmundson, has used some standard keyword methods from before such as word frequency, cue words, title and positioning to assign sentence weights. The Trainable Document Summarizer carried out sentence extraction which is weight based heuristic, in 1995. The machine learning techniques in natural language processing have utilized statistical techniques to create file summaries in 1990s [2].

Zemberek, one of the studies on Turkish Language, is a resource that many researches who work on natural language processing is commonly examined and used it for many times. Zemberek is an open source Turkish natural language processing library. It can be used for functions such as finding word roots, special names in texts, etc. in natural language processing. The second version, Zemberek2, which is currently published, can be used for Turkic languages [11] [13].

Autotext Summarization is used today by platforms such as search marketing, search engines, news websites, bots, and social media marketing. Google Infographics and Bing News Snippets are known examples of automatic text summarization.

Automatic Text Summarization is examined under two main titles: Extractive Text Summarization and Abstractive Text Summarization.

### *2.1. Extractive Text Summarization*

Extractive summarization is based on the method of sentence weighting by obtaining the words and phrases in the text with their frequencies. It does select the sentences with highest score from the document and removes the rest of the useless sentences. It uses many methods while scoring. It also uses automated methods (e.g. ROUGE) for algorithm evaluation.

### *2.2. Abstractive Text Summarization*

Abstractive summarization works differently than extractive text summarization. Interpreting the text and then creating a new and shorter summary text that differs from the original text. Similarly, ROUGE or different evaluation methods can be used for algorithm evaluation. It is more difficult to implement than extractive summarization. Although the accuracy rate of the obtained results is lower than the extractive method, the results are more similar to human-like summaries.

## 3. Pseudo Code

```
text = get(text path)
sentences = tokenize(text)
words = tokenize(text)
word frequency = [ ]
sentence score = [ ]
while:
    if not a stop word:
        word frequency[i] += 1
    else:
        word frequency[i] = 1
while:
    sentence score[sentence] += scoring method n
While:
    If sentence score[i] > average score:
        system summary += sentences[sentence]
reference summary = get(summary path)
eval. method n(reference and system summary)
```

## 4. Text Summarization

### *4.1. Data Acquisition and Preprocessing*

There are two types of processing methods: singular and multiple file/data processing. Regardless of the processing method, the data whose summary is requested must be parsed into words and sentences.

### *4.2. Creating a Frequency Table*

Separating all the words by their roots and placing them in a table with their number of occurrences in the whole text.

### *4.3. Sentence Scoring*

A table is needed to keep the sentence scores in it. It can be trimmed to have a good rating. For example, only the first twenty characters of all sentences can be processed for equality. The scoring process continues depending on the word frequency table. Scoring can be done with various methods.

#### *4.3.1. Term Frequency*

The term frequency algorithm is used for weighting the sentences in the scoring process.

$$tf(x,y) = \frac{frequency\ of\ term\ x\ in\ document\ y}{number\ of\ the\ words\ in\ document\ y} \quad (4)$$

#### *4.3.2. Term Weighting*

Performing the division of the term frequencies of all sentences over the highest term frequency score in the document represents the Term Weighting method [2].

$$tw = \frac{\sum tf_x}{Max.\sum tf_x} \quad (5)$$

#### *4.3.3. Numerical Data*

Finding numerical data within sentences is one of the useful ways to understand sentence significance. Numerical data calculation is done as follows [7]:

$$nd = \frac{numerical\ data\ in\ document\ x}{length\ of\ the\ sentence\ x} \tag{6}$$

### *4.3.4. Sentence Length*

The importance of sentences may increase depending on the length of the sentences. It’s calculated as follows [7]:

$$sl = \frac{number\ of\ the\ words\ in\ document\ x}{num.\ of\ the\ words\ in\ the\ longest\ sentence} \tag{7}$$

### *4.3.5. Proper Nouns*

Quantity of proper nouns can identify dominant sentences in the document.
Its value is computed as follows [2]:

$$pn = \frac{number\ of\ proper\ nouns\ in\ sentence\ x}{lenght\ of\ the\ sentence\ x} \tag{8}$$

### *4.3.6. Sentence Location*

Location of sentences are also important to see their importance. For the following formula, N is the number of sentences and $P_i$ is the location of the sentence, in the document. Sentence location value is computed as follows [5]:

$$slo = \frac{N-P_i}{N} \tag{9}$$

### *4.3.7. Sentence Similarity*

The next step will be scoring the sentences according to similarity to the first and last sentences in the text. The cosine similarity formula is needed for similarity computation. The cosine similarity is computed as below [5]:

$$\cos(x,y) = \frac{X^{\rightarrow} \cdot Y^{\rightarrow}}{\|X^{\rightarrow}\|\ \|Y^{\rightarrow}\|} \tag{10}$$

To calculate the similarity of current sentence to the first sentence, the variables are selected as follows; x is equal to the current sentence, y is equal to the first sentence of the text. Likewise, for the resemblance to last sentence, the variables should be as follows; x is equal to the current sentence and y is equal to the last sentence of the text.

Cosine Similarity requires vector forms of texts. There are some practical models (i.e. Bag of Words) that you can use to convert text to vectors.

You can check out this article for more scoring methods [1].

## 5. Evaluation

There are three evaluation methods in the test part of the study: ROUGE-N, ROUGE-L and ROUGE-S. The following texts explain the methods for readers.

### *5.1. ROUGE-N*

In the DUC organization, ROUGE (Recall-Oriented Understudy for Gisting Evaluation) was used as automated evaluation method. N-gram based ROUGE evaluation measurements package was first introduced in 2003 [8].

For example, ROUGE-1 represents the number of matches of unigrams between the system summary and the reference summary. ROUGE-2 stands for the number of matches of the bigrams between the system summary and the reference summary [9].

To understand ROUGE-N, we must first know what N-Gram is. Suppose that we have a sentence; "Data science is very important". If we split the given sentence according to the N-gram models;

*According to unigram (N=1) model, it gets:*
*[“Data”, “science”, “is”, “very”, “important”]*
*According to bigram(N=2) model, it gets:*
*[“Data science”, “science is”, “is very”, “very important”]*
*According to trigram (N=3) model, it gets:*
*[“Data science is”, “science is very”, “is very important”]*
*According to fourgram (N=4) model, it gets:*
*[“Data science is very”, “science is very important”]*

The ROUGE-N evaluation for a system summary is performed as below:

$$\text{ROUGE} - \text{N} = \frac{\begin{array}{c} number\ of\ overlapping \\ n{-}grams\ between\ summaries \end{array}}{\begin{array}{c} number\ of\ n{-}grams\ in \\ the\ reference\ summary \end{array}} \tag{11}$$

### *5.2. ROUGE-L*

For the ROUGE-L evaluation, the longest common subsequence (LCS) between the system summary and the reference must be found.

One of the advantages of LCS usage is that it does not necessitate consecutive matches like other n-gram models. A pre-prepared n-gram model is not required, because it automatically contains the longest common n-grams [9]. ROUGE-L test score of a system summary is computed as follows:

$$Precision_{lcs} = \frac{Length(LCS(system\ sum,\ reference\ sum))}{m} \tag{12}$$

*m: length of the system summary*

$$Recall_{lcs} = \frac{Length(LCS(system\ sum,\ reference\ sum))}{n} \tag{13}$$

*n: length of the reference summary*

Once the precision and recall values are found, the f-score value (ROUGE-L) is calculated as follows:

$$F - Score_{lcs} = \frac{(\beta^2+1)Precision_{lcs}Recall_{lcs}}{Recall_{lcs}+Precision_{lcs}\beta^2} \tag{14}$$

$$\beta = \frac{Precision_{lcs}}{Recall_{lcs}} \tag{15}$$

### *5.3. ROUGE-S*

Skip-bigram is a model that any word pair in the sentence order that allows arbitrary spaces. Skip-bigram co-occurrence statistics measure the matching bigrams between system summary and reference summary [3]. For the model calculations, you can optionally use formulas or functions in the NLTK library [10].

After all skip-bigrams in the system summary and the reference summary are found, it is necessary to find the matching objects and their matching quantities in order to continue the evaluation process. ROUGE-S test score is calculated as follows [3]:

$$Precision_{skip2} = \frac{SkipBigram(x,y)}{C(m,2)} \quad (16)$$

*m: length of system summary*

To find Precision, SkipBigram(x, y) is divided by Combination of m and 2.

$$Recall_{skip2} = \frac{SkipBigram(x,y)}{C(n,2)} \quad (17)$$

*n: length of reference summary*

To find Recall, SkipBigram(x, y) is divided by Combination of n and 2.

The last step is to find the F-Score (ROUGE-S) value. It is calculated as follows [3]:

$$F - Score_{skip2} = \frac{(\beta^2+1)Precision_{skip2}Recall_{skip2}}{Recall_{skip2}+Precision_{skip2}\beta^2} \quad (18)$$

$$\beta = \frac{Precision_{skip2}}{Recall_{skip2}} \quad (19)$$

## 6. Test Results

As mentioned before; Business(B) category has 510, Entertainment(E) has 386, Politics(P) has 417, Sports(S) has 511 and Technology(T) has 401 documents. ROUGE based test result table and graph:

**Table 1.** ROUGE based test results

| | rouge-1 | rouge-2 | rouge-3 | rouge-4 | rouge-l | rouge-s |
|---|---|---|---|---|---|---|
| B | 0.705675 | 0.481987 | 0.433117 | 0.407368 | 0.509271 | 0.516626 |
| E | 0.673612 | 0.454693 | 0.407432 | 0.382049 | 0.503325 | 0.520260 |
| P | 0.691478 | 0.479287 | 0.428719 | 0.402299 | 0.496871 | 0.547352 |
| S | 0.628446 | 0.419015 | 0.369324 | 0.344946 | 0.467810 | 0.575610 |
| T | 0.700808 | 0.473677 | 0.420522 | 0.392563 | 0.497099 | 0.544775 |

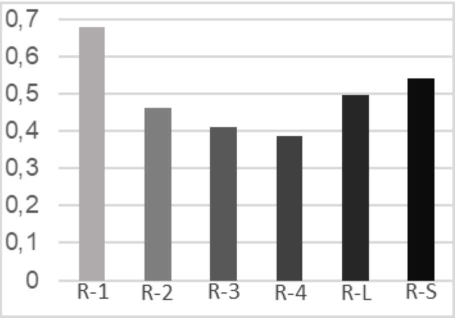

**Figure 1.** Avg. Results Pivot Chart

The ROUGE-N (N: 1, 2, 3, 4,…) metric requires the use of the N-Gram algorithm to calculate the result. The ROUGE-L metric requires finding the longest common substring in order to calculate the result. Also, the ROUGE-S metric requires the use of the Skip-bigram algorithm.

You can check Lin's article for detailed evaluation steps. [3]

## 7. Conclusion

Automatic extractive text summarization work carried out the steps described above. According to the evaluation result table, we can say that when N increases, the ROUGE-N test result decreases. The reason for the drop is the N-Gram algorithm. When N increases, the N-Gram algorithm returns fewer list objects and the denominator of the ROUGE-N formula does not change. In this way, the division ratio of values is reduced.

Aside, this work can be integrated to search engines to gather summaries of news or any text. Also it can be used to retrieve information from long and important data groups to make the time cost lower.

This paper contains simplified formulas and process steps to be helping to understand and apply on easily for anyone who wants to work on extractive text summarizaton topic.

## 8. Future Works

This article is about how the Extractive Text Summarization model works. We hope that this work helps and inspire anyone who reads it. For future projects, along with the methods used in the study, other articles can be used to examine and obtain new features. Also, the Hidden Semantic Analysis method can be used to see new and different results. In fact, new metrics can be added to get more evaluation results. Moreover, it would be good to work on the Abstractive Text Summarization project or to developq a model with deep learning techniques.

## References

[1] T. Sri Rama Raju and Bhargav Allarpu, *Text Summarization using Sentence Scoring Method*. April 2017. Volume: 04 Issue: 04 | pages 1777-1779

[2] S.A. Babar and Pallavi D. Patil, *Improving Performance of Text Summarization*. Procedia Computer Science 46, 2015. 354 – 363, (ICICT 2014)

[3] Lin, C.Y., *ROUGE: A Package for Automatic Evaluation of Summaries*. Spain, In Proceedings of the Workshop on Text Summarization Branches Out, 25 – 26 July 2004.

[4] Josef Steinberger and Karel Ježek, *Evaluatıon Measures For Text Summarızation*. Computing and Informatics, March 2009, Vol. 28, 2009, 1001–1026.

[5] Aysun Güran, *Otomatik Metin Özetleme Sistemi*. Yıldız Teknik Üniversitesi, Fen Bilimleri Enstitüsü, 2013

[6] R. Satapathy and C. Guerreiro and I. Chaturvedi and E. Cambria, *Phonetic-Based Microtext Normalization for Twitter Sentiment Analysis.* IEEE International Conference on Data Mining Workshops (ICDMW), New Orleans, 2017, pp. 407-413, doi: 10.1109/ICDMW.2017.59

[7] Kaynar, O. and Işık, Y.E and Görmez, Y. and Demirkoparan F., *Genetic Algorithm Based Sentence Extraction For Automatic Text Summarization.* Yönetim Bilişim Sistemleri Dergisi, 2017, Cilt:3, Sayı:2, Sayfa:62-75, ISSN: 2148-3752

[8] Lin, Ch. and Hovy, E., *Automatic Evaluation of Summaries Using n-Gram Co-Occurrence Statistics.* Canada, In Proceedings of HLT-NAACL, 2003.

[9] https://www.ccs.neu.edu/home/vip/teach/DMcourse/5_topic model_summ/notes_slides/What-is-ROUGE.pdf

[10] https://github.com/nltk/nltk/blob/develop/nltk/util.py#L53

[11] Gündoğdu, Ö.E. and Duru, N., *Türkçe Metin Özetlemede Kullanılan Yöntemler*. Aydın, 18. Akademik Bilişim Konferansı, , 30 Ocak-5 Şubat 2016, Adnan Menderes Üniversitesi.

[12] P. Yıldırım and M. Uludağ and A. Görür, *Hastane Bilgi Sistemlerinde Veri Madenciliği.* Çanakkale, Akademik Bilişim, Ocak 2008, Çanakkale Onsekiz Mart Üniversitesi.

[13] A.A. Akın and M.D. Akın, *Zemberek, An Open Source Nlp Framework For Turkic Languages*. 2007, Structure 10, 1-5, 185.

[14] K. Deniz and B. Fatma and O. Akin and Y. Fatih and B. Emin, *Metin Madenciliği Kullanılarak Yazılım Kullanımına Dair Bulguların Elde Edilmesi.* 2015.

[15] S. Çelik, *Metin Madenciliği ile Shakespeare Külliyatının İncelenmesi*. MANAS Sosyal Araştırmalar Dergisi, *9*(3), 1343-1357.

[16] Moratanch, N. and S. Chitrakala, *A Survey On Extractive Text Summarization*. Chennai, ICCCSP, 2017, 1-6. 10.1109/ICCCSP.2017.7944061.

[17] https://www.kaggle.com/pariza/bbc-news-summary