ROMANIAN JOURNAL OF INFORMATION SCIENCE AND TECHNOLOGY
Volume **25**, Number 2, 2022, 117–132

The Impact of Features and Preprocessing on Automatic Text Summarization

Salih BAL[1] and Efnan SORA GUNAL[2, *]

[1]Dept. of Electrical and Electronics Engineering, Eskisehir Osmangazi University, Eskisehir, Turkiye
[2]Dept. of Computer Engineering, Eskisehir Osmangazi University, Eskisehir, Turkiye
Email: salihbal26@gmail.com, esora@ogu.edu.tr*
* Corresponding author

Abstract. Automatic text summarization obtains a shortened and informative version of a given text without manual intervention based on specific features, preprocessing methods, and decision mechanisms. This paper aims to thoroughly analyze the impact of common features and preprocessing techniques on the performance of automatic text summarization, particularly in the Turkish language. Also, a new distinctive feature based on latent semantic analysis is proposed as another contribution. Two datasets consisting of a total of 120 documents and 1,466 sentences were used for the analysis. Two different success metrics were utilized to assess the performance of automatic text summarization. A set of comprehensive experimental studies revealed the optimal feature subset and the most useful preprocessing methods that can improve the summarization performance. Moreover, it has been verified that the proposed feature further improves the performance.

Key-words: Computational linguistics, automatic text summarization, feature extraction, feature selection, machine learning, preprocessing.

1. Introduction

As the number of electronic documents increases exponentially, extracting relevant information from those documents becomes an even more challenging task. People have to deal with hundreds or even thousands of documents on the internet when they are searching for any topic. Therefore, automatic text summarization systems have become very popular to be able to retrieve brief information on a particular topic. These systems allow for quick access to fundamental information and to learn what any document is about.

Automatic text summarization is mainly divided into two categories: extractive summarization and abstractive summarization [1, 2]. In abstractive summarization, natural language generation methods are used and the input text is reconstructed without affecting the integrity of the content and coherence of sentences. On the other hand, extractive summarization selects a subset of sentences from the input text based on certain selection criteria and constructs a summary with the selected sentences without modifying them. Since abstractive summarization is more complex and impractical than extractive summarization [2–5], the majority of automatic text summarization studies have preferred extractive approaches to bring out a summary [1, 2, 6–8].

Since extractive text summarization attracts more attention in the literature, this paper aims to thoroughly analyze the impact of common features and preprocessing techniques on the performance of extractive summarization, particularly in the Turkish language considering the limited number of related studies in the literature for Turkish. Also, a new distinctive feature based on latent semantic analysis (LSA) is proposed as another contribution. Two datasets consisting of 120 documents and 1,466 sentences were used in the experimental work. Two different success metrics were utilized to assess the performance of automatic text summarization. A set of comprehensive experimental studies revealed the optimal feature subset and the most useful preprocessing methods that can improve the summarization performance. Moreover, it has been verified that the proposed feature further improves the performance.

The rest of this paper is organized as follows: the related work on automatic text summarization is discussed in Section 2. The preprocessing methods and common features used in text summarization are described in Section 3. The proposed feature is introduced in the same section, as well. The experimental work evaluating the impact of the features and preprocessing methods is given in Section 4. Finally, the conclusions and possible future works are provided in Section 5.

2. Related Work

In the literature, many models are proposed for automatic text summarization. As mentioned earlier, most of them are based on the extractive approach in which important sentences are selected from an input text to obtain the summary.

Altan et al. proposed a text summarization method for Turkish texts [9]. In that study, term frequency and sentence location were used as the features for the summarization process. Steinberger used cross methods in his approach to select important sentences [10]. Another study utilizing fuzzy logic was proposed in [6]. Tokenization, stop-word removal, and stemming were used for preprocessing. They implemented the automatic text summarization system in the Android platform for English documents. Tardan et al. analyzed the text summarization system for documents in the Indonesian Language using a semantic analysis approach [11]. The Indonesian version of WordNet was a significant part of their study. They utilized stemming and stop-word removal in the preprocessing stage. Cigir et al. proposed a Turkish text summarization system based on a sentence scoring approach [12]. These scores were calculated using several features such as term frequency, keyphrase, centrality, sentence position, and title similarity. The weights of features were adjusted using machine learning techniques. Stemming was applied as a preprocessing task. Hingu et al. proposed two methods adjusting the frequency of words by using their stem forms and synonyms [13]. The sentences included in references and citations were assigned a higher weight for selection. Geetha and Deepamala focused on the LSA algorithm for text summarization in Kannada [14]. They utilized two preprocessing methods including the detection of sentence boundary and stop-word removal. LSA was utilized in [15]. As preprocessing steps, they used part-of-speech (POS) tagging, stop-word removal, and stemming. They implemented the automatic text summarization system in the Android platform for English documents. Yadav and Meeana used the bushy path method to obtain the relation of sentences to the topic [16]. In the study, the WordNet synonyms method [17] was used to attain the semantics of the text. Also, a fuzzy logic-based method was employed to evaluate the score of each sentence. They removed stop words and special symbols in the preprocessing stage. A sentence-level semantic graph model was proposed in [18]. The authors used semantic analysis to estimate the relevance values between sentences. These values were assigned as the weights of edges. A variant of the traditional PageRank graph ranking algorithm was used to calculate the values of sentences. Guran et al. proposed an automatic text summarization system that was based on the fuzzy analytical hierarchy process (FAHP) [19]. The utilized preprocessing methods were stemming and stop word removal. In reference [1], the syntactic feature space was obtained for the utilized dataset. Also, GloVe and Word2Vec embeddings were used for the capability of semantic features for text summarization. Besides, a long short-term memory-based deep neural network model, which includes the joint use of syntactic and semantic features, was proposed. Hark and Karci presented a tool for text processing to provide semantic structure between sentences [20]. Stop words were removed in the preprocessing step. Besides, an entropy measure was used for summarization. Two different datasets were used to evaluate the success of the proposed system. Belwal et al. performed the text summarization technique that comprises topic modeling and semantic measure within the vector space model [5]. The preprocessing steps were lemmatization, removing stop words, and punctuations. Epignosis and CNN / DailyMail datasets were used to evaluate the performance of the proposed method.

3. Materials and Methods

In this section, common preprocessing methods in automatic text summarization are briefly described. Widely used feature extraction approaches for text summarization are explained. Also, a novel feature extraction method is introduced.

3.1. Preprocessing

The use of preprocessing methods is quite common in not only text summarization but also any type of text processing or text mining application. Tokenization, lowercase conversion, stemming, and stop word removal are among the most common preprocessing methods [21–24].

Tokenization is the process of dividing a text into words, phrases, or other meaningful parts (i.e., tokens). Typically, the tokenization is handled by considering alphabetic or alphanumeric characters, which are delimited by non-alphanumeric characters such as spaces or punctuation marks. Lowercase conversion converts all uppercase characters in a given text into their lowercase forms for further processing. Stemming provides the stem or root forms of the derived words. The stemming algorithms are specific to the language being studied. Since our work is specific to the Turkish language, an open-source natural language processing library for Turkish was used for the stemming process [25]. Stop word removal aims to eliminate irrelevant words that do not carry distinctive information. Just like stemming, stop words are specific to the language being studied. A few examples of stop words for Turkish are listed in Table 1.

A common practice in most automatic text summarization studies is to apply all preprocessing methods mentioned above without thoroughly analyzing their contribution to the summarization performance. Therefore, in our work, all combinations of these four preprocessing methods

Table 1. Sample stop words for the Turkish language

acaba, ayrıca, bazı, beri, çünkü, da, eğer, gibi, herhangi, hiç, için, kadar, nasıl, ne, nereye, niçin, rağmen, sadece, sen, siz, şey, üzere, yani

were evaluated comparatively. In this way, it was aimed to reveal possible interactions between the preprocessing methods and to find out the best combination of the preprocessing tasks providing the highest automatic summarization performance. The combinations were obtained by considering tokenization as alphanumeric or alphabetic, lowercase conversion as enabled or disabled, stemming as enabled or disabled, and stopword removal as enabled or disabled. Hence, 16 different combinations are obtained as listed in Table 2.

Table 2. The combinations of the preprocessing methods

Combination	Tokenization	Lowercase Conversion	Stemming	Stop Word Removal
1	Alphanumeric	Disabled	Disabled	Disabled
2	Alphanumeric	Disabled	Disabled	Enabled
3	Alphanumeric	Disabled	Enabled	Disabled
4	Alphanumeric	Disabled	Enabled	Enabled
5	Alphanumeric	Enabled	Disabled	Disabled
6	Alphanumeric	Enabled	Disabled	Enabled
7	Alphanumeric	Enabled	Enabled	Disabled
8	Alphanumeric	Enabled	Enabled	Enabled
9	Alphabetic	Disabled	Disabled	Disabled
10	Alphabetic	Disabled	Disabled	Enabled
11	Alphabetic	Disabled	Enabled	Disabled
12	Alphabetic	Disabled	Enabled	Enabled
13	Alphabetic	Enabled	Disabled	Disabled
14	Alphabetic	Enabled	Disabled	Enabled
15	Alphabetic	Enabled	Enabled	Disabled
16	Alphabetic	Enabled	Enabled	Enabled

3.2. Common Features

Commonly used features in most studies on automatic text summarization are listed in Table 3 [10, 26–28]. In general, these features are first calculated for each sentence in a given document. Then, a scoring mechanism assigns a score to each sentence based on these features. Finally, the sentences with the highest scores are selected for the summary of the document. Each of these features is described in detail below.

Position (*f1*): is originated from the fact that the position of a sentence in a document specifies the relevancy of that sentence to the document summary. The position feature for a sentence (S_i) in a given document is calculated using

$$f_1\left(S_i\right) = \frac{N - P\left(S_i\right)}{N}, \tag{1}$$

where *N* is the total number of sentences and *P*(S_i) is the position of the sentence in the document.

Table 3. Common features for automatic text summarization

Feature	Description
f1	Position
f2	Length
f3	Similarity to the first sentence
f4	Similarity to the last sentence
f5	Similarity to the title
f6	Total sentence similarity
f7	Common similarity score
f8	Distributional feature
f9	Term frequency
f10	Inclusion of numeric characters
f11	Inclusion of "?" and "!"
f12	Inclusion of concluding words
f13	Inclusion of nouns
f14	Inclusion of named entities
f15	LSA score

Length (*f2*): corresponds to the number of words in a given sentence.

Similarity to the first sentence (*f3*): is the cosine similarity of the related sentence to the first sentence (S_{first}) in the document:

$$f_3\left(S_i\right) = cosine\left(S_i, S_{\text{first}}\right) \ . \tag{2}$$

Similarity to the last sentence (*f4*): is obtained by calculating the cosine similarity of the related sentence to the last sentence (S_{last}):

$$f_4\left(S_i\right) = cosine\left(S_i, S_{\text{last}}\right) \ . \tag{3}$$

Similarity to the title (*f5*): is obtained by calculating the cosine similarity of the related sentence to the title (T) of the document:

$$f_5\left(S_i\right) = cosine\left(S_i, T\right) \ . \tag{4}$$

Total sentence similarity (*f6*): is calculated as the sum of the cosine similarities of the related sentence to the remaining sentences in the document.

Common similarity score (*f7*): is calculated using

$$f_7\left(S_i\right) = \sum_{j=1}^{N} \text{similarity}\left(S_i,\ S_j\right) = \sum_{j=1}^{N} \frac{S_i\left(\text{similars}\right) \bigcap S_j\left(\text{similars}\right)}{S_i\left(\text{similars}\right) \bigcup S_j\left(\text{similars}\right)} \ \text{for i} \neq \text{j}, \tag{5}$$

where S_x*(similars)* is the set of sentences whose similarities to the sentence S_x is above a certain threshold, S_i*(similars)* $\cap$ S_j*(similars)* and S_i*(similars)* $\cup$ S_j*(similars)* indicate the numbers of the sentences in the intersection and union of the regarding sets, respectively. The threshold was empirically determined as 0.03.

Distributional feature (*f8*): is calculated based on three different compactness measurements: the number of parts where a word appears ($Compact_{Partnum}$), the distance between the first and last appearance of a term ($Compact_{FirstLastDist}$), the variance of the positions of all appearances of a term ($Compact_{PosVar}$) in a given sentence. For a document d containing N sentences, the distributional array of the word t is $array(t,d)=[c_1,c_2,\ldots,c_N]$. Then, $Compact_{Partnum}$ is calculated using

$$Compact_{PartNum}(t,d) = \sum_{i=1}^{N} c_i > 0 \; ? \; 1 : 0 \; , \tag{6}$$

where c_i is the term frequency of t in the sentence S_i.

$Compact_{FirstLastDist}$ is calculated using

$$FirstAppear(t,d) = \min_{i \in \{1 \ldots N\}} c_i > 0 \; ? \; i : N, \tag{7}$$

$$LastAppear(t,d) = \max_{i \in \{1 \ldots N\}} c_i > 0 \; ? \; i : -1, \tag{8}$$

$$Compact_{FirstLastDist}(t,d) = LastAppear(t,d) - FirstAppear(t,d) \; . \tag{9}$$

$Compact_{PosVar}$ is calculated using

$$Count(t,d) = \sum_{i=1}^{N} c_i \tag{10}$$

$$Centroid(t,d) = \frac{\sum_{i=1}^{N} c_i \times i}{Count(t,d)} \; , \tag{11}$$

$$Compact_{PosVar}(t,d) = \frac{\sum_{i=1}^{N} c_i \times |i - Centroid(t,d)|}{Count(t,d)} \; . \tag{12}$$

The overall distributional feature of a term is obtained by averaging the three compactness values. Then, the distributional feature of a sentence is calculated as the sum of distributional feature values of all terms in that sentence.

Term frequency (*f9*): is calculated by adding up the term frequency of each term in a given sentence.

Inclusion of numeric characters (*f10*): Numeric characters may include detailed information about a document. Therefore, the inclusion of numeric characters, which could play a prominent role in text summarization, is defined as another feature. This feature is simply calculated as the total number of numeric characters in a given sentence.

Inclusion of "?" and "!" (*f11*): When a sentence ends with a question mark or an exclamation point, it is an indicator of the importance of that sentence over the others in a given document. Therefore, the inclusion of these two characters is defined as a feature for a given sentence and calculated using

$$f_{11}(S_i) = \text{Number of "?" or "!" in } S_i > 0 \; ? \; 1 : 0 \; . \tag{13}$$

Inclusion of concluding words (*f12*): The sentences including certain concluding words such as “sonuçta (Eng. in conclusion)”, “özetle (Eng. in summary)”, and “neticede (Eng. eventually)” are most likely to be part of the summary. This feature corresponds to the number of the concluding words in a sentence.

Inclusion of nouns (*f13*): is calculated as the total number of nouns in a given sentence. In our work, the Zemberek library [25] was used to detect the nouns of a sentence in Turkish.

Inclusion of named entities (*f14*): considers the effects of the named entities, which are real-world objects such as a person, location, organization, product, and so on. The feature is calculated as the total number of named entities in a given sentence.

LSA score (*f15*): is calculated using LSA between terms and sentences. The term-sentence matrix consists of words and sentences in a text. The cell values in this matrix indicate how many times the word occurs in the sentence. This matrix is divided into three factors ($A=USV^T$) with the help of Singular Value Decomposition (SVD). The text summarization approach, which was proposed by [10], is used for this feature. According to that study, matrix B is attained by the right singular vector matrix V^T and the diagonal matrix S:

$$B = S^2 V^T. \tag{14}$$

The feature is then calculated using

$$f_{15}\left(S_i\right) = \sqrt{\sum_{i=1}^{r} b_{\mathrm{ik}}^2} \tag{15}$$

where b_{ik} is a sentence vector of B, and r is the number of sentences in the document.

3.3. Proposed Feature

In addition to the widely used features explained in the previous subsection, a new feature based on LSA is proposed in our work. To calculate this feature, firstly, LSA scores of all sentences in a given document are obtained using (15). Then, the sentence that has the highest LSA score is determined. Subsequently, the value of this feature for a given sentence (S_i) is obtained based on the cosine similarity to the sentence that has the highest LSA score in the document. The steps of the proposed feature extraction algorithm are summarized below:

i) $D=\{S_i \mid i=1,2,\ldots,N\}$ represents the sentences in a given document where N is the total number of sentences.

ii) $TF=\{T_j \mid j=1,2,\ldots,M\}$ represents the terms in a given sentence where M is the total number of terms.

iii) $TS_{N,M}=\left[\mathrm{ts}_{1,1} \cdots \mathrm{ts}_{1,M} \vdots \ddots \vdots \mathrm{ts}_{N,1} \cdots \mathrm{ts}_{N,M}\right]$, where $ts_{i,j}$ is the term frequency of the jth term in the ith sentence, for i=1,2,...,N, and j=1,2,...,M.

iv) By applying SVD, the matrix $TS_{N,M}$ is decomposed into three parts:

$$TS_{N,M} = USV^T. \tag{16}$$

v) L_{S_i}, which corresponds to the LSA score of S_i, is calculated as previously formulated in (15).

vi) The sentence with the highest LSA score is obtained in terms of

$$ML_{S_i} = \max\left(L_{S_i}\right), \;\; i{=}1{,}2{,}\ldots{,}N \; . \tag{17}$$

vii) The proposed feature (f_{16}) for a given sentence (S_i) is finally obtained by calculating the cosine similarity of the sentence to ML_{S_i}:

$$f_{16}\left(S_i\right) = cosine(S_i, ML_{S_i}) \; . \tag{18}$$

4. Experimental Results

The contributions of features and preprocessing methods to the performance of automatic text summarization were thoroughly investigated with extensive experimental work. This section elaborates on the experimental work and provides the results of the experiments. In the following subsections, the utilized datasets and success metrics are first described. Then, the impacts of features and preprocessing are evaluated.

The experiments were carried out using MS Visual Studio, SQL Server Management Studio, and C# programming language on a computer equipped with a 2.80 GHz CPU and 16 GB of RAM.

4.1. Datasets

Two different datasets in Turkish were utilized to evaluate the contributions of the features and preprocessing tasks to the performance of automatic text summarization. All documents in both datasets have been manually summarized by a number of human assessors in an extractive manner so that a certain number of sentences from those documents are selected and tagged for the summary of each document.

The first dataset (Dataset I) consists of 100 Turkish documents consisting of news in the fields of economy, sports, and art [28]. The second dataset (Dataset II) includes 20 Turkish documents consisting of news in various fields such as magazines, politics, and health [19]. The properties of these datasets are listed in Table 4.

Table 4. The properties of the utilized datasets

Property	**Dataset I**	**Dataset II**
Number of Documents	100	20
Number of Sentences in Documents	1,265	201
Maximum Number of Sentences per Document	42	10
Number of Assessors	17	43

4.2. Success Metrics

In this work, two different success metrics were utilized to assess the performance of automatic text summarization. The two metrics compare the automatically selected sentences to the ones manually selected by the assessors with a different strategy. In both metrics, a score is assigned to each sentence in a given document by adding up the values of the utilized features. Each score is then divided by the maximum sentence score of that document and normalized to the range from 0 to 1. The sentences are then sorted in descending order of their scores. Finally, the top 35% of the sentences in the sorted list are selected for the summary of that document. In other words, almost a third of the total number of sentences in a document (based on their score) are automatically chosen for the summary.

The first success metric (Success Metric I) has a sentence-level approach and uses the algorithm in Table 5 to evaluate the performance of the automatic text summarization against the ground truth.

Table 5. The evaluation algorithm for the Success Metric I

```
i)    $D=\{S_i \mid i=1,2,\ldots,N\}$ is a document consisting of the sentences $S_i$.
ii)   $S_i$ is the $i^{th}$ sentence in a document, where $i=1,2,\ldots,N$.
iii)  $A_j$ is the $j^{th}$ assessor, where $j=1,2,\ldots,T$.
iv)   $voteA_{i,j}$ represents the vote (1: selected | 0: unselected) of $A_j$ for $S_i$.
v)    The total vote of each sentence is calculated as $voteS_i = \sum_{j=1}^{T} voteA_{i,j}$.
vi)   The sentences are then sorted in descending order of their total votes.
vii)  The top 35% of the sentences in the sorted list form the ground truth for the summary of
      D.
viii) The performance of the automatic summarization for D is calculated as the ratio of the
      number of common sentences in the automatic and ground truth summary to the total
      number of sentences in the summary of D.
ix)   Overall summarization performance for the entire dataset is obtained by averaging the
      performances of the documents in the dataset.
```

On the other hand, the second success metric (Success Metric II) has a document-level approach [19, 27] and uses the algorithm in Table 6 to evaluate the performance.

4.3. The Impact of Features

In this subsection, the contributions of the common features, as well as our proposed feature, to the performance of automatic text summarization, were extensively examined. For this purpose, an exhaustive feature selection strategy [29–33] was employed so that every single combination of 16 different features including our proposed feature was considered. Hence, 65,535 different feature subsets were comparatively evaluated and the optimal feature subset was obtained.

Table 6. The evaluation algorithm for the Success Metric II

i) *$D=\{S_i \mid i=1,2,\ldots,N\}$ is a document consisting of the sentences S_i.*

ii) *A_j is the j^{th} assessor, where $j=1,2,\ldots,T$.*

iii) *$performanceD_j$ is the performance of the automatic summarization of D against the manual summarization by A_j. It is calculated as the ratio of the number of common sentences of the manual summarization of A_j and the automatic summarization to the total number of sentences in the summary of D.*

iv) *Overall performance of the automatic summarization of D is calculated as $\frac{1}{T}\sum_{j=1}^{T} performanceD_j$.*

v) *Overall summarization performance for the entire dataset is obtained by averaging the performances of the documents in the dataset.*

According to these experiments, the individual performances of each feature are listed in Table 7, where the highest scores for each success metric and dataset are indicated in bold. As shown in that table, our proposed feature offered the best performance in both datasets for the Success Metric I. In the meantime, based on the Success Metric II, the proposed feature offered the best performance in Dataset I and the second-best performance right after *f3* in Dataset II.

Table 7. The summarization performance of each feature for two datasets

	Dataset I		**Dataset II**	
Feature	**Success Metric I**	**Success Metric II**	**Success Metric I**	**Success Metric II**
f1	0.677	0.632	0.576	0.515
f2	0.587	0.587	0.621	0.529
f3	0.690	0.636	0.750	**0.604**
f4	0.551	0.542	0.508	0.455
f5	0.641	0.561	0.725	0.588
f6	0.493	0.500	0.180	0.367
f7	0.651	0.609	0.415	0.392
f8	0.580	0.583	0.613	0.520
f9	0.611	0.596	0.606	0.512
f10	0.366	0.330	0.372	0.371
f11	0.012	0.018	0.013	0.073
f12	0.138	0.138	0.244	0.217
f13	0.595	0.597	0.582	0.504
f14	0.597	0.604	0.566	0.532
f15	0.640	0.609	0.545	0.495
f16	**0.695**	**0.643**	**0.760**	0.599

Among 65,535 feature subsets, the top-10 feature subsets providing the highest summarization performance on Datasets I and II are listed in Tables 8 and 9, respectively. One can observe from these tables that the features *f1* and *f5* were present in all of the top-10 feature subsets in Dataset I for both success metrics. Also, the features *f3* and *f5* played an active role in these subsets for Dataset II and both success metrics. Besides, the proposed feature *f16* was present in most of the subsets in both datasets for both success metrics.

In this part of the experiments, the best feature subsets for each of the 16 dimensions were fi-

Table 8. The top-10 feature subsets providing the highest performances in Dataset I

Feature Subset	**Success Metric I**	**Feature Subset**	**Success Metric II**
f1, f2, f5, f8, f16	0.724	*f1, f2, f3, f5, f8, f15, f16*	0.698
f1, f2, f5, f8, f15, f16	0.723	*f1, f2, f5, f8, f15, f16*	0.697
f1, f2, f3, f5, f7, f8, f16	0.722	*f1, f2, f5, f15, f16*	0.696
f1, f2, f5, f8, f11, f16	0.722	*f1, f2, f3, f5, f8, f11, f12, f15, f16*	0.696
f1, f2, f3, f5, f8, f10, f16	0.722	*f1, f2, f3, f5, f7, f8, f16*	0.696
f1, f2, f3, f5, f8	0.721	*f1, f2, f3, f5, f8, f11, f15, f16*	0.695
f1, f2, f3, f5, f7, f8, f11, f16	0.721	*f1, f2, f3, f5, f15*	0.695
f1, f3, f4, f5, f9, f15, f16	0.721	*f1, f2, f3, f5, f8, f12, f15, f16*	0.695
f1, f2, f3, f5, f8, f10, f11, f16	0.721	*f1, f2, f3, f5, f8, f10, f16*	0.695
f1, f2, f3, f5, f8, f11	0.721	*f1, f3, f5, f8, f15, f16*	0.694

Table 9. The top-10 feature subsets providing the highest performances in Dataset II

Feature Subset	**Success Metric I**	**Feature Subset**	**Success Metric II**
f3, f16	0.770	*f5, f12*	0.615
f3, f11, f16	0.770	*f5, f11, f12*	0.615
f16	0.760	*f3*	0.604
f11, f16	0.760	*f3, f11*	0.604
f3	0.750	*f16*	0.599
f3, f11	0.750	*f11, f16*	0.599
f5	0.725	*f3, f16*	0.593
f5, f11	0.725	*f3, f11, f16*	0.593
f1, f2, f5, f12, f16	0.696	*f5*	0.588
f1, f2, f3, f5, f12, f16	0.696	*f5, f11*	0.588

nally obtained. The best subsets for each dataset are listed in Tables 10 and 11, where the optimal subsets providing the highest summarization performance for each success metric and dataset are indicated in bold. In Dataset I, the feature subset (*f1,f2,f5,f8,f16*) offered the highest summarization performance using Success Metric I, whereas the feature subset (*f1,f2,f3,f5,f8,f15,f16*) was the best for Success Metric II. Meanwhile, in Dataset II, the feature subsets (*f3,f16*) and (*f5,f12*) offered the best performance for Success Metric I and II, respectively. Also, it can be easily noted that the proposed feature *f16* was present in most of the best combinations with just a few exceptions. This outcome confirmed the effectiveness of the proposed feature on automatic text summarization.

4.4. The Impact of Preprocessing

Though applying all preprocessing methods is a common practice in automatic text summarization, the contributions of the presence or absence of these methods to the performance of automatic text summarization are not considered as mentioned earlier. Therefore, in this part of the experimental work, all 16 combinations of four common preprocessing methods (listed in Table 2), including tokenization (TK), lowercase conversion (LC), stemming (ST), and stopword removal (SR) were comparatively evaluated. For this analysis, the best feature subsets, which have been determined in the previous subsection, were used. Each combination of the

Table 10. The best feature subsets for each dimension and their performances in Dataset I

Dim.	Best Feature Subset	Success Metric I	Best Feature Subset	Success Metric II
1	*f16*	0.695	*f16*	0.643
2	*f11,f16*	0.698	*f15,f16*	0.668
3	*f1,f2,f16*	0.706	*f5,f13,f16*	0.681
4	*f1,f5,f13,f16*	0.714	*f1,f5,f13,f16*	0.689
5	***f1,f2,f5,f8,f16***	**0.724**	*f1,f2,f5,f15,f16*	0.696
6	*f1,f2,f5,f8,f15,f16*	0.723	*f1,f2,f5,f8,f15,f16*	0.697
7	*f1,f2,f3,f5,f7, f8,f16*	0.722	***f1,f2,f3,f5,f8,f15,f16***	**0.698**
8	*f1,f2,f3,f5,f7, f8,f11,f16*	0.721	*f1,f2,f3,f5,f8,f11,f15,f16*	0.695
9	*f1,f2,f10,f11,f12,f13,f14,f15, f16*	0.636	*f1,f2,f3,f5,f8,f11,f12,f15,f16*	0.696
10	*f1,f2,f3,f10,f11,f12,f13,f14, f15,f16*	0.642	*f1,f2,f3,f5,f7,f8,f11,f12,f15,f16*	0.690
11	*f1,f2,f3,f4,f10,f11,f12,f13,f14,f15,f16*	0.650	*f1,f2,f3,f5,f7,f8,f9,f11,f12,f15,f16*	0.686
12	*f1,f2,f3,f4,f5,f10,f11,f12,f13,f14,f15,f16*	0.669	*f1,f2,f3,f4,f5,f8,f9,f10,f11,f13,f15,f16*	0.672
13	*f1,f2,f3,f4,f5,f7,f8,f9,f10,f11,f13,f15,f16*	0.697	*f1,f2,f3,f4,f5,f7,f8,f9,f10,f11,f13, f15,f16*	0.667
14	*f1,f2,f3,f4,f5,f7,f8,f9,f10,f11,f12,f13,f15,f16*	0.696	*f1,f2,f3,f4,f5,f7,f8,f9,f10,f11,f12, f13,f15,f16*	0.666
15	*f1,f2,f3,f4,f5,f7,f8,f9,f10,f11,f12,f13,f14,f15,f16*	0.671	*f1,f2,f3,f4,f5,f7,f8,f9,f10,f11,f12,f13,f14,f15,f16*	0.648
16	*f1,f2,f3,f4,f5,f6,f7,f8,f9,f10,f11,f12,f13,f14,f15,f16*	0.665	*f1,f2,f3,f4,f5,f6,f7,f8,f9,f10,f11,f12,f13,f14,f15,f16*	0.630

Table 11. The best feature subsets for each dimension and their performances in Dataset II

Dim.	Best Feature Subset	Success Metric I	Best Feature Subset	Success Metric II
1	*f16*	0.760	*f3*	0.604
2	***f3,f16***	**0.770**	***f5,f12***	**0.615**
3	*f3,f11,f16*	0.770	*f5,f11,f12*	0.615
4	*f3,f5,f11,f16*	0.682	*f1,f11,f13,f16*	0.577
5	*f1,f2,f5,f12,f16*	0.696	*f1,f3,f5,f13,f16*	0.574
6	*f1,f2,f3,f5,f12,f16*	0.696	*f1,f3,f5,f12,f13,f16*	0.568
7	*f1,f2,f3,f5,f11,f12,f16*	0.696	*f1,f3,f5,f11,f12,f13,f16*	0.568
8	*f5,f8,f9,f10,f11,f12,f13,f14*	0.680	*f1,f3,f5,f10,f11,f12,f13,f16*	0.553
9	*f1,f2,f3,f4,f5,f7,f8,f9,f16*	0.663	*f1,f2,f3,f4,f5,f6,f8,f9,f16*	0.548
10	*f1,f2,f3,f4,f5,f6,f7,f8,f9,f16*	0.663	*f2,f3,f4,f5,f6,f7,f8,f9,f12,f16*	0.548
11	*f1,f2,f3,f4,f5,f6,f7,f8,f9,f10,f16*	0.663	*f1,f2,f3,f4,f5,f6,f7,f8,f9,f10,f16*	0.548
12	*f1,f2,f3,f4,f5,f6,f7,f8,f9,f10,f13,f16*	0.663	*f1,f2,f3,f4,f5,f6,f7,f8,f9,f10,f11,f16*	0.548
13	*f1,f2,f3,f4,f5,f6,f7,f8,f9,f10,f11,f12,f16*	0.663	*f1,f2,f3,f4,f5,f6,f7,f8,f9,f10,f11,f12,f16*	0.548
14	*f1,f2,f3,f4,f5,f6,f7,f8,f9,f10,f11,f12,f13,f16*	0.663	*f1,f2,f3,f4,f5,f6,f7,f8,f9,f10,f11,f12,f13,f16*	0.548
15	*f1,f2,f3,f4,f5,f6,f7,f8,f9,f10,f11,f12,f13,f14,f16*	0.663	*f1,f2,f3,f4,f5,f6,f7,f8,f9,f10,f11,f12,f13,f14,f16*	0.548
16	*f1,f2,f3,f4,f5,f6,f7,f8,f9,f10,f11,f12,f13,f14,f15,f16*	0.618	*f1,f2,f3,f4,f5,f6,f7,f8,f9,f10,f11,f12,f13,f14,f15,f16*	0.541

preprocessing tasks and the corresponding summarization performance in Dataset I and II are respectively listed in Tables 12 and 13, where the highest scores for each metric are indicated in bold. In Dataset I, the combination (ST: Enabled | SR: Enabled | TK: Alphanumeric | LC: Disabled) offered the best performance for both success metrics, whereas the combination (ST: Enabled | SR: Enabled | TK: Alphabetic | LC: Enabled) provided the same performance for Success Metric I, as well. In Dataset II, on the other hand, four different combinations offered the best performance for Success Metric I. Two of these four combinations were the same as those that provided the best performance in Dataset I. Also, two different combinations including (ST: Enabled | SR: Disabled | TK: Alphanumeric | LC: Disabled) and (ST: Enabled | SR: Disabled | TK: Alphabetic | LC: Disabled) attained the best performance for Success Metric II. Based on these results, it was revealed that enabling certain preprocessing tasks, rather than all, can improve summarization performance depending on the dataset and success metric used.

In the last analysis, the processing times of the best feature subsets together with the best combination of preprocessing methods were also measured and listed in Table 14. From this table, one can note that the use of appropriate features and preprocessing tasks not only improves the summarization performance but also reduces the computational load.

Table 12. The preprocessing tasks vs. the summarization performance in Dataset I

Preprocessing Tasks	**Success Metric I for the Best Feature Subset (*f1,f2,f5,f8,f16*)**	**Success Metric II for the Best Feature Subset (*f1,f2,f3,f5,f8,f15,f16*)**
ST: Disabled | SR: Disabled | TK: Alphanumeric | LC: Disabled	0.686	0.655
ST: Disabled | SR: Disabled | TK: Alphanumeric | LC: Enabled	0.696	0.662
ST: Disabled | SR: Disabled | TK: Alphabetic | LC: Disabled	0.681	0.652
ST: Disabled | SR: Disabled | TK: Alphabetic | LC: Enabled	0.693	0.669
ST: Disabled | SR: Enabled | TK: Alphanumeric | LC: Disabled	0.681	0.661
ST: Disabled | SR: Enabled | TK: Alphanumeric | LC: Enabled	0.683	0.665
ST: Disabled | SR: Enabled | TK: Alphabetic | LC: Disabled	0.686	0.676
ST: Disabled | SR: Enabled | TK: Alphabetic | LC: Enabled	0.683	0.679
ST: Enabled | SR: Disabled | TK: Alphanumeric | LC: Disabled	0.709	0.690
ST: Enabled | SR: Disabled | TK: Alphanumeric | LC: Enabled	0.709	0.686
ST: Enabled | SR: Disabled | TK: Alphabetic | LC: Disabled	0.716	0.696
ST: Enabled | SR: Disabled | TK: Alphabetic | LC: Enabled	0.716	0.694
ST: Enabled | SR: Enabled | TK: Alphanumeric | LC: Disabled	**0.724**	**0.698**
ST: Enabled | SR: Enabled | TK: Alphanumeric | LC: Enabled	0.719	0.691
ST: Enabled | SR: Enabled | TK: Alphabetic | LC: Disabled	0.723	0.693
ST: Enabled | SR: Enabled | TK: Alphabetic | LC: Enabled	**0.724**	0.693

Table 13. The preprocessing tasks vs. the summarization performance in Dataset II

Preprocessing Tasks	**Success Metric I for the Best Feature Subset (*f3, f16*)**	**Success Metric II for the Best Feature Subset (*f5, f12*)**
ST: Disabled | SR: Disabled | TK: Alphanumeric | LC: Disabled	0.680	0.471
ST: Disabled | SR: Disabled | TK: Alphanumeric | LC: Enabled	0.680	0.530
ST: Disabled | SR: Disabled | TK: Alphabetic | LC: Disabled	0.680	0.471
ST: Disabled | SR: Disabled | TK: Alphabetic | LC: Enabled	0.680	0.530
ST: Disabled | SR: Enabled | TK: Alphanumeric | LC: Disabled	0.663	0.465
ST: Disabled | SR: Enabled | TK: Alphanumeric | LC: Enabled	0.628	0.523
ST: Disabled | SR: Enabled | TK: Alphabetic | LC: Disabled	0.663	0.465
ST: Disabled | SR: Enabled | TK: Alphabetic | LC: Enabled	0.637	0.523
ST: Enabled | SR: Disabled | TK: Alphanumeric | LC: Disabled	0.730	**0.627**
ST: Enabled | SR: Disabled | TK: Alphanumeric | LC: Enabled	0.730	0.623
ST: Enabled | SR: Disabled | TK: Alphabetic | LC: Disabled	0.730	**0.627**
ST: Enabled | SR: Disabled | TK: Alphabetic | LC: Enabled	0.730	0.623
ST: Enabled | SR: Enabled | TK: Alphanumeric | LC: Disabled	**0.770**	0.615
ST: Enabled | SR: Enabled | TK: Alphanumeric | LC: Enabled	**0.770**	0.610
ST: Enabled | SR: Enabled | TK: Alphabetic | LC: Disabled	**0.770**	0.615
ST: Enabled | SR: Enabled | TK: Alphabetic | LC: Enabled	**0.770**	0.610

5. Discussion

The experiments provided useful insights on the effects of the features, as well as the preprocessing methods, on automatic text summarization performance. An appropriate selection of the

Table 14. The processing time analysis of the preprocessing tasks

Dataset	Success Metric	Best Feature Subset	Preprocessing Tasks	Processing Time
I	I	*f1, f2, f5, f8, f16*	ST: Enabled | SR: Enabled | TK: Alphanumeric | LC: Disabled	813 sec
			ST: Enabled | SR: Enabled | TK: Alphabetic | LC: Enabled	871 sec
I	II	*f1, f2, f3, f5, f8, f15, f16*	ST: Enabled | SR: Enabled | TK: Alphanumeric | LC: Disabled	1161 sec
			ST: Enabled | SR: Enabled | TK: Alphabetic | LC: Enabled	1223 sec
II	I	*f3, f16*	ST: Enabled | SR: Enabled | TK: Alphanumeric | LC: Disabled	79 sec
			ST: Enabled | SR: Enabled | TK: Alphabetic | LC: Enabled	79 sec
II	II	*f5, f12*	ST: Enabled | SR: Disabled | TK: Alphanumeric | LC: Disabled	55 sec
			ST: Enabled | SR: Enabled | TK: Alphabetic | LC: Enabled	90 sec

features and preprocessing methods yielded a superior summarization performance compared to the relevant literature, though a one-to-one comparison is not always possible. For example, in one of the related works [28], the inclusion of sentences in a summary was decided based on a classification approach and by considering the category of the documents. Also, the effectiveness of 8 different features for the summarization performance was evaluated on Dataset I using a suboptimal feature selection method rather than an exhaustive search. While the classification results cannot be directly compared to our study, the summarization performance of the related work based on the Success Metric I was reported between 0.502 and 0.595 for specific document categories, whereas our work achieved a maximum performance of 0.724 for the same experimentation. In another example [27], 15 different features and some of their subsets were evaluated on Dataset II without a systematic feature selection strategy. The performance of their work based on the Success Metric I was reported between 0.657 and 0.738, whereas our work achieved a maximum performance of 0.774 for the same experimentation. Considering these two examples and other similar studies in the literature, our work is superior thanks to the appropriate selection of features as well as the new feature proposed in our work. Furthermore, to the best of our knowledge, the impacts of preprocessing methods on summarization performance have not been evaluated either in the examples above or in other relevant studies in the literature. Hence, the extensive analysis of preprocessing methods is another important contribution of our work.

6. Conclusions

Though there are many studies on automatic text summarization for English, there is just a limited number of works in the same field for Turkish. Hence, our work focused on text summarization, particularly for the Turkish language, and comprehensively analyzed the contributions of common features and preprocessing methods to the performance of automatic text summarization. Moreover, a new feature based on LSA was proposed as another contribution. A series of extensive experimental studies revealed the best feature subsets and the best combination of preprocessing methods that could improve the summarization performance. It has been observed that using appropriate combinations of features as well as preprocessing tasks rather than using all not only improved the summarization performance but also reduced the required computational load. In addition, the proposed feature has been confirmed to further improve the performance. Based on these results, it can be concluded that the features and preprocessing tasks, which need to be used to achieve the best summarization performance, may vary depending on the domain of the text and the utilized success metric. In future work, a similar analysis can be carried out for languages other than Turkish.

References

[1] B. MUTLU, E. A. SEZER and M. A. AKCAYOL, *Candidate sentence selection for extractive text summarization*, Information Processing & Management **57**(6), 102359, 2020.

[2] S. LAMSIYAH, A. EL MAHDAOUY, B. ESPINASSE and S. E. A. OUATIK, *An unsupervised method for extractive multi-document summarization based on centroid approach and sentence embeddings*, Expert Systems with Applications **167**, 114152, 2021.

[3] H. N. FEJER and N. OMAR, *Automatic Arabic text summarization using clustering and keyphrase extraction*. Proceedings of the 6th International Conference on Information Technology and Multimedia, Putrajaya, Malaysia, pp. 293–298, 2014.

[4] A. GURAN, N. G. BAYAZIT and M. Z. GURBUZ, *Efficient feature integration with Wikipedia-based semantic feature extraction for Turkish text summarization*, Turkish Journal of Electrical Engineering & Computer Sciences **21**(5), pp. 1411–1425, 2013.

[5] R. C. BELWAL, S. RAI and A. GUPTA, *Text summarization using topic-based vector space model and semantic measure*, Information Processing & Management **58**(3), 102536, 2021.

[6] L. SUANMALI, N. SALIM and M. S. BINWAHLAN, *Fuzzy logic based method for improving text summarization*, International Journal of Computer Science and Information Security **2**(1), pp. 65–70, 2009.

[7] R. M. ALIGULIYEV, *A new sentence similarity measure and sentence based extractive technique for automatic text summarization*, Expert Systems with Applications **36**(4), pp. 7764–7772, 2009.

[8] L. CAGLIERO and M. LA QUATRA, *Extracting highlights of scientific articles: a supervised summarization approach*, Expert Systems with Applications **160**, 113659, 2020.

[9] Z. ALTAN, *A Turkish automatic text summarization system*, Proceedings of IASTED International Conference on Artificial Intelligence and Applications, Innsbruck, Austria, pp. 311–316, 2004.

[10] J. STEINBERGER, *Text summarization within the LSA framework*, Ph.D. thesis., Computer Science and Engineering, University of West Bohemia, Plzeň, Czech Republic, 2007.

[11] P. P. TARDAN, A. ERWIN, K. I. ENG and W. MULIADY, *Automatic text summarization based on semantic analysis approach for documents in Indonesian language*, Proceedings of International Conference on Information Technology and Electrical Engineering, Yogyakarta, Indonesia, pp. 1–6, 2013.

[12] C. CIGIR, M. KUTLU and I. CICEKLI, *Generic text summarization for Turkish*, Proceedings of 24th International Symposium on Computer and Information Sciences, Guzelyurt, Northern Cyprus, pp. 224–229, 2009.

[13] D. HINGU, D. SHAH and S. S. UDMALE, *Automatic text summarization of Wikipedia articles*, Proceedings of International Conference on Communication, Information & Computing Technology, Mumbai, India, pp. 1–4, 2015.

[14] J. K. GEETHA and N. DEEPAMALA, *Kannada text summarization using latent semantic analysis*, Proceedings of International Conference on Advances in Computing, Communications and Informatics, Kochi, India, pp. 1508–1512, 2015.

[15] O. M. FOONG, S. P. YONG and F. A. JAID, *Text summarization using latent semantic analysis model in mobile Android platform*, Proceedings of 9th Asia Modelling Symposium, Kuala Lumpur, Malaysia, pp. 35–39, 2015.

[16] J. YADAV and Y. K. MEEANA, *Use of fuzzy logic and wordnet for improving performance of extractive automatic text summarization*, Proceedings of International Conference on Advances in Computing, Communications and Informatics, Jaipur, India, pp. 2071–2077, 2016.

[17] G. A. MILLER, R. BECKWITH, C. FELLBAUM, D. GROSS and K. J. MILLER, *Introduction to wordnet: an on-line lexical database*, International Journal of Lexicography **3**(4), pp. 235–244, 1990.

[18] X. HAN, T. LV, Q. JIANG, X. WANG and C. WANG, *Text summarization using sentence-level semantic graph model*, Proceedings of 4th International Conference on Cloud Computing and Intelligence Systems, Beijing, China, pp. 171–176, 2016.

[19] A. GURAN, M. UYSAL, Y. EKINCI and C. B. GURAN, *An additive FAHP based sentence score function for text summarization*, Information Technology and Control **46**(1), pp. 53–69, 2017.

[20] C. HARK and A. KARCI, *Karci summarization: a simple and effective approach for automatic text summarization using Karci entropy*, Information Processing & Management **57**(3), 102187, 2020.

[21] A. K. UYSAL and S. GUNAL, *The impact of preprocessing on text classification*, Information Processing & Management **50**(1), pp. 104–112, 2014.

[22] R. HE, J. TANG, P. GONG, Q. HU and B. WANG, *Multi-document summarization via group sparse learning*, Information Sciences **349**, pp. 12–24, 2016.

[23] M. BIDOKI, M. R. MOOSAVI and M. FAKHRAHMAD, *A semantic approach to extractive multi-document summarization: applying sentence expansion for tuning of conceptual densities*, Information Processing & Management **57**(6), 102341, 2020.

[24] F. HOROSAN and B. BILEN, *Extractive text summarization system for news texts*, International Journal of Applied Mathematics Electronics and Computers **8**(4), pp. 179–184, 2020.

[25] A. A. AKIN and M. D. AKIN, *Zemberek, an open source NLP framework for Turkic languages*, Structure **10**(2007), pp. 1–5, 2007.

[26] A. GURAN, *Automatic text summarization system*, Ph.D. thesis, Yildiz Technical University, Istanbul, Turkiye, 2013.

[27] A. GURAN, S. N. ARSLAN, E. KILIC and B. DIRI, *Sentence selection methods for text summarization*, Proceedings of IEEE 22nd Signal Processing and Communications Applications Conference, Trabzon, Turkiye, pp. 192–195, 2014.

[28] S. BAL and E. SORA GUNAL, *A new model on automatic text summarization for Turkish*, Eskisehir Technical University Journal of Science and Technology A - Applied Sciences and Engineering **22**(2), pp. 189–198, 2021.

[29] G. COXSON and J. RUSSO, *Efficient exhaustive search for optimal-peak-sidelobe binary codes*, IEEE Transactions on Aerospace and Electronic Systems **41**(1), pp. 302–308, 2005.

[30] S. GUNAL, O. N. GEREK, D. G. ECE and R. EDIZKAN, *The search for optimal feature set in power quality event classification*, Expert Systems with Applications **36**(7), pp. 10266–10273, 2009.

[31] L. HUI and C. YONGHUI, *Study of heuristic search and exhaustive search in search algorithms of the structural learning*, Proceedings of Second International Conference on Multimedia and Information Technology, Kaifeng, China, pp. 169–171, 2010.

[32] J. A. WRIGHT, E. NIKOLAIDOU and C. J. HOPFE, *Exhaustive search; does it have a role in explorative design?*, Proceedings of Third Building Simulation & Optimization Conference, Newcastle, United Kingdom, pp. 1–8, 2016.

[33] K. NATALIA, K. MIKHAIL and B. GEORGII, *On using gray codes to improve the efficiency of the parallel exhaustive search algorithm for the knapsack problem*, Proceedings of IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering, Saint Petersburg and Moscow, Russia, pp. 1821–1825, 2019.