Jurnal Ilmu Komputer dan Informasi (Journal of Computer Science and Information). 8/1 (2015), 1-10
DOI: http://dx.doi.org/10.21609/jiki.v8i1.278

COVERAGE, DIVERSITY, AND COHERENCE OPTIMIZATION FOR MULTI-DOCUMENT SUMMARIZATION

Khoirul Umam, Fidi Wincoko Putro, Gulpi Qorik Oktagalu Pratamasunu, Agus Zainal Arifin, and Diana Purwitasari

Department of Informatics, Faculty of Information Technology, Institut Teknologi Sepuluh Nopember, Kampus ITS Sukolilo, Surabaya, 60111, Indonesia

E-mail: khoirul.umam35@gmail.com, agusza@cs.its.ac.id

Abstract

A great summarization on multi-document with similar topics can help users to get useful information. A good summary must have an extensive coverage, minimum redundancy (high diversity), and smooth connection among sentences (high coherence). Therefore, multi-document summarization that considers the coverage, diversity, and coherence of summary is needed. In this paper we propose a novel method on multi-document summarization that optimizes the coverage, diversity, and coherence among the summary's sentences simultaneously. It integrates self-adaptive differential evolution (SaDE) algorithm to solve the optimization problem. Sentences ordering algorithm based on topical closeness approach is performed in SaDE iterations to improve coherences among the summary's sentences. Experiments have been performed on Text Analysis Conference (TAC) 2008 data sets. The experimental results showed that the proposed method generates summaries with average coherence and ROUGE scores 29-41.2 times and 46.97-64.71% better than any other method that only consider coverage and diversity, respectively.

Keywords: *multi-document summarization, optimization, self-adaptive differential evolution, sentences ordering, topical closeness*

Abstrak

Peringkasan yang baik terhadap dokumen-dokumen dengan topik yang seragam dapat membantu pembaca dalam memperoleh informasi secara cepat. Ringkasan yang baik merupakan ringkasan dengan cakupan pembahasan (*coverage*) yang luas dan dengan tingkat keberagaman (*diversity*) serta keterhubungan antarkalimat (*coherence*) yang tinggi. Oleh karena itu dibutuhkan metode peringkasan multi-dokumen yang mempertimbangkan tingkat *coverage*, *diversity*, dan *coherence* pada hasil ringkasan. Pada *paper* ini dikembangkan sebuah metode baru dalam peringkasan multi-dokumen dengan mengoptimasi tingkat *coverage*, *diversity*, dan *coherence* antarkalimat hasil ringkasan secara simultan. Optimasi hasil ringkasan dilakukan dengan menggunakan algoritma *self-adaptive differential evolution* (SaDE). Algoritma pengurutan kalimat yang menggunakan pendekatan *topical closeness* juga diintegrasikan ke dalam tiap iterasi algoritma SaDE untuk meningkatkan koherensi antarkalimat hasil ringkasan. Uji coba dilakukan pada 15 topik *dataset Text Analysis Conference* (TAC) 2008. Hasil uji coba menunjukkan bahwa metode yang diusulkan dapat menghasilkan ringkasan dengan rata-rata koherensi 29-41,2 kali lebih tinggi serta skor ROUGE 46,97-64,71% lebih besar dibandingkan dengan metode yang hanya mempertimbangkan *coverage* dan *diversity* hasil ringkasan.

Kata Kunci: *optimasi, pengurutan kalimat, peringkasan multi-dokumen, self-adaptive differential evolution, topical closeness*

1. Introduction

The contents of a document can be long. It presents several information with specified topic. Current technological developments makes people can find related documents with similar topic easier than before. The other documents can be had a long contents too. It means there is a massive quantity of data or information with similar obtainable topic.

The massive quantity of data available in the Internet today has reached such a huge volume. It becomes humanly unfeasible to get efficiently useful information from the Internet [1]. Thus, automatic methods are needed in order to get useful information from the documents efficiently.

Document summarization is one of methods to process information automatically. It creates compressed version of documents that provides useful information that covers all information in the original documents relevantly. Document summarization can be classified based on the number of document processed simultaneously, i.e. single-document and multi-document summarization. Single-document summarization processes only one document into a summary, whereas multi-document summarization processes more than one document with similar topic into a summary.

Various kinds of algorithms are proposed on multi-document summarization problem. These algorithms include ontology-based, clustering, and heuristic approach. The example of document summarization method that uses ontology-based approach is the proposed method in [2]. It can perform multi-document summarization by utilizing Yago ontology to capture the intent and context of sentences in documents. It can choose the exact meaning of sentences that has ambiguous word based on Yago ontology scores.

Multi-document summarization methods based on clustering approach have been also proposed. For example, the method proposed in [3]. It generates a summary from sentences set that have been clustered based on similarity between sentences. In the other multi-document summarization method that has been proposed in [1] also there is a clustering stage.

Whereas the multi-document summarization methods based on heuristic approach are methods that utilize optimization algorithm in order to select the summary's sentences properly. One of multi-document summarization methods that use this approach is Optimization of Coverage and Diversity for Summarization using Self-adaptive Differential Evolution (OCDsum-SaDE) method that proposed in [4]. In the method, an optimal summary is searched by considering the coverage and diversity of summary's sentences.

Multi-document summarization cannot be separated from sentences ordering process. The process is needed to be performed in order to obtain the composition of the summary's sentences that allows users to get information easily. Several summary's sentences ordering methods had been proposed in [5-7]. It considers a variety of approaches, i.e. chronological, probabilistic, topical closeness, precedence, succession, semantic, and text entailment approaches. The process is generally carried out after the document summarization process completes. Thus, the results of sentences ordering depend on the summary.

A good summary is expected to meet three factors. These factors are: 1) an extensive coverage; 2) high diversity or minimum redundancy; 3) high coherences among summary's sentences [4]. Summary that have an extensive coverage indicates it has summarized all information from original documents. Summary's sentences with high diversity or minimum redundancy indicate the summary able to presents information without any convoluted. On other hand, the smooth connectivity between summary's sentences may help the users to understand and absorb information from summary easily.

Process to obtain the best summary can be considered as an optimization problem [8]. Therefore, the process to generate a summary with high level of coverage, diversity, and coherences among the sentences also can be considered as an optimization problem. Thus, a multi-document summarization method that considers optimizing those factors simultaneously is needed to study in order to generate a good summary.

In this paper, we propose a novel method for multi-document summarization that considers the coverage, diversity, and coherence of the summary. This method is inspired by self-adaptive differential evolution (SaDE) algorithm from [4] and sentences ordering algorithm using topical closeness approach in [6]. SaDE algorithm is used to solve the coverage, diversity, and coherence optimization problem. Whereas the topical closeness approach that integrated to SaDE iterations helps to find the solution of summary with optimal coherences. Thus, this method can generates summary with an extensive coverage, minimum redundancy, and high coherence among the summary's sentences.

2. Methods

Summary's Quality Factors

In this section, we describe three factors of summary's quality (i.e. coverage, diversity, and coherence) that optimized in our proposed method.

Coverage

Let N denotes the number of sentences from documents that will be summarized, M denotes the number of distinct terms in documents, sen_n denotes the nth sentence from documents which has normalized form sen_n^{norm}, $term_m$ denotes the m-th distinct term from documents, tf_{nm} denotes the number of occurrences of $term_m$ in sen_n^{norm}, isf_m denotes inverse sentence frequency of $term_m$, and N_m denotes the number of sentences containing $term_m$. Term's weight of $term_m$ in sen_n^{norm} (w_{nm}) can be calculated using term frequency inverse sentence frequency (TF-ISF) scheme in equation(1) and equation(2):

$$w_{nm} = tf_{nm} \times isf_m, \qquad (1)$$

$$isf_m = log\left(\frac{N}{N_m}\right). \quad (2)$$

The sen_n^{norm} is represented as a vector which has M components such that $sen_n^{norm} = [w_{n1}, \ldots, w_{nM}]$. The similarity between sentences can be calculated using cosine measure formulation in equation(3):

$$sim\left(sen_i^{norm}, sen_j^{norm}\right) = \frac{\sum_{k=1}^{M}(w_{ik}, w_{jk})}{\sqrt{\sum_{k=1}^{M} w_{ik}^2 \cdot \sum_{k=1}^{M} w_{jk}^2}}. \quad (3)$$

Summary's coverage value reflects the coverage of summary's contents towards contents in original documents. It can be calculated by considering similarity between main content in original documents with main content in candidate summary [4]. Radev et al. [9] describes that main content of documents set is reflected by its centroid or its term's weight means.

Centroid of original documents and candidate summary are represented as a vector with M components. Let $S_p(t)$ denotes set of sentences in p-th candidate summary on t-th generation and $N_p^S(t)$ denotes the number of sentences in $S_p(t)$. Each component o_m of the original documents's centroid O and each component $o_{p,m}^S(t)$ of the p th candidate summary's centroid on current generation $O_p^S(t)$ can be calculated using equation(4) and equation(5), respectively:

$$o_m = \frac{1}{N}\sum_{n=1}^{N} w_{nm}, \quad (4)$$

$$o_{p,m}^S(t) = \frac{1}{N_p^S(t)} \sum_{sen_n^{norm} \in S_p(t)} w_{nm}. \quad (5)$$

Alguliev et al. [4] also describes that by considering the similarity between main content of original documents and main content of summary, we will know the importance of summary towards original documents. Moreover, by considering the similarity between main content of original documents with each summary's sentence, we will know the importance of each summary's sentence towards its original documents. The greater similarity between main content of original documents with a summary's sentence reflects the more importance of the sentence towards original documents. Therefore, greater summary's coverage value reflects better summary. The formulation to calculate summary's coverage value $f_{coverage}$ is shown in equation(6). In the equation(6), $U_p^{bin}(t)$ denotes binary form of vector solution for the pth candidate summary on t-th generation and $u_{p,n}^{bin}(t)$ denotes the nth component of $U_p^{bin}(t)$. Process to generate this vector will be described in the next section.

Diversity

Summary's diversity value reflects the diversity of summary's sentences. It can be considered by calculating similarity between each summary's sentences. If the summary has high total value of sentences similarity, then it has low diversity. Otherwise, if the summary has low total value of sentences similarity, then it has high diversity between its sentences [4].

Summary with low diversity between its sentences tends to present a poor summary because its sentences tend to discuss redundant information. Therefore, in order to get a good summary, the combination of summary's sentences that has high diversity have to be found. In other words, the combination of summary's sentences with low total value of its sentences similarity have to be found, because it can present the information with minimum redundancy.

In this paper, the summary's diversity value is defined as total value of its sentences similarity. Therefore, its diversity value is related with diversity of its sentences inversely. The lower its diversity value reflects the more diversity in its sentences and also the better summary.

The formulation to calculate summary's diversity value $f_{diversity}$ is shown in equation(7). Equation(7) only sums similarity between summary's sentences and ignores sentences which not in the summary [4].

Coherence

Summary's coherence value reflects the summary's sentences coherences degree. It corresponds with smooth connectivity between summary's sentences. Thus, it also corresponds with readability of information in summary by readers. A summary with higher coherences degree is expected to be easier for reader in order to understand the information which presents by the summary.

Generally a summary can simplify the read-

$$f_{coverage}\left(U_p^{bin}(t)\right) = sim\left(O, O_p^S(t)\right) \cdot \sum_{n=1}^{N} sim(O, sen_n^{norm}) u_{p,n}^{bin}. \quad (6)$$

$$f_{diversity}\left(U_p^{bin}(t)\right) = \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} sim(sen_i, sen_j) u_{p,i}^{bin} u_{p,j}^{bin}. \quad (7)$$

$$f_{coherence}\left(U_p^{ord}(t)\right) = \frac{\sum_{i=1}^{N_p^S(t)-1} sim\left(sen_{s(i)}^{norm}, sen_{s(i+1)}^{norm}\right)}{N_p^S(t)-1}, s(i) = u_{p,i}^{ord}(t). \quad (8)$$

Algorithm 1. Sentences Ordering Type A Algorithm

```
1. From sentences in the candidate summary, choose two sentences ($sen_i$ and $sen_j$) which has highest similarity ($sim(sen_i, sen_j)$) and then make it as initialization of ordering result → $ord = [sen_i, sen_j]$.
2. Change $sen_i$ and $sen_j$ status to be head and tail, respectively.
3. For each sentence which has not in ordering result, choose a sentence ($sen_x$) which has highest similarity if paired with head or tail.
4. Do one of following conditional:
    a. If $sim(head, sen_x) \geq sim(tail, sen_x)$, then put $sen_x$ in front of the head and change $sen_x$ status to be head → $ord = [sen_x, sen_i, sen_j]$.
    b. If $sim(head, sen_x) < sim(tail, sen_x)$, then put $sen_x$ behind the tail and change the $sen_x$ status to be tail → $ord = [sen_i, sen_j, sen_x]$.
5. Repeat steps 3-4 until the entire sentences are in the ordering result.
```

Algorithm 2. Sentences Ordering Type B Algorithm

```
1. From sentences in the candidate summary, choose two sentences ($sen_i$ and $sen_j$) which has highest similarity ($sim(sen_i, sen_j)$) and then make it as initialization of ordering result → $ord = [sen_i, sen_j]$.
2. Choose the other sentence ($sen_k$) which has highest similarity if paired with one of sentence in ordering result ($sen_i$or $sen_j$).
3. Do one of following conditional:
    a. If $sim(sen_i, sen_k) \geq sim(sen_j, sen_k)$, then put $sen_k$ beside $sen_i$ and set $sen_j$ and $sen_k$ status as head and tail, respectively → $ord = [sen_j, sen_i, sen_k]$.
    b. If $sim(sen_i, sen_k) < sim(sen_j, sen_k)$, then put $sen_k$ beside $sen_j$ and set $sen_i$ and $sen_k$ status as head and tail, respectively → $ord = [sen_i, sen_j, sen_k]$.
4. For each sentence which has not in ordering result, choose a sentence ($sen_x$) which has highest similarity if paired with tail.
5. Put $sen_x$ behind the tail and change the $sen_x$ status as tail.
6. Repeat steps 4-5 until the entire sentences are in the ordering result.
```

ers to understand the information if its sentences are ordered such as two adjacent sentences discuss similar content or topic. It has same principle with topical closeness approach that has been presented in [6]. The closeness between sentence's topics can be considered using similarity value between the sentences. The greater similarity between adjacent sentences reflects that they have similar contents or topics.

Based on the description we can make conclusion that a good summary is a summary with high coherences degree between its adjacent sentences. However, a good summary have to presents the information about its original documents contents to readers in simple form (i.e. the summary has a little number of sentences). Therefore, the summary's coherence value $f_{coherence}$ in this paper is formulated as mean value of adjacent summary's sentences similarities as shown in equation(8). The $U_p^{ord}(t)$ in equation(8) denotes the ordered form of vector solution for the p-th candidate summary on t-th generation and $u_{p,n}^{ord}(t)$ denotes the n-th component of $U_p^{ord}(t)$. Process to generate this vector will be described in the next section.

In order to improve the coherences among summary's sentences, the sentences ordering process is performed. In this paper we proposed two types of sentences ordering algorithm as described in Algorithm 1 and 2. The proposed algorithms are inspired from topical closeness approach that had been presented in [6]. The first type (Type A) is an algorithm that maximizes similarity between adjacent sentences. Whereas the second type (Type B) is an algorithm which emphasizes two sentences with most similar topic should be at the beginning of summary's paragraph.

Example. Let *S1*, *S2*, *S3*, *S4*, and *S5* as five summary's sentences which will be ordered by sentences ordering algorithm type A and B. Assume that they have similarities as shown in Figure 1. Their ordering processes using sentences ordering algorithm type A and B are shown in Table 1.

Based on the algorithms, pair of sentences which have highest similarity are chosen as the initial of sentences ordering result. Therefore pair of *S3* and *S5* which has the highest similarity (0.9) is chosen on the first iteration in each algorithm. In algorithm type B, after the initial sentences are chosen, each sentence is labeled as head and tail. Therefore on this iteration *S3* and *S5* are labeled as head and tail, respectively.

On the second iteration, *S1* is chosen to pair

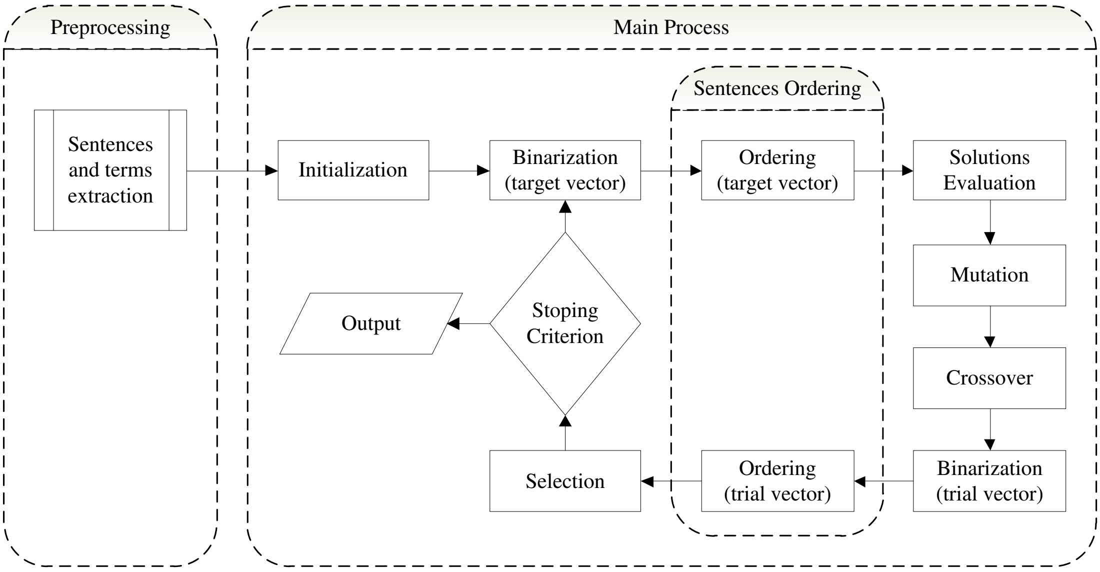

Figure 2. CoDiCo method flowchart.

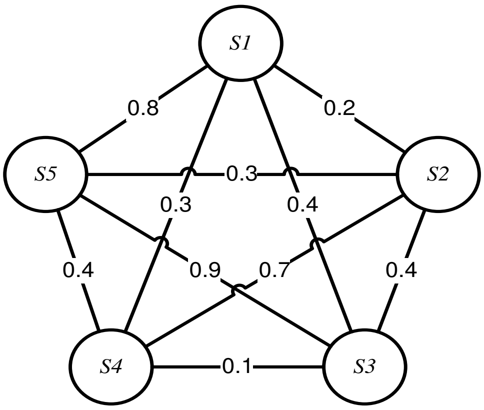

Figure 1. Example of sentences similarities.

with *S5* because it has higher similarity (0.8) than the pair of *S3* with *S1* or *S2* (0.4). Using the same rule in algorithm type A, *S2* and *S4* are chosen to put in front of *S3* on the next iterations. But in algorithm type B, *S4* and *S2* are put behind *S1* since *S1* become the tail and the algorithm only pairs remaining sentences with tail after the second iteration by considers their similarities.

Summary's Quality Factors Optimization

The coverage, diversity, and coherence optimization process in our proposed method consists of preprocessing and main process phase. The main process implements self-adaptive differential evolution (SaDE) algorithm inspired from [4] with the additions of the sentences ordering phase. For the convenience, we denote our proposed method as CoDiCo method which stands for three factors that we would be optimized i.e. coverage, diversity, and coherence. Figure 2 depicts the flowchart of CoDiCo method.

TABLE 1
EXAMPLE OF ORDERING PROCESS USING PROPOSED SENTENCES ORDERING ALGORITHM

Iteration	Ordering Process	
	Type A	Type B
1	*S3-S5*	*S3-S5*
2	*S3-S5-S1*	*S3-S5-S1*
3	*S2-S3-S5-S1*	*S3-S5-S1-S4*
4	*S4-S2-S3-S5-S1*	*S3-S5-S1-S4-S2*

Preprocessing Phase

Preprocessing phase is a step to prepare the data which would be used in main process. In this step there are some processes, i.e.: 1) sentences extraction; 2) sentences normalization; 3) distinct terms extraction; 4) term weights matrix preparation; 5) sentences similarity matrix preparation.

Sentences extraction is a process to take each sentence from documents that have same topic in dataset. The process will produce *N* sentences. Each extracted sentence sen_n is represented as a single line of data in sentences list D such that $D = [sen_1, \dots, sen_N]$.

After the extraction process, each sentence sen_n is normalized into sen_n^{norm} using stop-word removal, punctuation removal, and stemming process. We use 571 stop-words from Journal of Machine Learning Research stop-word list[1] for the

[1] http://jmlr.org/papers/volume5/lewis04a/a11-smart-stop-list/english.stop

stop-word removal process. For the stemming process, we use Porter Stemmer algorithm[2].

On the next step we perform distinct terms extraction from each sen_n^{norm}. This process produces M distinct terms. Each extracted term $term_m$ is stored into terms list T such that $T = [term_1, \ldots, term_M]$.

Based on N normalized sentences and M distinct terms, we generate a terms weight matrix W which has $N \times M$ dimensions. Each component in W stores the term's weight of $term_m$ in normalized sentence sen_n^{norm} (w_{nm}). The weights calculation is conducted using TF-ISF scheme in equation(1) and equation(2).

Each term's weight then used to calculate the sentences similarity. Similarity value between sen_i^{norm} and sen_j^{norm} for $i, j = [1, \ldots, N]$ can be calculated using cosine measure scheme in equation(3). This process will produce a sentences similarity matrix that has $N \times N$ dimensions.

Main Process Phase

The main steps in main process phase of CoDiCo method as shown in Figure 2 consist of initialization, binarization, ordering, evaluation, mutation, crossover, stopping criterion, and output steps. Binarization and ordering steps can be divided into two phases, i.e. binarization and ordering for target vectors (i.e. solution vectors which generated by initialization and selection steps)

and bina-rization and ordering for trial vectors (i.e. solution vectors which generated by crossover step). The brief descriptions for each step is describes in the next subsection.

Initialization

Initialization is a step to provide a set of solutions U that would be used to find the optimal solution of summarization. Let P and t denote the number of generated solutions and the current generation, respectively, such that $U(t) = [U_1(t), \ldots, U_P(t)]$ for $t = 0$. Each solution in U is referred as a target vector. Each target vector $U_p(t)$ for $p = [1, \ldots, P]$ is represented as a vector which has N components such that $U_p(t) = \left[u_{p,1}(t), \ldots, u_{p,N}(t)\right]$ where $u_{p,n}(t)$ denotes the n-th component in p-th target vector.

Each target vector's component in this step $u_{p,n}(0)$ is randomly initialized by a real-value between specified lower bound u_{min} and upper bound u_{max}. The formulation to initialize $u_{p,n}(0)$ is shown in equation(9):

$$u_{p,n}(0) = u_{min} + (u_{max} - u_{min}) \cdot rand_{p,n}, \qquad (9)$$

where $rand_{p,n}$ denotes a uniform random value between 0 and 1 for the nth component in p-th target vector [4].

Binarization

Binarization is a step to encode real-value of $u_{p,n}(t)$ into binary-value. The binary-values are used to indicate the sentences from D which used as sentences in the p-th candidate summary $S_p(t)$. If $u_{p,n}(t) = 1$, then it indicates that the sen_n in D is selected as sentence in the $S_p(t)$. Otherwise, if $u_{p,n}(t) = 0$, then it indicates that the sen_n in D is not a sentence in the $S_p(t)$.

Alguliev et.al. [4] describes that encoding pro-cess of real-value $u_{p,n}(t)$ into binary-value $u_{p,n}^{bin}(t)$ can be performed by comparing $rand_{p,n}$ value with sigmoid value of $u_{p,n}(t)$. The formulation for this process is shown in equation(10) and equation(11):

$$u_{p,n}^{bin}(t) = \begin{cases} 1, & \text{if } rand_{p,n} < sigm\left(u_{p,n}(t)\right) \\ 0, & \text{otherwise} \end{cases}, \qquad (10)$$

$$sigm(A) = \frac{1}{1 + e^{-A}}. \qquad (11)$$

The $rand_{p,n}$ in this step has same value with the one which have been used in initialization step.

Ordering

In this step, $N_p^S(t)$ sentences for each $S_p(t)$ derived from $U_p(t)$ solution are ordered using sentences ordering algorithm which described in Subsection 2.3. Ordered form of $U_p(t)$ is stores in ordering-solution vector $U_p^{ord}(t)$ which has $N_p^S(t)$ components $u_{p,y}^{ord}(t)$ such that,

$U_p^{ord}(t) = \left[u_{p,1}^{ord}(t), \ldots, u_{p,N_p^S}^{ord}(t)\right]$

The $u_{p,y}^{ord}(t)$ component stores sentences index which include as summary's sentences in $S_p(t)$ (i.e. $u_{p,y}^{ord}(t) = [1, \ldots, N]$).

Solutions Evaluation

The evaluation step is used to calculate fitness value for each summarization solution. Evaluations are performed for each $U_p(t)$ which has been encoded to binary form ($U_p^{bin}(t)$) and ordered form ($U_p^{ord}(t)$). Based on our purpose in this paper, calculation for fitness value of $U_p(t)(fit\left(U_p(t)\right))$ is conducted by considering the three factors of summary's quality, i.e. the its coverage, diversity, and coherence values. The formulation is shown in equation(12).

[2] http://tartarus.org/martin/PorterStemmer/

$$fit\left(U_p(t)\right) = \frac{f_{coverage}\left(U_p^{bin}(t)\right)}{f_{diversity}\left(U_p^{bin}(t)\right)} \cdot f_{coherence}\left(U_p^{ord}(t)\right). \tag{12}$$

$$U_{gbest}(t) = \begin{cases} U_{best}(t), & \text{if } fit\left(U_{best}(t)\right) > fit\left(U_{gbest}(t-1)\right) \\ U_{gbest}(t-1), & \text{otherwise} \end{cases}. \tag{13}$$

$$V_p(t) = U_p(t) + \left(1 - F(t)\right) \cdot \left(U_{gbest}(t) - U_{p1}(t)\right) + F(t) \cdot \left(U_{best}(t) - U_{p1}(t)\right). \tag{14}$$

$$v_{p,n}(t) = \begin{cases} 2u_{min} - v_{p,n}(t), & \text{if } v_{p,n}(t) < u_{min} \\ 2u_{max} - v_{p,n}(t), & \text{if } v_{p,n}(t) > u_{max}. \end{cases} \tag{15}$$

The best and the worst solutions on current generation can be determined using each solution's fitness value. The best solution on current generation (local best) $U_{best}(t)$ is a target vector that has the highest fitness value. Otherwise, the worst solution on current generation $U_{worst}(t)$ (local worst) is a target vector that has the lowest fitness value. In this step we also can update the global best $U_{gbest}(t)$ i.e. the best solution until current generation using the rule which formulated in equation(13).

Mutation

Mutation is a step to generate mutant vectors set V from target vectors set U. Mutation process of $U_p(t)$ is conducted by involving $U_{gbest}(t)$ vector, $U_{best}(t)$ vector, a randomly selected vector $U_{p1}(t)$ where $p1 = [1,\ldots,P]$ and $p1 \neq p$, and a mutation factor for current generation $F(t)$. The formulation to generate p-th mutant vector on current generation $V_p(t)$ is shown in equation(14), whereas the formulation to calculate the $F(t)$ value is shown in equation(16):

$$F(t) = e^{-2t/t_{max}}. \tag{16}$$

In equation(16) t_{max} denotes maximum generateon which specified in initialization step [4].

One or more $V_p(t)$ components $v_{p,n}(t)$ have a probability to violate the boundary constraints. Its values can be less than u_{min} or greater than u_{max}. Each $v_{p,n}(t)$ which its value violates the boundary constraints have to been reflected back. The rules to reflect back the $v_{p,n}(t)$ value is formulated in equation(15).

Crossover

Crossover is a step to generate trial vectors set Z. Each trial vector $Z_p(t)$ has N components $z_{p,n}(t)$, which its value is derived from the value of $u_{p,n}(t)$ or $v_{p,n}(t)$ [4]. The purpose of this operation is to increase the diversity of solution vectors in order to expand the search space.

Alguliev et.al. [4] describes that to generate the $Z_p(t)$ vector, relative distance between $U_p(t)$ vector and $U_{best}(t)$ vector $RD_p(t)$ has to be calculated first. The $RD_p(t)$ then used to calculate the crossover rate $CR_p(t)$. Equation(17-19) shows the formulation to calculate the $RD_p(t)$ and $CR_p(t)$.

The rule to determine trial vector component $z_{p,n}(t)$ value is formulated in equation(20). In equation(20) k is a randomly selected integer value for $k = [1,\ldots,N]$. It ensures that at least one component of trial vector is obtained from the mutant vector. It will ensure that the solutions on the next generation have differences with the solutions on current generation [4].

$$RD_p(t) = \frac{fit\left(U_{best}(t)\right) - fit\left(U_p(t)\right)}{fit\left(U_{best}(t)\right) - fit\left(U_{worst}(t)\right)}. \tag{17}$$

$$CR_p(t) = \frac{2tanh\left(2RD_p(t)\right)}{1 + tanh\left(2RD_p(t)\right)}. \tag{18}$$

$$tanh(A) = \frac{e^{2A}-1}{e^{2A}+1}. \tag{19}$$

$$z_{p,n}(t) = \begin{cases} v_{p,n}(t), & \text{if } rand_{p,n} \le CR_p \text{ or } n = k \\ u_{p,n}(t), & \text{otherwise} \end{cases}, \tag{20}$$

$$U_p(t+1) = \begin{cases} Z_p(t), & \text{if } fit\left(Z_p(t)\right) \ge fit\left(U_p(t)\right) \\ U_p(t), & \text{otherwise} \end{cases}. \tag{21}$$

Selection

Selection is a step to generate a novel target vectors set for the next generation $U(t+1)$. The vectors are derived from $U(t)$ vectors and $Z(t)$ vectors which have the best fitness value [4]. In other words only the best solution for each pair is survived from this operation. It ensures that the searching of optimal solution is always approach to the best solution until the last iteration. The rule to

TABLE 2
ROUGE SCORES COMPARISON OF EACH TESTED METHODS

Methods	ROUGE Score				Average
	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-SU	
OCDsum-SaDE	0.3701	0.0794	0.3424	0.1254	0.2293
CoDiCo-A (without threshold)	0.5144	0.1395	0.4820	0.2256	0.3404
CoDiCo-B (without threshold)	0.5279	0.1524	0.4933	0.2351	0.3522
CoDiCo-A ($T_{sim} = 0.9$)	0.5118	0.1443	0.4749	0.2172	0.3370
CoDiCo-B ($T_{sim} = 0.9$)	0.5163	0.1525	0.4785	0.2304	0.3444
CoDiCo-A ($T_{sim} = 0.8$)	0.5322	0.1473	0.4963	0.2310	0.3517
CoDiCo-B ($T_{sim} = 0.8$)	0.5464	0.1747	0.5127	0.2640	0.3744
CoDiCo-A ($T_{sim} = 0.7$)	0.5620	0.1611	0.5205	0.2674	0.3777
CoDiCo-B ($T_{sim} = 0.7$)	0.5296	0.1448	0.4923	0.2434	0.3525

choose the p-th target vector on the next generation $U_p(t+1)$ is formulated in equation(21).

Stopping Criterion

In this step, the iteration of optimal solution searching process is determined to be stopped or not. The stopping criterion in this paper is uses a specified number of generation. If the iteration has reached the maximum generation t_{max}, then the iteration is stopped. Otherwise, if the iteration has not reached the t_{max}, then the iteration is continued.

Output

This step is the final step in main process of CoDiCo method. In this step, the global best solution of summarization on the last generation U_{gbest} (t_{max}) has been acquired. Its binary form U_{gbest}^{bin} (t_{max}) denotes the sentences index in D which has selected as summary's sentences, whereas its ordered form $U_{gbest}^{ord}(t_{max})$ stores the order of the summary's sentences index. Furthermore a sentences set which indicated in $U_{gbest}(t_{max})$ are returned as the summary.

3. Results and Analysis

In this paper we use Text Analysis Conference (TAC) 2008 dataset from National Institute of Standards and Technology (NIST)[3] to test our CoDiCo method. This dataset provides articles that classified into some topics and coherent summaries which created manually by human for each topic. We choose 15 topics for the testing. Each topic contains 10 documents that would be summarized.

The experiments are performed using Matlab R2013a and run on Microsoft Windows platform. We test both the proposed sentences ordering algorithm type A and type B using CoDiCo method. We represented these methods as CoDiCo-A and CoDiCo-B, respectively. In order to compare the summarization results from CoDiCo method with another multi-document summarization method that considers coverage and diversity factors only, we use the OCDsum-SaDE method from [4].

We also test both of our proposed sentences ordering algorithm by involving a threshold T_{sim} in sentences similarity value in order to evaluate the impact of similarity between summary's sentences toward the optimal summary's solution. We use three threshold values, i.e. 0.7, 0.8, and 0.9. In this scenario, the sentences ordering process exclude every pair of sentences that have similarity value greater than or equal to T_{sim} value. The excluding pair of sentences will not be chosen as adjacent sentences in summary's solutions.

Both of our proposed method (CoDiCo) and the compared method (OCDsum-SaDE) use four specified parameters in the initialization state i.e. population size (P), maximum generation (t_{max}), lower bound (u_{min}), and upper bound (u_{max}) which sets to 20, 500, -5, and 5, respectively. Those parameters values assign based on heuristic choices. After the multiple-documents summarizations were processed using the tested methods, we get the summarization results as many as selected topics. Therefore, we have 15 summaries from 15 selected topics for each method.

The testing results are evaluated using Recall-Oriented Understudy of Gisting (ROUGE) method [10]. This method compares candidate summaries (i.e. summaries generated by proposed and compared methods) with reference summaries (i.e. summaries that created manually by human which provided in TAC 2008 dataset). There are 4 type of ROUGE method which is used to evaluate our experiments i.e. ROUGE-1, ROUGE-2, ROUGE-L, and ROUGE-SU.

ROUGE-1 and ROUGE-2 are variants of ROUGE-N that consider n-gram recall between summarization result from candidate summary and reference summary for n assigned by 1 and 2.

[3] http://www.nist.gov/tac/2008/summarization/

TABLE 3
AVERAGES COHERENCE VALUE COMPARISON FROM EACH TESTED METHOD

Methods	Average of coherences value
OCDsum-SaDE	0.005
CoDiCo-A (without threshold)	0.193
CoDiCo-B (without threshold)	**0.206**
CoDiCo-A (T_{sim} = 0.9)	0.151
CoDiCo-B (T_{sim} = 0.9)	0.167
CoDiCo-A (T_{sim} = 0.8)	0.148
CoDiCo-B (T_{sim} = 0.8)	0.160
CoDiCo-A (T_{sim} = 0.7)	0.140
CoDiCo-B (T_{sim} = 0.7)	0.145

It is computed by divides the maximum number of n-grams co-occurring in candidate summary and set of reference summary with total sum of the number of n-grams occurring at the reference summary. ROUGE-L is a ROUGE method that considers about longest common subsequence (LCS) between candidate summary and reference summary. It is computed as the ratio between LCS's length with reference summary's length. In other hand ROUGE-SU considers the unigram value on candidate summary and reference summary as counting unit [4,10]. The formulas and complete explanation about usage of ROUGE method can be read in [10].

ROUGE score for CoDiCo-A, CoDiCo-B, and OCDsum-SaDE methods are presented in Table 2. It shows the comparison of ROUGE scores among each tested methods. The highest ROUGE scores for each ROUGE type are indicated by bolded text. We also evaluate our proposed method by considering the averages of summary's coherence value which generated by each tested method. The comparison of averages coherence value from tested methods is shown in Table 3. The highest value is indicated by bolded text.

Discussion

Series of experiment has been conducted to evaluate our proposed method (i.e. CoDiCo-A and CoDiCo-B) in comparison with compared method (OCDsum-SaDE). Based on the evaluation results as shown in Table II, we know that the CoDiCo-A method using $T_{sim} = 0.7$ has higher ROUGE score on ROUGE-1, ROUGE-L, and ROUGE-SU than the other methods. Whereas in ROUGE-2 can be shown that CoDiCo-B method using $T_{sim} = 0.8$ has higher score than the others. From Table II we also know that the lowest average ROUGE score of CoDiCo method is 0.3370 which obtained by CoDiCo-A using $T_{sim} = 0.9$ and the highest average ROUGE score is 0.3777 which obtained by CoDiCo-A using $T_{sim} = 0.7$. Whereas the averages ROUGE score of OCDsum-SaDE method only reached 0.2293. It means all of CoDiCo method variants have better performance than the compared method.

It should be noted that in CoDiCo method, by considering the coherences of sentences while selecting the best solution will adjust the coverage and diversity factors simultaneously to find the optimal solution. It will produce different summary compared with method that only considers the coverage and diversity factors. But the summary is more similar with summary that created manually by human. It causes the ROUGE scores of CoDiCo method are greater than ROUGE scores of compared method.

By comparing the averages ROUGE score for each CoDiCo method with the average ROUGE score of OCDsum-SaDE method, we know that CoDiCo methods have averages RO-UGE score in range 46.97-64.71% higher than the averages ROUGE score of compared method. It shows that the multi-document summarization method that considers coverage, diversity, and coherence simultaneously can produce better summary than summarization method that considers coverage and diversity only.

Based on the evaluation of averages of summary's coherence value that shown in Table 3, we know that CoDiCo-B method without threshold reaches the average coherence value higher than the others do. We also know that the lowest average coherence value among the variants of CoDiCo method is 0.145 which obtained by CoDiCo-B method using $T_{sim} = 0.7$. Nevertheless, the value is higher than the average coherence value of OCDsum-SaDE. If we compare it with OCDsum-SaDE method, CoDiCo-B without threshold can produce summary with average coherence value about 41.2 times higher than the OCDsum-SaDE method, whereas CoDiCo-B method using $T_{sim} = 0.7$ can reach average coherence value about 29 times higher than OCDsum-SaDE method. It shows that the CoDiCo method, which involves ordering step in optimization process, can produce summary with better coherences or smoother connectivity among sentences than the other method which does not consider the ordering of summary's sentences.

The comparison of two proposed sentences ordering algorithm with same threshold value using their ROUGE scores shows that CoDiCo-B is better than CoDiCo-A. As shown in Table 2, CoDiCo-B has higher average ROUGE score than CoDiCo-A when do not using a threshold, using $T_{sim} = 0.9$, and using $T_{sim} = 0.8$. Whereas CoDiCo-A only has higher average ROUGE score than

CoDiCo-B when using $T_{sim} = 0.7$.

If we compare the averages coherence value between both of sentences ordering algorithms as shown in Table 3, we know that CoDiCo-B has higher average coherence value than CoDiCo-A when using all variants of threshold value. This performance comparison indicates that ordering sentences strategy which arrange sentences by higher similarity on the beginning of paragraph is better than ordering sentences strategy which maximize similarity between two sentences in the summarization result.

4. Conclusion

This paper proposes and describes a new method on multi-document summarization by considering the coverage, diversity, and coherence factors among the summary's sentences simultaneously. The proposed method tries to improve connectivity among summary's sentences in order to enhance the readability of summary by adding sentences ordering phase in summary optimization process. Thus, the process of sentences ordering is no longer relying on the summary.

The experimental results show that the multi-document summarization method that considers the coverage, diversity, and coherence factors in summary simultaneously is able to provide better summary than other methods, which only considers the coverage and diversity of summary. In our experiments it has performances 46.97-64.71% better than the compared method. In addition to the consideration of the coherence factor in summary, method that consider this factor can provide summary with better readability compared with another method that do not consider this factor. In our experiments, it provides summaries that have readability rate 29-41.2 times better than the other method. This research can be developed further. Further development can be done by considering the other approaches in the sentences ordering algorithm in order to improve the readability of the summary.

References

[1] R. Ferreira, L.S. Cabral, F. Freitas, R.D. Lins, G.F. Silva, S.J. Simske, & L. Favaro, "A Multi-Document Summarization System Based on Statistics and Linguistic Treatment", *Expert Systems with Applications*, vol. 41, pp. 5780-5787, 2014.

[2] A. Baralis, L. Cagliero, S. Jabeen, A. Fiori, & S. Shah, "Multi-Document Summarization Based on The Yago Ontology", *Expert Systems with Applications*, vol. 40, pp. 6976-6984, 2013.

[3] R.M. Aliguliyev, "A New Sentence Similarity Measure and Sentence Based Extractive Technique for Automatic Text Summarization", *Expert Systems with Applications*, vol. 36, pp. 7764-7772, 2009.

[4] R.M. Alguliev, R.M. Aliguliyev, & N.R. Isazade, "Multiple Documents Summarization Based on Evolutionary Optimization Algorithm", *Expert Systems with Applications*, vol. 40, pp. 1675-1689, 2013.

[5] R. Barzilay, N. Elhadad, & K. McKeown, "Inferring Strategies for Sentence Ordering in Multidocument News Summarization", *Journal of Artificial Intelligence Research*, vol. 17, pp. 35–55, 2002.

[6] D. Bollegala, N. Okazaki, & M. Ishizuka, "A Preference Learning Approach to Sentence Ordering for Multi-Document Summarization", *Information Sciences*, vol. 217, pp. 78-95, 2012.

[7] P. Sukumar, & K.S. Gayathri, "Semantic Based Sentence Ordering Approach for Multi-Document Summarization", *International Journal of Recent Technology and Engineering (IJRTE)*, vol. 3, no. 2, pp. 71-76, 2014.

[8] L. Huang, Y. He, F. Wei, & W. Li, "Modeling Document Summarization as Multi-objective Optimization", *Third International Symposium on Intelligent Information Technology and Security Informatics (IITSI)*, 2010.

[9] D.R Radev, H. Jing, M. Stys, & D. Tam, "Centroid-based Summarization of Multiple Documents", *Information Processing and Management*, vol. 40, pp. 919-938, 2004.

[10] C.Y Lin, "ROUGE: A Package for Automatic Evaluation Summaries", *in Proceedings of the workshop on text summarization branches out*, pp. 74–81, 2004.