Review assessment support in Open Journal System using TextRank

S R Manalu¹, Willy¹, A M Sundjaja², Noerlina²

¹ School of Computer Science, Bina Nusantara University, Jl. K. H. Syahdan No. 9, Kemanggis, Palmerah, Jakarta, Indonesia
² School of Information System, Bina Nusantara University, Jl. K. H. Syahdan No. 9, Kemanggis, Palmerah, Jakarta, Indonesia

E-mail: smanalu@binus.edu

Abstract. In this paper, a review assessment support in Open Journal System (OJS) using TextRank is proposed. OJS is an open-source journal management platform that provides a streamlined journal publishing workflow. TextRank is an unsupervised, graph-based ranking model commonly used as extractive auto summarization of text documents. This study applies the TextRank algorithm to summarize 50 article reviews from an OJS-based international journal. The resulting summaries are formed using the most representative sentences extracted from the reviews. The summaries are then used to help OJS editors in assessing a review’s quality.

1. Introduction
Open Journal System (OJS) is an open-source journal management platform, created by the Public Knowledge Project in 2002 as part of the research program at the University of British Columbia, under the direction of John Willinsky [1]. OJS was designed to provide an easy to use, technical infrastructure of the entire journal publishing workflow, such as article submissions, reviewing to editing work, and online publication. In addition, OJS also demonstrate how the cost of journal publishing can be reduced significantly while maintaining a quality publication. This study focuses on improving the reviewing procedure of OJS. Particularly, it is an attempt to develop a decision support for the reviewer ratings feature in OJS.

In this study, we propose a summarization technique to support the review assessment procedure in OJS. To illustrate, when an article is submitted to OJS, the editor will take a look to see if it is suitable for the journal and if it is, the editor will proceed to assign a relevant reviewer to evaluate the paper. After the review process is completed, the editor will assign a rating to indicate the review’s quality, based on various parameters defined by each journal. By employing the TextRank algorithm, an automatic summary of each review can be acquired to give the editor a quick grasp of their content.

2. Methodology
Text summarization is a process of creating a condensed form of text document, containing few important sentences that represent the whole document. TextRank [2] is an unsupervised, graph-based algorithm for automatic text summarization. It is an extractive summarization technique where several sentences will be extracted from the original document to create the summary, contrary to the abstractive summarization technique that produces a summary containing rephrased words or phrases,
based on the concept or idea in the document. Currently, the extractive technique is the standard in auto summarization because it is easier to implement. TextRank uses graph-based algorithm to represent the text document by the calculating the importance of each sentence. The best scoring sentences will then be used to form the summary.

2.1. Similar methods

Some of the successful methods of extractive summarization uses supervised approach [3], by learning from a large number of labelled training data to determine how to make a good summary, such as [3,4]. However, the drawback of using such methods is the inability to adapt to a new domains or languages, as it needs an entirely new training data each time. TextRank uses an unsupervised approach where no training data is required to produce a summary, independent of the language used.

In a graph-based summarization, text is represented as a graph where nodes represent words or sentences, and edges represent the semantic relations between nodes. Graph based ranking model takes into account all the information within a graph, rather than relying only on each local node information [3]. One graph-based algorithm for summarization is LexRank [6], which uses cosine similarity and TF-IDF to calculate the weight of edges, applied to summarize multiple documents. In comparison, TextRank’s measurement is based on the number of words two sentences have in common, applied to summarize single documents or to extract important keywords. While graph-based ranking model offers good result, the time it takes to complete the algorithm is proportional to the complexity of the graph, therefore it might not be a very efficient algorithm, timewise [7].

2.2. TextRank for sentence extraction

TextRank is a variant of Google’s PageRank algorithm [8], which uses the concept of recommendation. When a node is connected to other node, it is said that the node is recommending that other node. As the number of recommendations a node has increases, the importance of that node also increases. In addition, the importance of each node also influences the importance of its recommendations. Formally described, the PageRank algorithm used to score of a node \( V_i \) is as follows:

\[
S(V_i) = (1 - d) + d \sum_{j \in \text{In}(V_i)} \frac{1}{\text{Out}(V_j)} S(V_j)
\]  

(1)

where the damping factor \( d \) is set between 0 and 1, used to give a random probability of a node jumping to another random node in the graph. To illustrate, assume a random surfer that is given a random web page and keeps opening links on provided on the page, until the surfer becomes bored and decided to open a new random page. The algorithm takes into account the random factor by utilizing \( d \), usually set to 0.85 according to Brin and Page [8]. At first, each node in the graph will be assigned the value of 1. Every node will then be computed iteratively using the PageRank algorithm until they reach convergence. Convergence will be achieved once the error rate of each node falls below a given threshold. The error rate \( \Delta S \) can be approximated by calculating the difference between two successive score computation of a node \( V_i \):

\[
\Delta S = S^{k+1}(V_i) - S^k(V_i)
\]  

(2)

where \( k+1 \) is the current algorithm iteration. In this implementation, an undirected weighted graph is used. Unlike the PageRank algorithm, TextRank takes into account the strength of connections (edge) between each node. Therefore, a new formula that integrates weighted edges will be used, as follows:

\[
WS(V_i) = (1 - d) + d \sum_{V_j \in \text{In}(V_i)} \sum_{V_k \in \text{Out}(V_j)} \frac{W_{jk}}{W_{kj}} WS(V_j)
\]  

(3)

To apply the graph-based ranking model in natural language sentence extraction, a graph representation of the text document will be built. The graph will have nodes containing text units and
edges containing the relationship value between every text unit pairs. Depending on the implementation, text units could mean words, sentences, n-gram sentences, or others. The relationship between text units will also be dependent on the algorithm used to calculate it. In this study, sentence similarity is determined as the overlap of two sentences \( S_i \) and \( S_j \), formally defined as follows:

\[
\text{Similarity}(S_i, S_j) = \frac{\left| \{w_k \mid w_k \in S_i \land w_k \in S_j\} \right|}{\log(|S_i|) + \log(|S_j|)}
\]  

(4)

where a sentence is represented as a set of words: \( S_i = w'_1, w'_2, \ldots, w'_n \). The end result is a complete graph with weighted edges and nodes, ready to be extracted to create the summary. One advantage of using a ranking model is the capability to freely adjust the summary length, as sentences used in the summary are taken from the ordered list of weight-sorted sentences. Followings are the core steps used to implement the TextRank algorithm:

- Preprocess the text document as clean text units and build the nodes of the graph using those units. In this case, each node contains one sentence.
- Decide the form of relationship between each node in the form of edges, and calculate each their weight. In this case, the weight is represented by the overlap between two sentences.
- Iterate the TextRank algorithm to calculate the weight of nodes recursively until convergence is achieved or until the iteration limit is passed.
- Extract the values of each node as sentences, sorted by their TextRank score.

3. Discussion & Evaluation

In this study we use a collection of 50 article reviews, taken with permission from Binus Business Review, an international journal hosted by the Research and Technology Transfer Office of Bina Nusantara University. Using Python as the programming language to implement the TextRank algorithm, a typical process of the summarization technique presented in Section 2 will be demonstrated. First, one of the review will be preprocessed to segmentate the review text into sentences, as shown in figure 1.

**Figure 1.** Segmented review.
Next, a graph model will be built, where a node will be inserted to the graph for each sentence in the text. The nodes will be given an arbitrary initial score of 1. Each node will then be linked to each other. After that, the weight of each edge will be determined using equation (4). The more similar two sentences get, the higher the value that will be assigned to them. Edges with zero similarity value will be eliminated. The resulting graph is presented in figure 2.

![Figure 2. A highly connected graph consisting of nodes and edges. Some node pairs don't have an edge between them (0-14, 0-7, etc.) because their similarity value is zero. In total, 93 edges are built.](image)

Once the graph is properly built, the score of each node can be calculated iteratively using equation (3), taking into account the TextRank score of each neighbour node and their similarity values. The more adjacent edges a node has, the higher the score of that node will be. In addition, the more high scoring neighbour a node has, the higher also the score of that node will be. The process ends when the error rate (2) falls below the threshold value (0.0001) as it achieves convergence. Figure 3 and figure 4 illustrate the iterative process.

![Figure 3. The graph after the second iteration of TextRank, consisting of nodes and their corresponding scores. The value was obtained after using equation (3) to calculate each node’s value.](image)

![Figure 4. The graph after the third iteration of TextRank. The algorithm will keep repeating and readjusting the value of each node until convergence is achieved.](image)
Figure 3 and figure 4 shows the score of each node during the second and third iterations, respectively. As seen above, the error rate of several nodes between the second and third iterations are still higher than the threshold value. In this case, once the error rate of each node fall below 0.0001, the algorithm will finish. Finally, to create the summary TextRank will extract the best scoring nodes from the graph. In this case, three sentences will be extracted. For this review a total of 12 iterations are run before the graph achieved convergence in Figure 5. The final result is presented in Figure 6.

![Figure 5](image)

**Figure 5.** The final graph after the 12th TextRank iteration and reaching convergence. Node 8, 9, and 12 will be extracted to form the summary.

![Figure 6](image)

**Figure 6.** The final TextRank summary of the sample review.

After the required number of sentences is extracted, they will be presented in the summary with the same order they appeared in the original document, regardless of their TextRank score. The summary will then be ready to be read by the editor in charge of assessing the review, to provide a quick insight as to the quality of the reviews. The editor will then set the review’s rating on a 1 to 5 scale. The review rating will impact a reviewer’s total rating.

4. Conclusions

This paper proposes a way to develop a review assessment support for the OJS, by implementing TextRank to auto summarize reviews and giving editors a quick insight to the content of each review, thus helping them assess the reviews with less resources and time required. TextRank works by extracting the most representative sentences in a document, measured by the similarities of each sentences. TextRank is easy to implement as it doesn’t require a lot of computational power nor a deep linguistic knowledge, allowing it to be used in other languages or document types.

This study suggests that TextRank as a summarization technique could be applied for various purposes. In our future work, we will develop additional review assessment features which will be used in conjunction with TextRank to provide automatic review score recommendations. We are also planning to develop more usability support for the OJS, whether it is to improve the reviewing process or other parts of the publishing procedure.
References

[1] Edgar B D and Willinsky J 2010 A survey of the scholarly journals using Open Journal Systems Sch. Res. Commun. 1 1–22
[2] Mihalcea R and Tarau P 2004 TextRank: bringing order into texts Proc. EMNLP 85 404–411
[3] Mihalcea R 2005 Language independent extractive summarization Evaluation 49–52
[4] Hirao T, Sasaki Y, Isozaki H and Maeda E 2002 Ntt’s text summarization system for duc-2002 Proc. Doc. Underst. Conf. 2002 104–7
[5] Ko Y and Seo J 2008 An effective sentence-extraction technique using contextual information and statistical approaches for text summarization Pattern Recognit. Lett. 29 1366–71
[6] Erkan G and Radev D R 2004 LexRank: Graph-based lexical centrality as salience in text summarization J. Artif. Intell. Res. 22 457–79
[7] Ferreira R, De Souza Cabral L, Lins R D, Pereira E Silva G, Freitas F, Cavalcanti G D C, Lima R, Simske S J and Favaro L 2013 Assessing sentence scoring techniques for extractive text summarization Expert Syst. Appl. 40 5755–64
[8] Brin S and Page L 1998 The anatomy of a large scale hypertextual Web search engine Comput. Networks ISDN Syst. 30 107–17