Document Similarity Measurement Using Ferret Algorithm and Map Reduce Programming Model

Condro Wibawa, Irwan Bastian, Metty Mustikasari

Department of Information Systems, Faculty of Computer Science and Technology, Gunadarma University, Indonesia

Abstract—Plagiarism of digital documents that happened frequently should be prevented and reduced. In order to measure the degree of document’s similarity, Ferret Algorithm is used on MapReduce and Hadoop framework. This application program is tested using ten pair documents in English and ten pair documents in Indonesian to get the degree of document’s similarity between document’s pairs. In this experiment, it was also tested the accuracy of the system by using several kind of n-grams, that is 1-gram, 2-gram, 3-gram, 4-gram, 5-gram, and 6-gram. In addition, this experiment is conducted on stand-alone computers and Hadoop multi-node cluster system. In order to compare the time taken of the system, it is used nine pairs of documents with various size of documents.

Keywords—Similarity, Ferret Algorithm, MapReduce, Hadoop, Jaccard.

I. INTRODUCTION

The growth of information technology which running fast have many positive and negative impacts. One of the negative impacts is plagiarism of digital documents. It’s because plagiarism can be done easily.

There are many definitions describing plagiarism. According to research resources in plagiarism.org, plagiarism is one of the following activities turning in someone else’s own, copying words or ideas of others without giving credit, failed to put the quote in quotes, giving incorrect information about the source quote, changing words but copying the sentence structure of a source without giving credit, copying so many words or ideas from a source that makes up the majority of our work, whether we give credit or not [5].

Plagiarism can occur in the text, music, images, maps, technical drawings, paintings, etc. Although academics sectors are most remarkable, other sectors such as journalism and information retrieval is also suffering from a serious problem. For example, the Internet is full of information that is redundant and duplicate or near duplicate documents that will decrease the effectiveness of the search engine [6].

In order to rapidly develop large-scale data processing, MapReduce framework is introduced by Google. Mapreduce scalable data processing framework and its open source realization Hadoop, provide and offer effective programming model to manage large documents.

This research proposed a document similarity detection using Ferret algorithm on MapReduce and Hadoop framework. It also covers the Hadoop Distributed File System which is used MapReduce algorithm to manage large amount of data by splitting up datasets across multiple servers and parallely processing each part and then combining the result of each part to produce the final answer.

Using Ferret model as the document similarity index, it can be determined whether the documents is product of plagiarism or not. The programming code will be implemented in stand-alone computer and Hadoop multi-node cluster system to determine which is better in terms of effectiveness and efficiency. This algorithms is complemented with n-gram model and Jaccard Index. The programming language used in this research are Python and mrjob library.

II. METHOD

A. MapReduce Framework

Jayram Chandar [2] define MapReduce as a programming model that is used to process a big data. Whereas according to AnikMomtaz [3] data processing on the MapReduce was performed by distributed and parallel in a cluster. A cluster can consist of one master computer and several slave computers (workers) or only one computer that acts as both a master computer and a slave computer. The number of slave computers is unlimited and hardware specification that used can be different.

In general, the MapReduce process is divided into two processes that is Map Processing and Reduce Processing. Both of these processes are distributed to all slave computers connected in a cluster and run parallely. Map Processing charge of dividing the problem into sub-problems and distribute it to the slave computers. Then, results of the process on a slave computers will be collected by the master computer. It’s call Reduce Process. The results of the Reduce Processing is then delivered to the user as a final output [4].
Map function assigned to read input as a pairs of key/value and also produce output of pairs of key/value.

map: \((\text{key 1, value 1}) \rightarrow [(\text{key 2, value 2})]\)

Furthermore, Reduce function will read the output from Map function and then it will be combined or grouped based on the key. Values that have the same key will be combined into one group. Reduce function also produce the output of pairs of key/value.

reduce: \((\text{key 2, [value 2]})) \rightarrow [(\text{key 3, value 3})]\)

Figure 2 depicts the MapReduce process scheme to count the number of words in a the document:

Hadoop is a Java-based software framework that is used to process big data by using clustering method [7]. Hadoop consists of four main parts: Hadoop Common, Hadoop Distributed File System, Hadoop YARN, and HadoopMapReduce.

MapReduce programming codes that are created using Java can be directly run on Hadoop System. Otherwise, programming codes that are created with other programming language must pass through Hadoop Streaming before processed in Hadoop System. Hadoop Streaming is a Hadoop translator in order to MapReduce function that can be read in Hadoop System[7].

B. Ferret Model

Ferret is very effective in finding plagiarism or copied part in English and other languages, but this approach does not turn into useful in finding semantically similar text written independently. For example, a language processing technologies that purposed is to translate English into Chinese. The method is based on successive token processing; in English tokens are words, whereas in Chinese they are a character or string of characters [1, 3].

Ferret Algorithm divides the documents into trigrams. The set of trigrams then compared to the documents to be compared. The trigrams similarity of the two documents then used to calculate similarity index using Jaccard Index.

Jaccard Index or also known as the Jaccard Similarity Coefficient or Jaccard Coefficient is a statistical calculation that was created by Paul Jaccard and used to compare the degree of similarity and dissimilarity of sample sets. Coefficient Jaccard (similarity index) is denoted by \(J(A,B)\) which is defined as number of intersection divided by number of union in the sample [3].

\[
J(A,B) = \frac{\text{i}}{\text{i} + (\text{a} \times \frac{\text{a}}{\text{A}}) + (\text{b} \times \frac{\text{b}}{\text{B}})}
\]

Where A is determine to number \(n\) of object A, B to object B, i to object which exist in both A and B, a to object which exist in A but not in B, and b to object which exist in B but not in A [1, 3].
III. EXPERIMENTAL RESULT

Hadoop-based MapReduce programming model require more than one computer. It can be a dedicated computers or virtual computers. In this experiments, each computer requires minimum specification, such as 384 MB of memory, Network Interface Card, 5GB free space of Hardisk, Ubuntu Server 12.04 Operating System. It also requires software such as: Java Development Kit, Hadoop-1.0.3, Python-2.7.7, and mrjob-0.4 (python library).

A. Global Program Design

In general, the system flowchart depicts executing the system and selecting two documents to be compared. After that the process continues with executing mapping and reducing process. If the system is executed in stand-alone computer, the document will be processed only in that computer. However, if the system is executed in Hadoop System, the document processing will be continued in Hadoop System. The results of the data processing will be returned to the user as a percentage that indicates the level of document similarity.

![Flowchart of Hadoop System Configuration](image)

The next step is installation of Hadoop and configuration some of Hadoop files, such as: hadoop-env.sh, core-site.sh, mapred-site.sh, hdfs-site.sh, conf/master, and conf/slave. The installation of Hadoop and configuration some of Hadoop files are presented in TABLE II.

<table>
<thead>
<tr>
<th>Computer Name</th>
<th>IP Address</th>
<th>Master Node</th>
<th>Slave Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>master</td>
<td>192.168.0.1</td>
<td>V</td>
<td>V</td>
</tr>
<tr>
<td>slave1</td>
<td>192.168.0.2</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>slave2</td>
<td>192.168.0.3</td>
<td></td>
<td>V</td>
</tr>
</tbody>
</table>

B. Hadoop System Design

Hadoop System consists of several computers that communicate with each other. Therefore, the first step is to set the IP Address and SSH. Setting the IP Address and SSH can be seen in TABLE I.

C. Programming code Design

The programming code consist of three moduls of MapReduce functions (sub-programming code) and one modul of main programming code. The description of the programming codes is depicted in TABLE II.
The main programming code is responsible for the whole processes. First step is to obtain the document to be compared and check if the document exist, then store both documents into temporary folder. After that, main programming code execute code1 (input1.py), code2 (input2.py) and code3 (proses.py) for the final step, than the result is displayed.

Technically, there is only a small difference between creating program for stand alone computer and Hadoop System. The difference is only on statement code to execute MapReduce function. Statement code to execute MapReduce function in Hadoop System should be added by line:

```python
-r hadoop
```

![Fig. 8 Sample of Code Program in Stand Alone Computer](#)

![Fig. 9 Sample of Code Program in Hadoop System](#)
Code1 is a sub-programming code to check the first document. There are two main processes in this code. First process is parsing, get the word by word from the text and then filtering it with stopword list. The second process is to create n-gram from each word and give them value ‘1’, therefore the output will be like this:

\[ 1|0|\text{n-gram}. \]

The position of value ‘1’ is in the first place to determine that the n-gram identify first document. The value ‘1’ will be used later to count the number of the n-gram.

Code1 and code2 have the same process flow. The difference of two codes is in the output. In the code1, the position of value ‘1’ is in the first place, whereas in the code2 value ‘1’ is in the second place. The output will be like:

\[ 0|1|\text{n-gram}. \]

First step in code3 is to do a group by n-gram the combined output of code1 and code2. Therefore, if both documents have the same n-gram, the statement is written as:

\[ \text{sum1|sum2|}x|\text{n-gram}. \]

“x” is a value to indicate whether n-gram exist both in documents. If n-gram exist both in the documents the “x” value will be value of sum1 or sum2 depend on which one has the smaller value. If n-gram not exist in both documents, “x” value will be “0”. With this information, the jaccard computation can be done.

To execute the programming code type: python [main programming code name].

---

**D. Result Evaluation**

In this experiment, it was also tested the accuracy of the system by using several kind of n-grams, that is 1-gram, 2-gram, 3-gram, 4-gram, 5-gram, and 6-gram. In this experiment, it used 10 pair documents (articles) in English and 10 pair documents (articles) in Indonesian. The result are shown as follows:
From the experimental results, the pair of the documents is grouped into two groups, the first group is a pair documents that the result of similarity index tend to be increased. This group reaches the highest value mostly in 3-gram column. This indicates that using 3-gram provides the best performance. In the second group the result of similarity index tend to be decreased. As can be seen in TABLE III this group mostly the similarity index in 6-gram has smallest value and the largest value are in 1-gram column. It indicates that the pair documents are not similar and not identical.

The experiment is not only compared the accuracy of measuring documents similarity, but is also evaluate the time taken to measure various large of documents. The purpose of the experiment is to evaluate which is faster, the system that runs on stand-alone computer or on Hadoop System. TABLE IV shows the time needed to run the system in various size of documents. As can be seen in the TABLE IV and figure 14, in file size around 1 MB the system using stand alone computer is much faster than Hadoop system. Whereas in file size around 200 MB time taken using Hadoop System faster than using Stand Alone Computer. It indicates that using Hadoop system in big size is more stable and event faster than using stand alone computer (The time taken depends on computer hardware specification).

**TABLE III**

<table>
<thead>
<tr>
<th>Pair of Article</th>
<th>1-gram (%)</th>
<th>2-gram (%)</th>
<th>3-gram (%)</th>
<th>4-gram (%)</th>
<th>5-gram (%)</th>
<th>6-gram (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Increased Group</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pair 4-English</td>
<td>72</td>
<td>93</td>
<td>93</td>
<td>93</td>
<td>93</td>
<td>93</td>
</tr>
<tr>
<td>Pair 5-English</td>
<td>71</td>
<td>93</td>
<td>93</td>
<td>93</td>
<td>93</td>
<td>93</td>
</tr>
<tr>
<td>Pair 7-English</td>
<td>64</td>
<td>94</td>
<td>93</td>
<td>93</td>
<td>94</td>
<td>94</td>
</tr>
<tr>
<td>Pair 9-English</td>
<td>55</td>
<td>94</td>
<td>94</td>
<td>94</td>
<td>94</td>
<td>94</td>
</tr>
<tr>
<td>Pair 10-English</td>
<td>44</td>
<td>57</td>
<td>56</td>
<td>54</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>Pair 1-Indonesian</td>
<td>62</td>
<td>91</td>
<td>91</td>
<td>89</td>
<td>89</td>
<td>89</td>
</tr>
<tr>
<td>Pair 3-Indonesian</td>
<td>57</td>
<td>91</td>
<td>91</td>
<td>89</td>
<td>89</td>
<td>89</td>
</tr>
<tr>
<td>Pair 7-Indonesian</td>
<td>46</td>
<td>82</td>
<td>80</td>
<td>77</td>
<td>77</td>
<td>77</td>
</tr>
<tr>
<td>Pair 9-Indonesian</td>
<td>11</td>
<td>30</td>
<td>22</td>
<td>22</td>
<td>22</td>
<td>22</td>
</tr>
</tbody>
</table>

**TABLE IV**

<table>
<thead>
<tr>
<th>File Size (MB)</th>
<th>Stand Alone Computer (h:mm:ss)</th>
<th>Hadoop System (h:mm:ss)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>00:00:46</td>
<td>00:04:02</td>
</tr>
<tr>
<td>2</td>
<td>00:01:34</td>
<td>00:04:54</td>
</tr>
<tr>
<td>4</td>
<td>00:01:38</td>
<td>00:05:56</td>
</tr>
<tr>
<td>8</td>
<td>00:02:01</td>
<td>00:05:28</td>
</tr>
<tr>
<td>16</td>
<td>00:03:04</td>
<td>00:06:48</td>
</tr>
<tr>
<td>32</td>
<td>00:05:54</td>
<td>00:09:44</td>
</tr>
<tr>
<td>64</td>
<td>00:10:25</td>
<td>00:13:46</td>
</tr>
<tr>
<td>128</td>
<td>00:20:28</td>
<td>00:21:56</td>
</tr>
<tr>
<td>200</td>
<td>00:41:42</td>
<td>00:40:46</td>
</tr>
</tbody>
</table>

![Graphic of Time Taken Comparison](chart.png)

**IV. CONCLUSION**

This research proposed an application program used to provide information of the degree of documents similarity between two documents using Ferret algorithm on MapReduce and Hadoop Framework. Based on the experimental result, using 3-gram is the most accurate. System that is run on a stand alone computer is faster than system that is run on Hadoop System for document which size is less than 128MB. However, system that is run on hadoop system which document size is greater than or equal to 200MB is faster than system run on stand alone computer. Hadoop system can only be run on the Linux operating system. However, system that is run on a stand alone computer can also be run on the Windows operating system. As further studies the proposed system should be evaluated for larger collection of documents and various similarity measurements.
REFERENCES


