UTM Computing Proceedings
Innovation in Computing Technology and Applications
Volume: 2| Year: 2017 |ISBN: 978-967-0194-95-0

Map Reducing Processing On Authorship Documentation

Toh Chin Eng [1] and Siti Mariyam Shamsuddin [2]

[1] Faculty of Computing, Universiti Teknologi Malaysia (UTM), Malaysia

cetoh2@utm.my, mariyam@utm.my

Abstract. Nowadays, Big Data already is the common issue all around the world. One of the issues is about authorship recognition. Generally, the handwritten document often relates to the copyright issue because there are no proper authentication measures to identify the real owner of the handwritten document. Although there are specialists in authorship identification, but it is time – consuming to analyze many documents. Therefore, integration of parallel computing models such as Hadoop MapReduce and Apache Spark is required to solve this issue. Map Reducing processing on Authorship recognition is implemented to solve the issue of Big Data in authorship recognition. The bridging for MapReduce on pre – processing and Spark's MLlib on post – processing that contributes to the success of authorship recognition.

Keywords: big data, MapReduce, Spark, handwriting image, authorship recognition.

1 Introductıon

Contemporary, many types of parallel programming model have developed to solve different Big Data issues. MapReduce framework is one of the batch – oriented parallel programming models which is used to process Big Data on the large – scale clusters and multi – core systems which is developed by Google [2]. Besides that, it is suitable to be used for the large scale of data for standard data mining tasks on medium – size clusters [1]. Parallelization, concurrency control, resource management, fault tolerance and many other issues are handled by the map-reduce runtime [2]. Nowadays, online documents are often disputed in the copyright or plagiarisms issues because there are no authentication measures in place to identify the rightful owners of the document [3]. The task to re – analyze the writer's previous work or the writing style which also referred as stylometry to determine the authorship of the document [3]. **İf there are many** documents, the time taken to complete analyzed is tedious.

The purpose of carrying out this research aims to implement MapReduce processing and Spark's MLlib to achieve authorship recognition when processing an enormous amount of data. Hadoop is used to deal with Big Data where the input data is pre – processed by using MapReduce framework before they are used for post – processing processes for authorship recognition to achieve better performance and higher accuracy. Therefore, able to recognize multiple authors writing with the integration of MapReduce and authorship recognition, preserve the authorship of the writer and avoid eligible handwriting authorship. Research focus is in two parts which are MapReduce and authorship recognition. The handwriting data source is downloaded from the online standard database. Next, only the MapReduce framework is used for the image processing purposes while Apache Spark's MLlib is used for handwriting recognition purpose. This research project is only carried out in the Linux environment which is installed in VMWare Workstation 12 Player. The discussion on the effectiveness of MapReduce in data mining and data – analytic have been started a few years ago. Therefore, this project not only can help to protect the writer's authorship but also propose an efficient way to process multiple handwriting documents. Writer's database might be able to be built up in the future if this project has the further continuous development.

2 Literature Review

2.1 Big Data

Big data in generally can be said as the data that is too large or very complex to be processed on a single machine. According to International Data Corporation's annual Digital Universe study, the amount of data on our planet is set to be reached about 44 zettabytes (4.4×1022 bytes) by 2020 which would be ten times larger than it was in 2013 [5]. Volume, velocity, variety, veracity and variability are the five main characteristics in term of Big Data. Those factors have the direct impact on the efficiency and performance of an application on processing a large amount of input data. Distributed storage system and the algorithm that offered parallelization are the main choices when coming to Big Data issue. With the availability of data parallelism in the framework or algorithm, the data can be divided or break down into smaller pieces or chunks and then processed simultaneously [5].

2.2 Hadoop MapReduce

From generally speaking, Hadoop MapReduce job mainly consists of two user – defined functions which are a map and reduce functions. The input for MapReduce job is in a set of keys – value pairs (k, v) where the map function will be called for each of these pairs [4]. Next, the intermediate key – value pairs are grouped with the intermediate key $k1$ by MapReduce framework. Finally, the output is either zero or more aggregated results which have been defined in the reduce script. This framework has utilized a distributed file system which is called as Hadoop Distributed File System (HDFS). This distributed file system is the open source counterpart of Google File System and the I / O performance of MapReduce job is strongly depending on this file system [4].

2.3 Authorship Recognition

Authorship recognition can be defined as the task to recognize the rightful author based on the text. Authorship recognition is one of the subs – scopes of authorship identification field because authorship identification can provide details information about the author or writer. To perform authorship recognition, pre – processing and post – processing operations are required to be implemented to make sure that the whole process can be running smoothly without affecting the accuracy.

2.4 Apache Spark's MLlib (Machine Learning library)

MLlib is the machine learning library that in Apache Spark which offers the machine learning functionalities over many domains. This module offers functionality such as collaborative filtering, optimization, feature extraction, regression, statistics, frequent Pattern Mining, classification, Dimensionality Reduction and clustering [5]. In general, MLlib has the objective to make machine learning easy to be implemented and scalable. Therefore, achieved by providing common learning algorithms and utilities to perform different functionalities with data parallelization feature.

2.5 Machine learning in authorship recognition

The purpose of machine learning is to enable a system to learn from the past or present and then use the knowledge to make decisions or predictions regarding the unknown future events [5]. Back – propagation (BP) artificial neural network is one of the algorithms that is provided in Spark's MLlib which is suitable for classification purpose and already widely used for recognition of the handwritten characters [6]. The backpropagation algorithm is implemented to reduce the error until the ANN learns from the training data. The inputs must be formatted into images with the fix size pixels before they are passed into the backpropagation artificial neural network for processing. Therefore, authorship recognition can be achieved.

3 Research Methodology

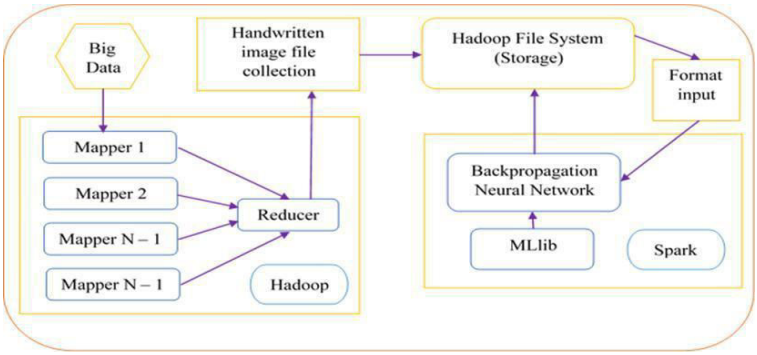

Figure 1. Overview of the overall workflow

The overall workflow is separated into five phases as shown in the above **Figure 1**. The first phase is the data collection phase where the input data is collected. The second phase is where the map reducing processing on the input data with the integration of OpenCV image processing algorithms. The third phase is where further processing on the input data with OpenCV such as extract pixel data from image files. The fourth phase is where the Spark's Mllib is used to perform authorship recognition. Lastly, the performance comparison between map reducing processing on authorship recognition and map reducing processing with map phase only on authorship recognition is carried out.

4 Results, Analysis and Discussion

4.1 Analysis on the pre – processed data

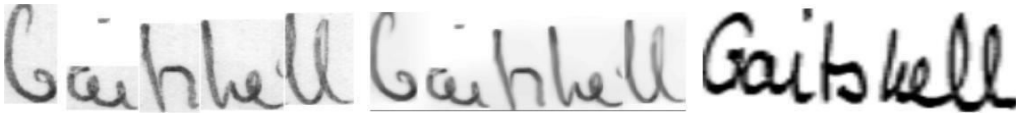

Figure 2. Denoised, rescale, binarization, conversion

In **Figure 2**, the image on the most left side is the raw data input which is downloaded from the online database while the rightmost image is the last output from pre – processing with MapReduce framework. Repeated testing of the image processing algorithms in OpenCV is carried out on the image files to select the best algorithms that produce the better output. Many algorithms can be used for the same purpose but the produced output is different. The last output is stored in HDFS to be prepared for authorship recognition phase.

4.2 Experiment with Map Reducing Processing

```
last tool output: |Execution time: 11634.0433118 seconds|
```

Figure 3. Execution time Map Reducing processing

Changes in the code are tested with the small and large dataset. The problem that is faced during large dataset is the broken pipe exception after mapping phase. This exception is where the MapReduce streaming process is terminating prematurely after mapping phase is completed. Map reducing processing on small dataset can be streamed successfully but not a large dataset for image processing purpose. After testing for numerous times, the threshold for the number of image files that are processed is discovered which is 16, 381 of image files. The streaming process failed when the image file is increased by one as input. The reducer script is not able to be executed which mean that the pre – processing process only executed 50 % only. This problem is discovered because when testing with the small dataset and it can execute perfectly fine but will fail when testing with a

large dataset which is more than 40, 000 image files. **Figure 3** is the approximate time to complete processing the whole dataset which is 115, 318 of image files. After analysis, few possibilities are figured out that might contribute to this problem to happen. Firstly, MapReduce framework is trying to pass the output from map phase through STDIN (command – line arguments) to reduce phase, but for this research project map phase does not return output in command – line arguments form. Next, MapReduce framework is not specialized on image processing but on key value pairs data. Second, Third, the timeout configuration for reducing phase. There is a timeout for the reducer to wait for map phase output by default is 600 seconds. However, the problem still exists although the timeout variable is changed few times and tested. Therefore, this assumption is eliminated after analysis on the testing results.

4.3 Experiment with Map Reducing Processing with mapping phase only

The proposed method is the combination of mapper and reducer script into mapper script only and implemented together with the multiprocessing module of Python. Firstly, the image processing functions in the reducer script are integrated into mapper script. Next, rewrites the mapper script to implement multiprocessing module. Lastly, the number of reducer task are set to 0 using the explicit command. Theoretically, the map phase output will be stored directly in the specified directory and skip the reduce phase. Therefore, the sorting, shuffling and aggregation of output from map phase will not be performed by MapReduce framework. Indeed, mentioned operations are not required for image processing purpose. Therefore, the time is taken to complete the pre – processing will be shortened which is more efficient and the streaming process can be successfully completed without error.

4.4 Experiment with Authorship Recognition

After pre – processing, the image files that are belonging to 76 directories are separated into 7 directories to perform k – fold validation during authorship recognition phase. The Python script is written to separate the image file into different directories. After that, the pixel data of all the image in training directory and testing directory are saved to two text files. Lastly, the text files are used to feed the backpropagation neural network for authorship recognition purpose where only contains pixel data. Mentioned process is repeating for 7 times where each of the directories will be in the testing folder once for validation purpose.

4.5 Comparison between Map Reducing and map reducing with map phase only

Without map reducing processing, the time taken to complete the whole process is tedious and not efficient. Distributed processing and parallel processing cannot be implemented. This is because the image files are processed in the sequence form. Therefore, the virtual machine cannot fully utilize the resources although quad cores are assigned which results in poor performance and slow. When MapReduce framework is implemented, the image files are processed in parallel form. The image processing processes are separated into the map and reduce phases. However, the threshold is found where the MapReduce framework streaming task will fail if the threshold value is passed which contribute to broken pipe exception. Therefore, only map task can be completed successfully but reduce task is failed. This is because when the output from map task is flushed in the pipe form but the pipe is broken.

Therefore, map reducing processing with map phase only is implemented. With the integration to map task only, the output is stored directly in a temporary directory without passing through reduce phase. Next, sort, shuffle and aggregate processes that are executed automatically can be skipped. Multiprocessing module in Python is implemented to achieve multi – core processing and improve the performance. Besides that, the increment in map tasks to handle the input data in smaller chunks to perform parallel processing. The result is where the time is taken to complete the whole pre – processing process is shortened when comparing to before optimizations are implemented and the issue of broken pipe exception can be resolved.

5 Conclusıons

The first objective is to recognize multiple authors writing with the integration of MapReduce and authorship recognition. Backpropagation neural network also can be called as multilayer perceptron classifier in Spark's MLlib has selected to be used to perform authorship recognition. The input which is the handwritten image files that are used to feed the backpropagation neural network must be pre - processed. The integration of MapReduce framework and OpenCV are used to pre – process the handwritten image files before passing into the backpropagation neural network to perform large scale authorship recognition. When the first objective is achieved, this solution can be used to preserve the authorship of the writer. Therefore, 3 objectives of this research project are achieved.

The recommendation for the future works is where the environment is setup on the host operating system instead of using the virtual machine. Therefore, the actual hardware can be fully utilized to improve the performance such as using the graphics card to achieve GPU acceleration purpose. Besides that, CaffeOnSpark can be implemented to boost up the performance. This is because deep learning convolutional neural network that provides by CaffeOnSpark for classification purpose has GPU acceleration feature. The performance of deep learning convolutional neural network has been proven that is better on classification purpose with higher accuracy. The next suggestion is the hardest part to be implemented in this project. This is where construct a universal filter where able to convert all the data into the data format that is accepted by Spark application without losing the information of the writer. It can collect the writer information from online source after receiving the pre – processed data. The writer data and the pre – processed data are feed to the learning algorithm for learning. This is where authorship recognition is upgraded to authorship identification. The Spark application not only able to recognize the writer's writing but at the same time can provide the writer information based on the handwriting that is given.

References

1. Wu, X., Zhu, X., Wu, G.-Q. and Ding, W. Data mining with big data. Knowledge and Data Engineering, IEEE Transactions on, 2014. 26(1): 97– 107.
2. Jiang, W., Ravi, V. T. and Agrawal, G. A map-reduce system with an alternate api for multi-core environments. 2010: 84–93.
3. Tan, R. H. R. and Tsai, F. S. Authorship identification for online text. 2010: 155–162.
4. Landset, S., Khoshgoftaar, T. M., Richter, A. N. and Hasanin, T. A survey of open source tools for machine learning with big data in the Hadoop ecosystem. Journal of Big Data, 2015. 2(1): 1–36.
5. Narayanan, A., Paskov, H., Gong, N. Z., Bethencourt, J., Stefanov, E., Shin, E. C. R. and Song, D. On the feasibility of internet-scale author identification. 2012: 300–314.
6. Zamora-**Mart´´ınez, F., Frinken, V., Espana**-Boquera, S., Castro-Bleda, M. J., Fischer, A. and Bunke, H.

   Neural network language models for off-line handwriting recognition. Pattern Recognition, 2014. 47(4): 1642–1652.