JOIV : Int. J. Inform. Visualization, 7(2) - June 2023 384-390

INTERNATIONAL JOURNAL ON INFORMATICS VISUALIZATION

journal homepage : www.joiv.org/index.php/joiv

Text Classification Using Genetic Programming with Implementation of Map Reduce and Scraping

Wirarama Wedashwara [a,*], Budi Irmawati [a], Heri Wijayanto [a], I Wayan Agus Arimbawa [b], Vandha Pradwiyasma Widartha [c]

[a] Department of Informatics Engineering, University of Mataram, Mataram, Indonesia
[b] Department of Technology Management, Economic, and Policy, Seoul National University, Seoul, Republic of Korea
[c] Department of Information System, Telkom University, Bandung, Indonesia

Corresponding author: *wirarama@unram.ac.id

***Abstract*— Classification of text documents on online media is a big data problem and requires automation. Text classification accuracy can decrease if there are many ambiguous terms between classes. Hadoop Map Reduce is a parallel processing framework for big data that has been widely used for text processing on big data. The study presented text classification using genetic programming by pre-processing text using Hadoop map-reduce and collecting data using web scraping. Genetic programming is used to perform association rule mining (ARM) before text classification to analyze big data patterns. The data used are articles from science-direct with the three keywords. This study aims to perform text classification with ARM-based data pattern analysis and data collection system through web-scraping, pre-processing using map-reduce, and text classification using genetic programming. Through web scraping, data has been collected by reducing duplicates as much as 17718. Map-reduce has tokenized and stopped-word removal with 36639 terms with 5189 unique terms and 31450 common terms. Evaluation of ARM with different amounts of multi-tree data can produce more and longer rules and better support. The multi-tree also produces more specific rules and better ARM performance than a single tree. Text classification evaluation shows that a single tree produces better accuracy (0.7042) than a decision tree (0.6892), and the lowest is a multi-tree(0.6754). The evaluation also shows that the ARM results are not in line with the classification results, where a multi-tree shows the best result (0.3904) from the decision tree (0.3588), and the lowest is a single tree (0.356).**

***Keywords*— Text classification; genetic programming; web scraping; map-reduce.**

Manuscript received 20 Mar. 2022; revised 15 Sep. 2022; accepted 12 Dec. 2022. Date of publication 30 Jun. 2023.

I. INTRODUCTION

Classification of text documents on online media is a big data problem and requires automation [1]–[3]. Text classification accuracy can decrease if many ambiguous terms exist between classes [4], [5]. Categorizing terms for large data requires parallel processing [6]. Hadoop Map Reduce is a parallel processing framework for big data that has been widely used as an OLAP (Online Analytic Processing) platform [7], [8]. Hadoop Map Reduce has also been widely used for text processing on big data [9].

The study presented text classification using genetic programming [10], [11] by pre-processing text using Hadoop map-reduce and collecting data using web scraping [12]–[14]. Genetic programming is used to perform association rule mining (ARM) before text classification to analyze big data patterns [15]–[17]. The data used are articles from science-direct with the keywords Internet of Things, Big Data, and Machine Learning.

This study aims to perform text classification with ARM-based data pattern analysis. It is hoped that data patterns between labels can be known through ARM, affecting the acquisition of accuracy. The research also aims to form a data collection system through web-scraping, pre-processing using Hadoop map-reduce, and text classification using genetic programming.

The evaluation begins with a discussion of the data that has been collected using web scraping and map-reduce to the translation of word tokenization [18]–[20]. Furthermore, a comparison is made between the single-tree and multi-tree models in genetic programming. Finally, a comparison of the accuracy results with the decision tree algorithm is carried out, which is considered to have similar properties.

Research related to text classification using tree-based algorithms has been carried out using decision trees as feature selection [21], term weighting schemes for short-text classification [5], [19], [22], [23] and text classification and clustering of Twitter data for business analytics [24]. The three studies used a decision tree, an algorithm that will be compared with genetic programming. In addition, no research combines it with pre-processing text using map-reduce.

Research related to the use of map-reduce for pre-processing has been carried out to review the algorithmic aspects of parallel processing [25], Scalable Distributed Data Processing [26]–[28], to Effective processing for unstructured data using python [29]. The proposed research uses the python programming language and parallel processing; however, it uses a different kind of pre-processing and algorithm.

Genetic programming for text processing purposes has been carried out for the automated selection and configuration of multi-label grammar-based [30], [31] and feature selection on highly dimensional skewed data [32]. Both studies did not involve web scraping and map-reduce as in this study. This study also compares single-tree and multi-tree models in performing rule extraction.

II. Materials and Method

Figure 1 shows an overview of the system. The data is collected through a web scraping process using the scrappy library in the python programming language. The web scraping process extracts specific HTML tags from the source HTML page, namely science direct. Class labels are separated based on the search keywords in the science direct search form: Internet of Things, Big Data, and Machine Learning.

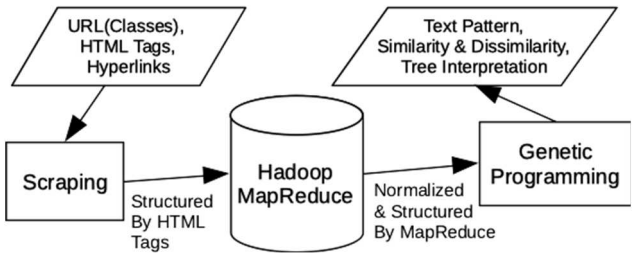

Fig. 1 General View of The System

Storage is carried out on Hadoop for the map-reduce process to be carried out. The map-reduce process allows parallel processing making it suitable for processing large amounts of data. The mapping process is carried out to separate the words in the collected articles. The stop-word removal process is also carried out in the mapping process. The tokenization process is carried out in the reduction process, namely counting the number of word occurrences. The calculation of tokenization in reducing is done by separating the occurrences of words that only occur in one label or appear in general.

Genetic programming is used to extract text patterns, calculate similarity and dissimilarity, and tree interpretation to the main research objective: text classification. Data processed by genetic programming is data that is already in the form of objects created through the map-reduce process in Hadoop.

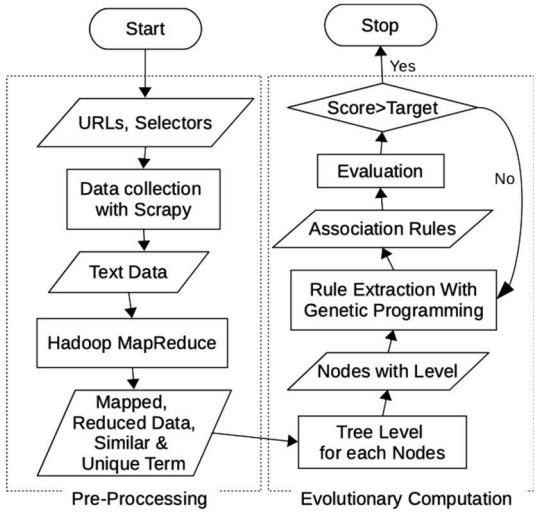

Fig. 2 The Flowchart

Figure 2 shows a flowchart of the system. The process is divided into pre-processing and evolutionary computation. The pre-processing process consists of an input URL (science direct) and tags downloaded by scrappy. Text data is processed by map-reduce until it becomes an object ready to be processed by genetic programming.

Genetic programming performs word levels based on the frequency of occurrence. Genetic programming will perform rule extraction by prioritizing words with high to low-frequency occurrences, and the extracted rules will perform the classification until it reaches the expected accuracy target.

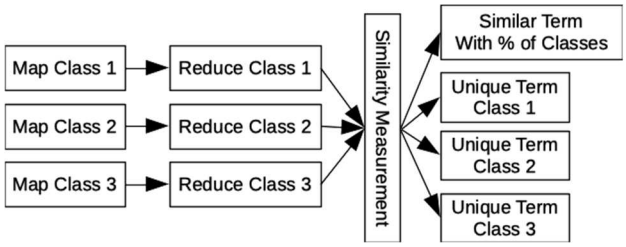

Fig. 3 Map-Reduce

Figure 3 shows the map-reduce process on the system. The mapping process is carried out based on data per class by separating the terms (words) from the downloaded articles. The reduction process is also carried out based on each class by tokenizing terms in the previous map. The final process is done by separating words that appear in their respective classes as unique and similar terms. Unique terms will be used to form specific rules, while similar terms will form common rules in genetic programming.

The structure of the rule extractor gene from the single-tree and multi-tree genetic programming models is described, which is used as a rule-based classifier in this study. The discussion includes the structure of genes in a tree graph view, the structure of objects in programming, and examples of rules generated by each rule extractor. The rules generated by the rule extractor are only partially displayed due to page limitations.

A. Single Tree Gene Structure

The image of the genetic programming gene structure with a single tree model is shown in Figure 4. In the single tree

structure, all nodes are joined in one tree with more than one root containing labels or keywords in the search. A node with a square shape contains a label. A node in a circle contains terms, namely words, and their frequency of occurrence. The tree level is divided into four parts, name labels and three levels of frequency of word occurrences represented by each node.

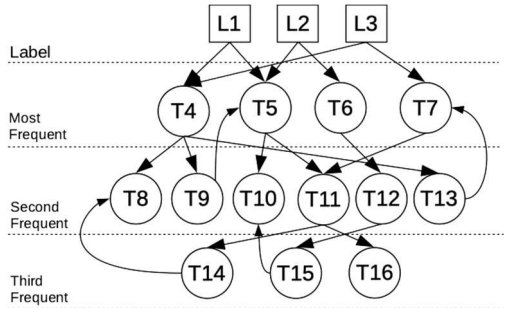

Fig. 4 Graph of The Single Tree Gene Structure

The tree structure in a single tree is a directed graph that allows multiple directions from one node to another. Direction is not always from top to bottom but allows going up to nodes at the above level. So even though a single tree allows very varied rule extraction.

TABLE I
GENE STRUCTURE OF SINGLE TREE

i	NT_i	Lv_i	C_i	T_i
1	L	0	4,5	IoT
2	L	0	5,6	BD
3	L	0	4,7	ML
4	T	1	8,9,13	Internet
5	T	1	10,11	Algorithm
6	T	1	12	Data
7	T	1	11	Iteration
8	T	2	-	Nodes
9	T	2	5	Sensor
10	T	2	-	Storage
11	T	2	14,16	Cloud
12	T	2	15	Network
13	T	2	7	Training
14	T	3	8	Information
15	T	3	10	Classification
16	T	3	-	Clustering

Table 1 shows the gene structure of genetic programming with a single-tree extractor rule in Figure 4. The first column i shows the index from nodes 1 to 16. The NTi column shows the node type of each node. Type L shows labels, namely Internet of Things (IoT), Big Data (BD), and Machine Learning (ML). At the same time, type T indicates terms, namely words that appear in articles that have been collected through the web scraping process.

Column Lvi shows the levels in the tree structure, namely 0 for root, 1 for the most frequent, 2 for the second, and 3 for third most frequent. The Ci column shows the connections between nodes. Multiple connections it is represented by an array in programming. Ti indicates the term, which is the word represented by each node.

Table 2 shows the rules extracted by a single tree. The first column's rule's structure arrows indicate the separator between precedent and dependent. The precedent is placed by a label, and the dependent indicates the frequent item set. The length shows the number of nodes in the dependent. Confidence and support show the evaluation results of association rule mining which can be seen in previous studies. The score contains a combination of length, confidence, and support, shown in formula 1.

TABLE II
EXTRACTED RULES OF SINGLE TREE

Rules	l	Conf	Supp	Score
$L1 \rightarrow T4 \wedge T13$	2	0.456	0.398	2.626
$L1 \rightarrow T4 \wedge T13 \wedge T7$	3	0.348	0.512	3.686
$L1 \rightarrow T4 \wedge T13 \wedge T7 \wedge T11$	4	0.321	0.234	4.394
$L1 \rightarrow T4 \wedge T13 \wedge T7 \wedge T11 \wedge T14$	5	0.234	0.102	5.219
$L1 \rightarrow T4 \wedge T13 \wedge T7 \wedge T11 \wedge T14 \wedge T8$	6	0.012	0.002	6.008
Average				4.386

Only the rule by L1 is shown in table 2 due to page limitations. The extracted rules can be more from L2 and L3. The extracted rule is incremental and does not always have to end at the bottom. The shorter the rule, the higher the support due to the fewer conditions. However, it becomes less robust for use in classification or regression processing. So, it is prioritized on the length of the rule to produce more specific conditions for determining labels.

B. Multi Tree Gene Structure

The multi-tree structure in genetic programming is shown in Figure 5. In contrast to the single tree between labels, there are no connected nodes. So, there are duplicate terms between trees, such as T4 contained in each tree. In contrast to the single tree structure, which allows going back to the nodes above, in one tree, there is also the same term as T10, which has three duplicates in the L2 tree. The multi-tree graph representation looks simpler but more complicated in the object structure that will be carried out next.

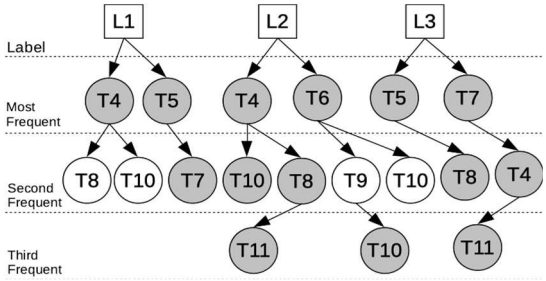

Fig. 5 Graph of The Multiple Tree Gene Structure

TABLE III
GENE STRUCTURE OF MULTI-TREE

i	NT_i	L_i	Lv_i	C_i	V_i
1	L		0	[4,5]	IoT
2	L		0	[4,6]	BD
3	L		0	[5,7]	ML
4	T	1,2,3	1,1,2	[8,10],[10,8],[11]	Internet
5	T	1,3	1	[7],[7,8]	Algorithm
6	T	2	1	[9,10]	Data
7	T	1,3	1,2	[],[14]	Iteration
8	T	1,2,3	2,2,2	[],[11],[]	Nodes
9	T	2	2	[10]	Network
10	T	1,2,2,2	2,2,2,3	[],[],[],[]	Storage
11	T	2,3	3,3	[],[]	Cloud

The structure of objects in a multi-tree is shown in table 3. Columns i, NTi, Lvi, Ci and Vi, have the same function as a single tree. Column Li has a function to show ownership by the label. The number of Li and Lvi is always the same to indicate the level of nodes in each tree. Because no directors can return to the node above it, Lvi can also be duplicated in the same tree as T10.

An additional array indicates connection Ci. The number of arrays is always equal to the sum of Li and Lvi. If there is no further connection will contain an empty array. This object structure allows node representation without duplicating array members.

TABLE IV
EXTRACTED RULES OF MULTI TREE

Rules	Length	Conf	Support	Score
$L1 \rightarrow T4 \wedge T8$	2	0.357	0.349	2.5275
$L1 \rightarrow T4 \wedge T10$	2	0.291	0.548	2.6935
$L1 \rightarrow T5 \wedge T7$	2	0.348	0.346	2.52
$L2 \rightarrow T4 \wedge T10$	2	0.267	0.647	2.7805
$L2 \rightarrow T4 \wedge T10$	2	0.479	0.178	2.4175
$L2 \rightarrow T4 \wedge T8$	2	0.678	0.789	3.128
$L2 \rightarrow T4 \wedge T8 \wedge T10$	3	0.567	0.658	3.9415
Average				2.858

Table 4 shows an example of a rule extracted by a multi-tree structure. The example shows some of the rules extracted by L1 and L2. L1 shows a maximum of two dependent combinations because there are only two levels. At the same time, L2 can reach three combinations because it has three levels. The difference between the results and a single tree will be discussed in the next sub-chapter.

III. RESULTS AND DISCUSSION

The evaluation results begin with a discussion of the data that has been collected using web scraping and map-reduce to the translation of word tokenization. Furthermore, a comparison is made between the single-tree and multi-tree models in genetic programming. Finally, a comparison of the accuracy results with the decision tree algorithm, which has similar properties, is carried out.

A. Web Scraping Result Data

Table 5 shows the data that has been collected through the web scraping process. Data is collected from the latest articles as of December 1, 2021, up to a limit of 6000 titles for each keyword IoT, Big Data (BD) and Machine Learning (ML). The horizontal header shows each keyword that was scraped and the total. Each keyword collected as many as 6000 titles and abstracts and a total of 18000.

TABLE V
DATA THAT HAS BEEN COLLECTED THROUGH WEB SCRAPING

	IoT	BD	ML	Total
Collected	6000	6000	6000	18000
Used	5923	5873	5922	17718
Duplicated	57	127	78	262
IoT		56	35	91
BD	17		43	60
ML	40	71		111

The vertical header shows a description of the number collected, used, the same total number of duplicates for each keyword, and a relationship with which the duplicates occurred. For example, the IoT label has 57 duplicates, 17 with BD and 40 with ML. BD owns most duplicates, as many as 127, namely 56 with IoT and 71 with ML. By subtracting 262 articles, the total data used is 17718. Through this duplicate collection, it can be estimated that false positives will appear between labels because of the similarities in the search results.

B. Map-reduce Results

Table 6 shows the extracted words from the articles that have been previously collected through the web scraping process. In the mapping process, each word in the article is separated, and the number of words shown by the table is specific words that have gone through the previous stop word removal process. In the mapping process, 209968 words were extracted from all keywords.

TABLE VI
RESULTS OF DATA COLLECTION USING MAP REDUCE

	IoT	BD	ML	Total
Map	66838	70327	72803	209968
Reduce	11867	12716	12056	36639
Unique	2189	1736	1264	5189
Duplicated	9678	10980	10792	31450
IoT		7897	5283	13180
BD	6689		5509	12198
ML	2989	3083		6072

The tokenizer process is carried out in the reduction process by counting the same words to get the term frequency. Next, the words appear only on each label (unique) and appear in other keywords (duplicated) or inverse document frequency. The bottom three lines show the similarity of the keywords to each other. For example, the keyword IoT has 9678 similar words, namely 6689 with BD and 2989 with ML.

The total of the same words is 31450, and only 5189 unique words will be used, which will be used to create the rule extractor tree in genetic programming. IoT has the unique words, which is 2189, followed by BD, which is 1736, and ML, which is 1264. Through this table, the tree complexity of each keyword can be analyzed.

C. Comparison of ARM Single Tree and Multi-tree Results

Table 7 shows the results of testing association rule mining from genetic programming with a single and multi-tree. The comparison of extracted rules between single and multiple trees is shown in figure 6.

TABLE VII
COMPARISON OF SINGLE TREE AND MULTI TREE

	Single Tree			Multi Tree		
Data	**rules**	**length**	**support**	**rules**	**length**	**support**
3544	127	2	0.189	203	2	0.278
7088	156	2	0.276	214	2	0.289
10631	167	2	0.307	223	3	0.349
14175	202	3	0.439	231	5	0.512
17718	234	4	0.569	245	6	0.524

The evaluation includes the number of generated rules, the average length of the resulting rules, and the average support generated. The evaluation was carried out with five different amounts of data, starting from 3544 to all data, namely 17718.

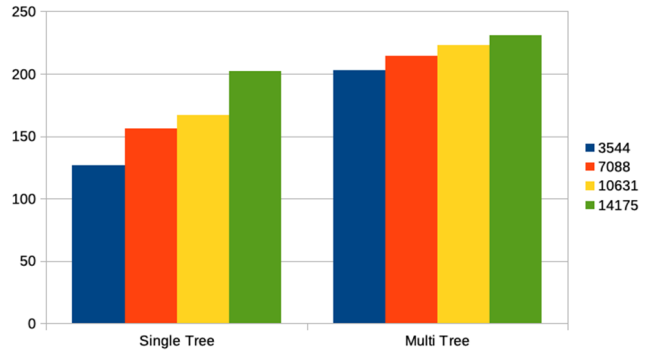

Fig. 6 Comparison of Extracted Rules between Single and multiple trees

The comparison of supports between single and multiple trees is shown in figure 7. The results show that the length of the resulting rules is getting higher in line with the increase in data, namely a maximum of four for a single tree and a higher multi-tree with a length of six. The number of supports also increases in line with the amount of data, but for support, the result for the single tree is higher at 0.569 compared to the multi-tree with a final value of 0.524.

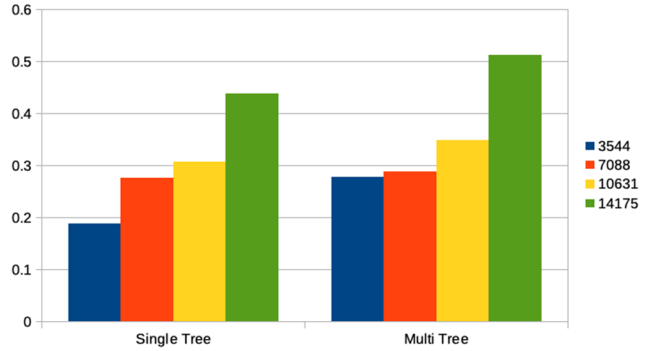

Fig.7 Comparison of Support between Single and multiple tree

The best support results are obtained in multi-tree data at 14175 with an average length of five and support of 0.512. The number of rules generated by multi-tree is greater than the initial test with little data, which is 203 degrees from a single tree with 127 rules.

TABLE VIII
DESCRIPTION OF COMMON AND SPECIFIC RULES IN SINGLE TREE

	Common Rules		Specific Rule		Total	
Data	**Rules**	**Supp**	**Rules**	**Supp**	**Rules**	**supp**
3544	83	0.112	44	0.077	127	0.189
7088	120	0.124	36	0.152	156	0.276
10631	142	0.156	25	0.151	167	0.307
14175	144	0.321	58	0.118	202	0.439
17718	156	0.324	78	0.245	234	0.569
	129	0.2074	48.2	0.1486	177.2	0.356

The description of common and specific rules is shown in table 8 for the single tree and table 9 for the multi-tree. The common rule is a rule that applies to all keywords, and a specific rule is a rule that only applies to one keyword. Previous studies have discussed this concept in applying genetic programming for classification. In previous studies, only a single tree was used, and this study has only compared single and muti trees.

TABLE IX
DESCRIPTION OF COMMON AND SPECIFIC RULES IN MULTI-TREE

	Common Rules		Specific Rule		Total	
Data	**Rules**	**Supp**	**Rules**	**Supp**	**Rules**	**supp**
3544	76	0.156	127	0.122	203	0.278
7088	112	0.114	102	0.175	214	0.289
10631	134	0.143	89	0.206	223	0.349
14175	136	0.312	95	0.2	231	0.512
17718	145	0.313	100	0.211	245	0.524
	120.6	0.2076	102.6	0.1828	223.2	0.3904

A single overall tree produces more common rules than a multi-tree. However, for the specific rule, the multi-tree produces better results with quite many differences from a single tree. The two models do not significantly differ in the number of rules generated for support, and the total rules and support results are the same as those shown in table 7.

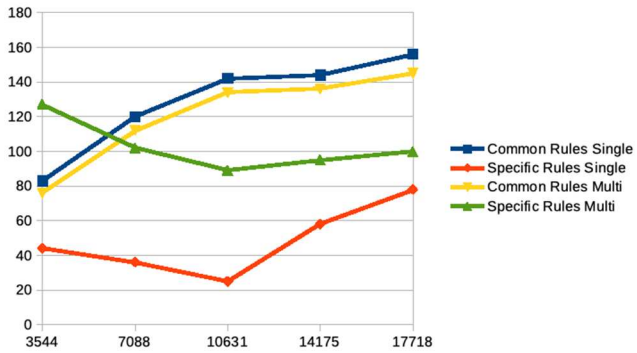

Fig. 8 Comparison of Extracted Rules between Single and multiple tree for the Common and Specific Rules

Figure 8 compares Extracted Rules between Single and multiple trees for the Common and Specific Rules. The figures show that the common rules are extracted more than specific rules both for single and multi-tree gene structures. The single tree has more extracted common rules than the multi-tree. But the multi-tree has a higher number of extracted specific rules than the single-tree.

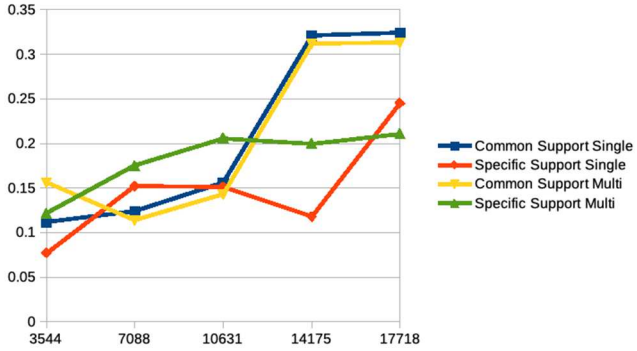

Fig. 9 Comparison of support between Single and multiple tree for the Common and Specific Rules

Figure 9 shows the comparison of support between Single and multiple trees for the Common and Specific Rules. The figures show that the common rules have higher supports than specific rule both for single and multi-tree gene structure. The single tree has higher number of supports for the common rules than the multi-tree, and however, the multi-tree supports the specific rule more than the single tree.

D. Comparison of Accuracy of Text Classification with Decision Tree

Table 10 compares text classification accuracy between genetic programming and decision trees. The evaluation includes the support of the generated rules, and the accuracy of the text classification results. The evaluation was carried out with data ranging from 3544 to all data, namely 17716. The test data was carried out using 3000 data scraped separately with 1000 data per keyword.

TABLE X
COMPARISON OF ACCURACY WITH DECISION TREE

	GP Single Tree		GP Multi Tree		Decision Tree	
Data	**support**	**Acc**	**supp**	**Acc**	**supp**	**Acc**
3544	0.189	0.598	0.278	0.546	0.234	0.679
7088	0.276	0.678	0.289	0.637	0.323	0.658
10631	0.307	0.708	0.349	0.649	0.324	0.618
14175	0.439	0.739	0.512	0.759	0.445	0.779
17718	0.569	0.798	0.524	0.786	0.468	0.712
	0.356	0.7042	0.3904	0.6754	0.3588	0.6892

For the least data, which is 3544, the decision tree has the highest accuracy, 0.679. These results show that the decision tree has better accuracy with less training data. A single tree has higher accuracy than a decision tree since the number of data is 7088. The multi-tree only produces better accuracy than the decision tree with total data of 10631 and 17718 only. On average, genetic programming with a single tree produces the highest accuracy, 0.7042, followed by a decision tree with 0.6892 and the smallest by the multi-tree with 0.6754.

For the acquisition of genetic programming support values with a single tree, the highest average support is 0.3904, followed by the decision tree with 0.3588 and the smallest single tree with 0.356. The multi-tree has the highest support results in all the training data. In comparison, the single tree has the lowest support for data from 3544 to 14175. In general, the number of supports is not in line with the accuracy value achieved.

IV. CONCLUSION

Research has developed a text classification system with pre-processing using map-reduce and web scraping data collection. Through web scraping, data has been collected by reducing duplicates as much as 17718. Map-reduce has tokenized and stopped-word removal with 36639 terms with 5189 unique terms and 31450 common terms. Evaluation of ARM with different amounts of multi-tree data can produce more and longer rules and better support. The multi-tree also produces more specific rules and better ARM performance than a single tree. Text classification evaluation shows that a single tree produces better accuracy (0.7042) than a decision tree (0.6892), and the lowest is a multi-tree (0.6754). The evaluation also shows that the ARM results are not in line with the classification results where multi-tree shows the best result (0.3904) from the decision tree (0.3588) and the lowest is single tree (0.356). Future research will be tested with different data topics, and hardware performance analysis will be carried out in data processing.

REFERENCES

[1] I. Pintye, E. Kail, P. Kacsuk, and R. Lovas, "Big data and machine learning framework for clouds and its usage for text classification," Concurr Comput, vol. 33, no. 19, p. e6164, 2021.

[2] M. Abdel-Basset, M. Mohamed, F. Smarandache, and V. Chang, "Neutrosophic association rule mining algorithm for big data analysis," *Symmetry (Basel)*, vol. 10, no. 4, p. 106, 2018.

[3] H. U. Rahman, R. U. Khan, and A. Ali, "Programming and Pre-Processing Systems for Big Data Storage and Visualization," in *Handbook of Research on Big Data Storage and Visualization Techniques*, IGI Global, 2018, pp. 228–253.

[4] B. Alt\inel and M. C. Ganiz, "Semantic text classification: A survey of past and recent advances," *Inf Process Manag*, vol. 54, no. 6, pp. 1129–1153, 2018.

[5] I. Alsmadi and G. K. Hoon, "Term weighting scheme for short-text classification: Twitter corpuses," *Neural Comput Appl*, vol. 31, no. 8, pp. 3819–3831, 2019.

[6] S. Du and J. Li, "Parallel processing of improved KNN text classification algorithm based on Hadoop," in *2019 7th International Conference on Information, Communication and Networks (ICICN)*, 2019, pp. 167–170.

[7] H. Jeong and K. J. Cha, "An efficient mapreduce-based parallel processing framework for user-based collaborative filtering," *Symmetry (Basel)*, vol. 11, no. 6, p. 748, 2019.

[8] H.-N. Dai, H. Wang, G. Xu, J. Wan, and M. Imran, "Big data analytics for manufacturing internet of things: opportunities, challenges and enabling technologies," *Enterp Inf Syst*, vol. 14, no. 9–10, pp. 1279–1303, 2020.

[9] K. v Ranjitha, B. S. V. Prasad, and others, "Optimization Scheme for Text Classification Using Machine Learning Na{\"\i}ve Bayes Classifier," in *ICDSMLA 2019*, Springer, 2020, pp. 576–586.

[10] A. Tahmassebi and A. H. Gandomi, "Genetic programming based on error decomposition: A big data approach," in *Genetic programming theory and practice XV*, Springer, 2018, pp. 135–147.

[11] T. Haryanto, A. Pratama, H. Suhartanto, A. Murni, K. Kusmardi, and J. Pidanic, "Multipatch-GLCM for texture feature extraction on classification of the colon histopathology images using deep neural network with GPU acceleration," *Journal of Computer Science*, vol. 16, no. 3, pp. 280–294, 2020.

[12] D. M. Thomas and S. Mathur, "Data analysis by web scraping using python," in *2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA)*, 2019, pp. 450–454.

[13] V. Krotov, L. Johnson, and L. Silva, "Tutorial: Legality and ethics of web scraping," *Communications of the Association for Information Systems*, vol. 47, no. 1, 2020, doi: 10.17705/1CAIS.04724.

[14] M. Dogucu and M. Çetinkaya-Rundel, "Web Scraping in the Statistics and Data Science Curriculum: Challenges and Opportunities," *Journal of Statistics Education*, 2020, doi: 10.1080/10691898.2020.1787116.

[15] A. Telikani, A. H. Gandomi, and A. Shahbahrami, "A survey of evolutionary computation for association rule mining," *Inf Sci (N Y)*, vol. 524, pp. 318–352, 2020.

[16] C. Gakii and R. Rimiru, "Identification of cancer related genes using feature selection and association rule mining," *Inform Med Unlocked*, vol. 24, 2021, doi: 10.1016/j.imu.2021.100595.

[17] W. Thurachon and W. Kreesuradej, "Incremental Association Rule Mining with a Fast Incremental Updating Frequent Pattern Growth Algorithm," *IEEE Access*, vol. 9, 2021, doi: 10.1109/ACCESS.2021.3071777.

[18] J. Ramsingh and V. Bhuvaneswari, "An efficient Map Reduce-Based Hybrid NBC-TFIDF algorithm to mine the public sentiment on diabetes mellitus--A big data approach," *Journal of King Saud University-Computer and Information Sciences*, 2018.

[19] A. K. Ngo Ho and F. Yvon, "Optimizing Word Alignments with Better Subword Tokenization," *Proceedings of Machine Translation Summit XVIII: Research Track*, 2021.

[20] K. Sirts and K. Peekman, "Evaluating sentence segmentation and word tokenization systems on estonian web texts," in *Frontiers in Artificial*

Intelligence and Applications, 2020, vol. 328. doi: 10.3233/faia200620.

[21] X. Deng, Y. Li, J. Weng, and J. Zhang, "Feature selection for text classification: A review.," *Multimed Tools Appl*, vol. 78, no. 3, 2019.

[22] T. Ma, R. Al-Sabri, L. Zhang, B. Marah, and N. Al-Nabhan, "The Impact of Weighting Schemes and Stemming Process on Topic Modeling of Arabic Long and Short Texts," *ACM Transactions on Asian and Low-Resource Language Information Processing*, vol. 19, no. 6, 2020, doi: 10.1145/3405843.

[23] S. S. Samant, N. L. Bhanu Murthy, and A. Malapati, "Improving Term Weighting Schemes for Short Text Classification in Vector Space Model," *IEEE Access*, vol. 7, 2019, doi: 10.1109/ACCESS.2019.2953918.

[24] A. S. Halibas, A. S. Shaffi, and M. A. K. V. Mohamed, "Application of text classification and clustering of Twitter data for business analytics," in *2018 Majan international conference (MIC)*, 2018, pp. 1–7.

[25] P. Koutris, S. Salihoglu, D. Suciu, and others, "Algorithmic aspects of parallel data processing," *Foundations and Trends® in Databases*, vol. 8, no. 4, pp. 239–370, 2018.

[26] B. Anjum, "MapReduce--The Scalable Distributed Data Processing Solution," in *Topics in Parallel and Distributed Computing*, Springer, 2018, pp. 173–190.

[27] S. Oliviandi, A. B. Osmond, and R. Latuconsina, "Implementasi Apache Spark Pada Big Data Berbasis Hadoop Distributed File System," *e-Proceeding of Engineering*, vol. 5, no. 1 Maret, 2018.

[28] N. D. Sapoetra, R. Ridwan, M. A. K. Sahide, and K. Masuda, "Local community's perception, attitude, and participation towards different level management of geopark: A comparison Geosite case study, between Muroto Cape and Rammang-rammang Geosite," in *IOP Conference Series: Earth and Environmental Science*, 2019, vol. 343, no. 1. doi: 10.1088/1755-1315/343/1/012044.

[29] K. Kousalya and S. J. Parvez, "Effective processing of unstructured data using python in Hadoop map reduce," *International Journal of Engineering \& Technology*, vol. 7, no. 2.21, pp. 417–419, 2018.

[30] A. G. C. de Sá, A. A. Freitas, and G. L. Pappa, "Automated selection and configuration of multi-label classification algorithms with grammar-based genetic programming," in *International Conference on Parallel Problem Solving from Nature*, 2018, pp. 308–320.

[31] L. W. Santoso, B. Singh, S. S. Rajest, R. Regin, and K. H. Kadhim, "A Genetic Programming Approach to Binary Classification Problem," *EAI Endorsed Transactions on Energy Web*, vol. 8, no. 31, 2021, doi: 10.4108/eai.13-7-2018.165523.

[32] F. Viegas *et al.*, "A genetic programming approach for feature selection in highly dimensional skewed data," *Neurocomputing*, vol. 273, pp. 554–569, 2018.