Hindawi
Scientific Programming
Volume 2021, Article ID 6645345, 10 pages
https://doi.org/10.1155/2021/6645345

Research Article

A New Big Data Feature Selection Approach for Text Classification

Houda Amazal and Mohamed Kissi

Computer Science Laboratory, Faculty of Sciences and Technologies, University Hassan II Casablanca, Mohammedia, Morocco

Correspondence should be addressed to Houda Amazal; houda.kamouss@gmail.com

Received 27 December 2020; Revised 16 March 2021; Accepted 4 April 2021; Published 19 April 2021

Academic Editor: Shaukat Ali

Feature selection (FS) is a fundamental task for text classification problems. Text feature selection aims to represent documents using the most relevant features. This process can reduce the size of datasets and improve the performance of the machine learning algorithms. Many researchers have focused on elaborating efficient FS techniques. However, most of the proposed approaches are evaluated for small datasets and validated using single machines. As textual data dimensionality becomes higher, traditional FS methods must be improved and parallelized to handle textual big data. This paper proposes a distributed approach for feature selection based on mutual information (MI) method, which is widely applied in pattern recognition and machine learning. A drawback of MI is that it ignores the frequency of the terms during the selection of features. The proposal introduces a distributed FS method, namely, Maximum Term Frequency-Mutual Information (MTF-MI), based on term frequency and mutual information techniques to improve the quality of the selected features. The proposed approach is implemented on Hadoop using the MapReduce programming model. The effectiveness of MTF-MI is demonstrated through several text classification experiments using the multinomial Naïve Bayes classifier on three datasets. Through a series of tests, the results reveal that the proposed MTF-MI method improves the classification results compared with four state-of-the-art methods in terms of macro-F1 and micro-F1 measures.

1. Introduction

Feature selection (FS) plays a key role in data mining [1], especially in text classification task that suffers from large dimensionality [2] in many application domains such as sentiment analysis [3], emotion identification [4, 5], and spam filtering [6]. Feature selection aims to select relevant and informative features (words) from large datasets [7]. Therefore, FS can reduce space dimensionality, decrease the running time in the classification process, and improve the efficiency of machine learning algorithms [8]. For this aim, FS is considered as a critical technique because it directly affects the accuracy of classification.

The FS methods can be divided into two major categories, namely, filter and wrapper methods [9]. Filter approach methods perform a statistical analysis of the feature space to select a distinguishing subset of features. Wrapper methods employ a search strategy to determine the goodness of a feature subset by providing it to the classifier and evaluating the performance. These two steps are repeated until reaching a suitable quality feature subset for a specific classifier. Wrapper methods primarily achieve better classification results than filter methods; however, they have a very high computational complexity [10] and are only efficient when the number of features is relatively small [11]. In contrast, the filter methods are efficient, scalable, and independent of any classifier interaction during the construction of the feature set. The need for classifier interaction may increase the execution time and make the FS method valuable only to a specific learning algorithm. Thus, filter methods are more suitable for large datasets [12].

Moreover, although most available FS methods for text classification are filter-based, these methods do not work when the datasets are large because they are based on the serial programming model. More precisely, classical FS algorithms need to read data into memory for analysis, but a limited memory cannot deal with the storage and processing of large datasets. Thus, FS methods are needed for

distributed environments, such as Hadoop, a powerful tool for distributed storage and distributed processing of large datasets [13]. Figure 1 presents a general overview of the distributed process of the filter FS approach for text classification.

Motivated by the above challenges, we introduce a parallel filter-based FS method for textual big data implemented on Hadoop. To this end, the proposed method focuses on the reduction of features using the term frequency (TF) [1] and mutual information (MI) techniques [14]. The MI technique is one of the most used filter FS techniques. However, the drawback of MI is that it chooses terms with high document frequency (DF) and low TF for features, which amplifies the importance of the low-frequency terms. Therefore, terms with low DF and high TF are not selected, which decreases the classifier performance because these terms are discriminative in classification.

(1) Documents are labeled and loaded into the Hadoop framework.

(2) An algorithm is introduced to calculate the TF values of features. Then, the average and maximum values of the TF for each feature are estimated based on the category under the Hadoop framework.

(3) An algorithm is proposed to calculate the MI value to evaluate the relationship between features and categories under the Hadoop framework.

In this paper, we present a hybrid distributed FS approach using the MapReduce paradigm to improve classification of textual big data. The proposal aims to select features with both high frequency and high feature-category dependency. Besides its independence from the classifiers, the proposed method is scalable and efficient for textual big data. The performance of the proposed method was compared with several state-of-the-art methods using three datasets, 20-Newsgroups, Reuters-21578, and WebKB, using multinomial NB as a classifier. According to the reported results, we can show that the proposal is outperforming standard methods.

The remainder of this paper is structured as follows. Section 2 introduces a brief literature review highlighting related work. In Section 3, the technical background in this work is discussed. The proposed method is explained in Section 4. Section 5 describes the experimental results, including the datasets, classifier, and performance measures used in the experiments. Finally, Section 6 presents the conclusion and future work.

2. Related Work

This work is focused on MI and parallel FS methods. Therefore, in the following context, we briefly present some related works on these two aspects.

Hadoop is the most used open-source MapReduce software to handle big data [15]. In [16], the authors presented a parallel FS method using MapReduce for text classification. Moreover, MI based on Renyi entropy was used to measure the correlation between features and classes.

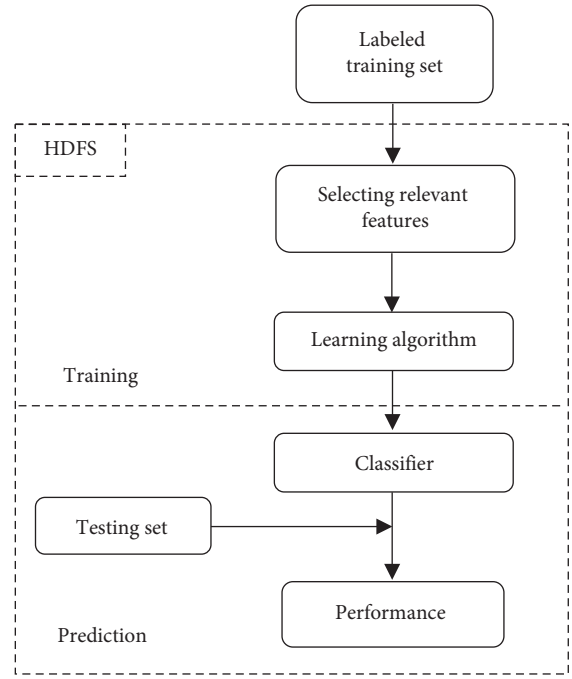

Figure 1: Feature selection process in HDFS.

Then, the maximum MI theory was used to generate the most distinguishing feature subset. In [17], the authors investigated the design and scalability of an MI-based algorithm, which is the minimum redundancy maximum relevance algorithm in MapReduce, and examined its performance in dense and sparse data.

In [18], the authors proposed a high-dimensional FS algorithm based on a variance study. The algorithm selects features by estimating their capacities to justify data variance. In [19], the authors explored a parallel FS method based on MI. However, the mentioned method is only applied to process discrete variables. In [20], the authors implemented a set of FS techniques based on a statistical test. All methods were parallelized using MapReduce on the Hadoop platform, and each feature was estimated independently. In [21], the authors introduced a MapReduce approach to derive a subset of features from large datasets. The proposed method was evaluated using classifiers, such as Support Vector Machine, Naïve Bayes, and Logistic Regression. The measurements revealed that the spark implemented framework was useful to perform evolutionary FS on massive datasets with improved classification precision and execution time.

In [22], the authors proposed a parallel FS algorithm, namely, the parallel forward-backward with pruning algorithm, for large datasets. The experimental study established increased scalability with running time. In [23], the authors proposed using MI to reduce dimensionality and improve accuracy for online streams. The proposed study focused on presenting a methodology to address the computational cost, the stability of the generated results, and the size of the final subset of selected features. In [24], the authors introduced a hybrid FS algorithm for a gene dataset by combining the MI maximization and adaptive genetic algorithm

(MIMAGA) to improve the competence of the MIMAGA algorithm. In [25], the authors proposed an evaluation of the MI-based FS methods.

In [26], the authors considered MI-based FS to increase the searching ability of the relevant subset of features. Based on MI, many studies have recently focused on maximizing the relevance of variables while minimizing variable redundancy to improve the quality of the selected features and reduce the space dimensionality [27–29].

In most of the works on FS, researchers have worked on binary classification rather than textual datasets. Selecting the most relevant features from a large volume of data has become the most significant challenge in many applications, especially in text classification [30]. As the amount of the data continues to grow, conventional algorithms cannot adapt in terms of memory requirements, execution time, and efficiency of the results. Thus, to address these large-dimensional problems, this work proposes selecting characteristics for text classification using the multicluster environment of Hadoop.

3. Technical Background

This section presents some basic concepts associated with the proposed FS approach, MTF-MI, and the parallelization technology used in our implementation (MapReduce).

3.1. Representation Phase. In this section, we denote $C = \{c_1, c_2, \ldots, c_k\}$ as the set of categories. Broadly, the documents from dataset are represented using word vectors. This representation is generated by the vector space model that uses the bag-of-words approach [31]. Thus, a text document of a category c_k is represented by a vector of features in this document. The jth document is denoted by vector $T_j = \{t_{1j}, t_{2j}, \ldots, t_{mj}\}$, where m is the number of terms in document d_j.

3.2. Mutual Information (MI). The MI is an essential concept in information theory. It is used to measure the degree of correlation between two random events [32]. In FS, MI is often used to represent the relationship between a feature and category. The MI between a feature t_i and a category c_k is defined as follows:

$$\mathrm{MI}(t_i, c_k) = \log \frac{p(t_i, c_k)}{p(t_i) \times p(c_k)}. \quad (1)$$

The approximate formula is the following:

$$\mathrm{MI}(t_i, c_k) = \log \frac{A \times N}{(A + C) \times (A + B)}, \quad (2)$$

where A is the number of documents in c_k containing t_i, B is the number of documents not in c_k containing t_i, C is the number of documents in c_k not containing t_i, and N is the total number of training documents.

Because MI does not consider the frequency of features in a text document, if two features appear in a document, their MI value is the same regardless of how often they occur. Thus, it is also necessary to consider the feature frequency in each document of the training dataset.

3.3. Hadoop Parallel Distributed Architecture. Faced with the continuous growth of data, traditional data analysis systems cannot store and process such a large volume of data. Thus, the best solution to manage the abundant data is to store it in the Hadoop distributed file system (HDFS). Due to its fault tolerance mechanism, the HDFS allows Hadoop to operate reliably and very efficiently. The HDFS can be viewed just like a regular file system; the only difference is that it handles larger datasets. This system splits data into 64 MB blocks by default, making it more efficient for large datasets. The data in HDFS are stored in two forms: the actual data and its metadata, such as file location and file size. Application data are stored in the data nodes of the HDFS, and the metadata are stored in the name node. The architecture of the parallel HDFS is illustrated in Figure 2.

The HDFS is the storage unit of Hadoop, and it follows the master-slave architecture. The master node includes three elements: the job tracker, name node, and secondary name node, whereas the slave node includes the task tracker and data node. The name node in the parallel HDFS architecture interacts with different data nodes residing in the slave nodes, whereas the job tracker in the master node organizes the task trackers on the slave nodes.

3.4. MapReduce. MapReduce is a programming model used in a distributed and parallel environment for processing large datasets [33]. The data processing in MapReduce is based on input data distribution; several tasks across many nodes execute the distributed data. A MapReduce program is divided into two main phases, map and reduce, and is executed in three steps: map, shuffle, and reduce. Figure 3 depicts the architecture of MapReduce. In the map step, input data are partitioned among nodes, and each partition of data is given as an input to a job that performs the map function. Each job processes the data and outputs key-value pairs. In the shuffle step, key-value pairs are grouped by key, and each group is sent to the reducer. The map and reduce functions are defined depending on the purpose of the application. The input and output of these functions are based on the key-value scheme. Thus, the MapReduce model allows the user to focus on the application without concern about issues, such as the program execution process on the distributed nodes, memory management, and fault tolerance. Apache Hadoop is a widely used open-source implementation of the MapReduce model.

4. Proposed Method

To deal with the problems described above, we introduced an improved MI FS approach called MTF-MI. This method introduces TF and term distribution to the classical MI method. The entire process of the proposed approach is described in Figure 4.

After the preprocessing step, including removing stop words, tokenization, and stemming, the documents are

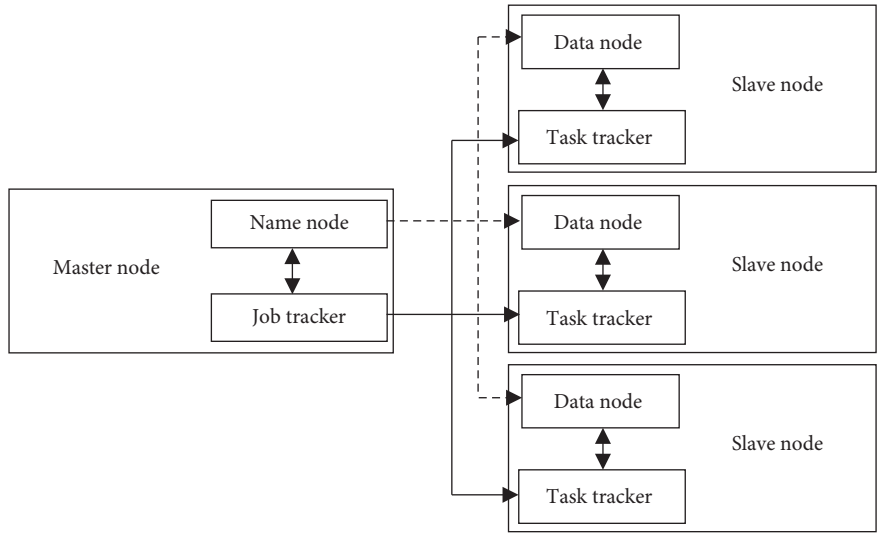

Figure 2: Hadoop distributed file system.

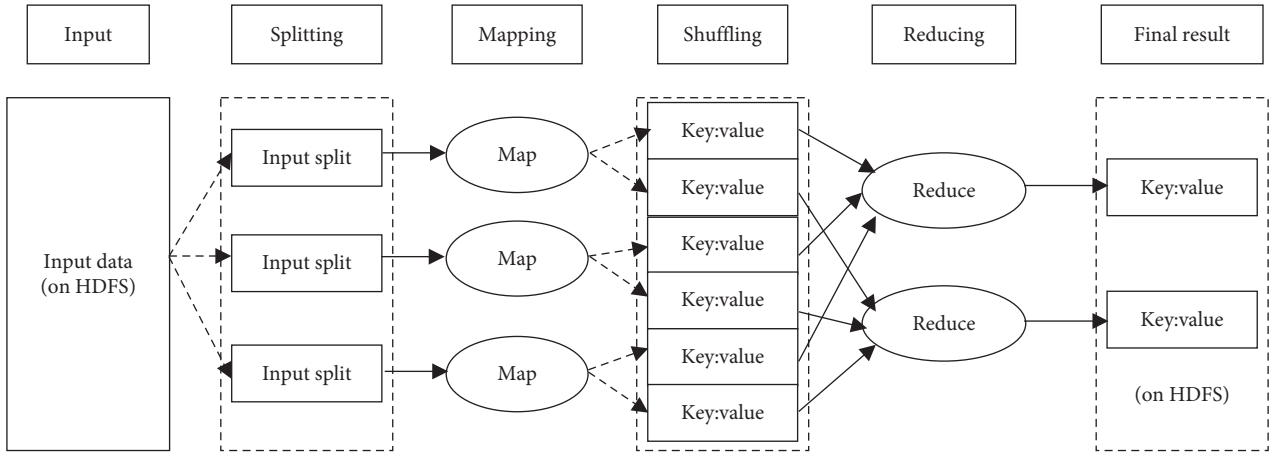

Figure 3: Phases of MapReduce.

loaded into the HDFS and are represented as described in Section 3. To redress the drawback of the traditional MI method, the TF is introduced first. tf_{ij} represents the TF of a term in a document d_j. Hence, the average term frequency $\overline{tf_i}$ and the maximum term frequency $tf_{i\max}$ for a specific category c_k can be calculated as follows:

$$\overline{tf_i} = \frac{1}{N_k} \sum_{j=1}^{N_k} tf_{ij},$$
$$tf_{i\max} = \max_{j=1}^{N_k} \{tf_{ij}\}, \tag{3}$$

where N_k is the number of documents belonging to category c_k. As the MI method is based on DF, according to its classical formula, if a term occurs many times in a document of a particular category when this type of document is rare, this term is not considered discriminative. Therefore, in this work, the TF is introduced in the MI formula. The term distribution is used to select more discriminative features. For a particular category, a feature has more discriminating power if it is regularly distributed. For this, the sample variance is used to evaluate the difference in term distributions. Sample variance is a commonly used statistics metric that measures the dispersion degree of a dataset. The sample variance is given as follows:

$$v(t_i, c_k) = \frac{1}{N_k - 1} \sum_{j=1}^{N_k} \left(tf_{ij} - \overline{tf_i}\right)^2 + \alpha. \tag{4}$$

The variable α denotes a very small real number. Finally, we introduce our approach based on the TF and term distribution to evaluate the feature t_i in category c_k as follows:

$$\text{MTF} - \text{MI}(t_i, c_k) = \frac{tf_{i\max} \times \text{MI}(t_i, c_k)}{v(t_i, c_k)}. \tag{5}$$

In the proposed method, to select terms with high discriminability power, as the TF is high and the DF is relatively low, we use the maximum TF $tf_{i\max}$, instead of the average. Based on the basic theory of MI, the greater the

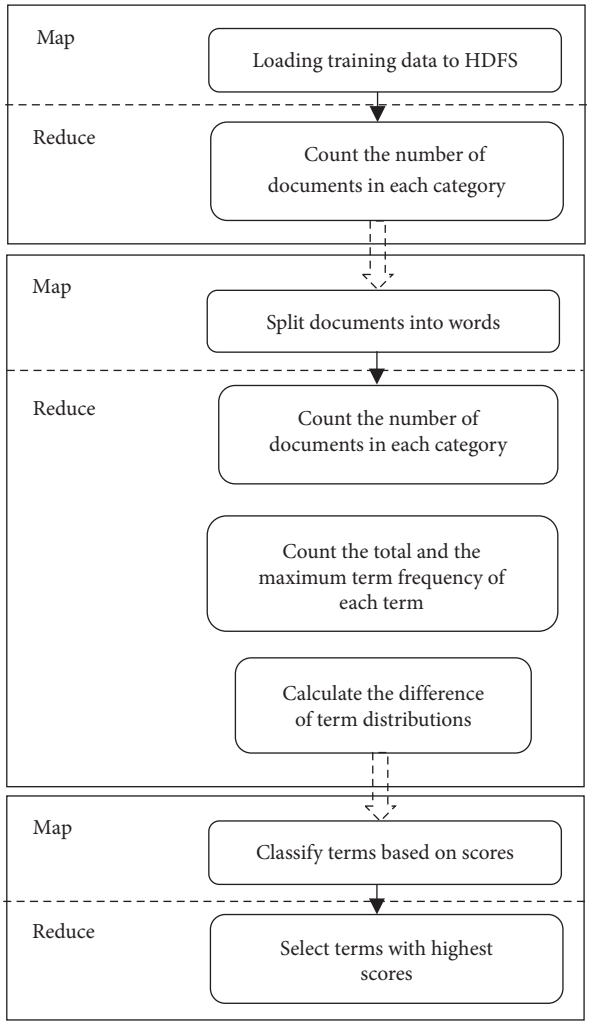

Figure 4: Proposed system.

value of MI is, the more category information the feature has. Hence, the formula is defined as follows:

$$\text{MTF} - \text{MI}_{\max}(t_i) = \max_{k=1}^{M}\{\text{MTF} - \text{MI}(t_i, c_k)\}, \quad (6)$$

where M is the total number of categories in the dataset.

Finally, the proposed method is implemented using MapReduce. The parallel implementation of the overall MTF-MI includes three process stages: job 1, job 2, and job 3.

Job 1 is achieved using Algorithm 1. Job 1 reads the incoming batch of training data and calculates the number of documents in each category. The results are used in job 2, which is achieved using Algorithm 2. For each term t_i belonging to category c_k, the total and maximum TF of t_i are calculated and stored in sum and $tf_{i\max}$, respectively. Then, using the value of sum, the average tf_i is calculated. Next, the difference in the term distributions is calculated for each term t_i in category c_k. Finally, the proposed approach calculates the value of term t_i in category c_k. Job 3 is achieved using Algorithm 3. Job 3 takes the values emitted by job 2 and assigns each term t_i to the category with the maximum score. Then, all features are sorted in descending order, and the x terms whose values are maximal are selected as the relevant features.

5. Experiments

The multinomial NB classifier [34] is used on three different datasets with different characteristics to validate the performance of the proposed MTF-MI. Two different commonly used measures, micro-F1 and macro-F1, were applied to observe the effectiveness of the MTF-MI method. The datasets and evaluation measures are briefly described in the following sections, and the experimental results are also presented.

5.1. Datasets. We achieved experiments with the Reuters-21578 [35], 20-Newsgroups [36], and WebKB [37] datasets. The Reuters-21578 dataset contains the news that appeared on the Reuters newswire in 1987 and belong to one out of eight possible categories. The 20-Newsgroups dataset contains around 20,000 documents that are taken from the Usenet newsgroup collection, and all documents were assigned uniformly to 20 different categories. The WebKB dataset is a subset of web documents, which contains 877 webpages from the computer science departments of four universities.

5.2. Naïve Bayes Classifier. The Naïve Bayes (NB) classifier is a simple probabilistic algorithm based on the Bayes theorem with a strong independence assumption [38]. The NB model is based on simplifying conditional independence assumption, which consists of, given a category, the words which are conditionally independent of each other. This assumption does not affect the accuracy of text classification and makes fast classification algorithms applicable to the problem. For this, NB is widely used in classification problems in real-world applications.

5.3. Performance Measures. In this study, two commonly used measures are employed, which are the macro-F1 and the micro-F1 [39]. In macro-F1, F-measure is calculated for each category within the dataset and then the average over all classes is obtained. Hence, the same weight is assigned to each category without regarding the class frequency. Macro-F1 can be formulated as follows:

$$\text{Macro} - F1 = \frac{\sum_{k=1}^{c} F_k}{C}, \qquad F_k = \frac{2 \times p_k \times r_k}{p_k + r_k}, \quad (7)$$

where couple of (p_k, r_k) corresponds to precision and recall values of class k, respectively.

However, in micro-F1, the F-measure is computed globally without class discrimination. In this way, all classification decisions in the whole dataset are considered. If the classes in a dataset are biased, large classes dominate small

(i) Map
(ii) Input:
(iii) key: document name
(iv) value: document content
(v) Emit (ck, dj)
(vi) EndMap
(vii) Reduce
(viii) Input:
(ix) key: ck
(x) values: list $[dj]$
(xi) $N_k \longleftarrow 0$ //total number of documents in the category c_k
(xii) for each value in values do
(xiii) $\llcorner N_k + +$
(xiv) Emit (c_k, N_k)
(xv) EndReduce.

Algorithm 1: Job 1.

(i) Map
(ii) Input:
(iii) key: Offset
(iv) value: line of document
(v) Emit $((t_i, c_k), dj)$
(vi) EndMap
(vii) Reduce
(viii) Input:
(ix) key: (t_i, c_k)
(x) values: list $[d_j]$
(xi) for each value in values do
(xii) $\llcorner \text{sum}(t_i) + = tf_{ij}$
(xiii) $\overline{tf_{i\max}} = \max\left\{\overline{tf_{i\max}}, tf_{ij}\right\}$
(xiv) $\overline{tf_i} = \text{sum}(t_i)/N_k$ for each value in values do
(xv) $v(t_i, c_k) = (tf_{ij} - \overline{tf_i})^2/(N_k - 1) + \alpha$
(xvi) MTF-MI $(t_i, c_k) = tf_{i\max} \times \text{MI}(t_i, c_k)/v(t_i, c_k)$
(xvii) emit $((t_i, c_k), \text{MTF} - \text{MI}(t_i, c_k))$
(xviii) EndReduce.

Algorithm 2: Job 2.

(i) Map
(ii) Input:
(iii) key: (t_i, c_k)
(iv) value: MTF-MI (t_i, c_k)
(v) emit $(t_i, (c_k, \text{MTF} - \text{MI}(t_i, c_k)))$
(vi) EndMap
(vii) Reduce
(viii) Input:
(ix) key: t_i
(x) values: list $[(c_k, \text{MTF} - \text{MI}(t_i, c_k))]$
(xi) MTF $-$ MI $(t_i) = \max \text{MTF} - \text{MI}(t_i, c_k)$
(xii) Emit $(t_i, \text{MTF} - \text{MI}(t_i))$
(xiii) EndReduce.

Algorithm 3: Job 3.

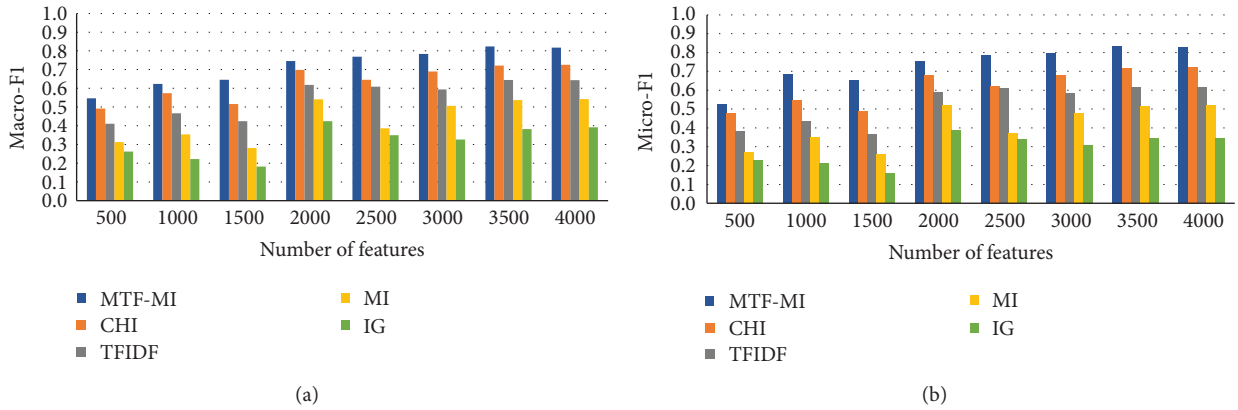

Figure 5: Macro-F1 and micro-F1 measures of multiclass text classification on the Reuters-21578 corpus with different numbers of features.

ones in microaveraging. Micro-F1 can be formulated as follows:

$$\text{Micro} - F1 = \frac{2 \times p \times r}{p + r}, \tag{8}$$

where pair (p, r) represents the precision and recall values, respectively, over all the classification decisions within the whole dataset, rather than the individual classes.

5.4. Results and Discussion. The macro-F1 and micro-F1 performances of MTF-MI are compared to four widely used feature selection techniques using Naïve Bayes classifier applied on three datasets (20 Newsgroups, Reuters-21578, and WebKB). The four feature selection techniques used for comparison are the classical MI, Chi-square (CHI), Term Frequency-Inverse Document Frequency (TF-IDF), and Information Gain (IG).

Figures 5–7 show the classification performance of the different feature selection methods for the three datasets. In all figures, the horizontal and vertical axes present the number of selected features and the corresponding classification performance, respectively.

Figure 5 presents the F1 classification performance based on 5 term weighting methods using NB classifier with different feature dimensionalities. It is noticeable that the proposed approach outperforms all other standard methods in terms of micro-F1 and macro-F1. Figure 5 shows that macro-F1 and micro-F1 of MTF-MI are close to those of CHI when 500 and 1000 features are selected. It is noticeable that the IG and MI techniques showed the lowest performance. The micro-F1 results are noticed to be high (more than 83%) using 3500 features, and the highest classification F1 value (82%) is achieved by the MTF-MI method. Moreover, it is noticeable that proposed method performs well for less than 1500 features as its F1 values range between 54% and 64%, while the performances of other methods were very weak on the same range of features. Although the categorical documents distribution in the Reuters-21578 dataset is highly skewed, the results show that NB classifier performs better on the representation of the proposed method MTF-MI. In the Reuters-21578 dataset, the boundaries between categories are apparent. Therefore, good classification performance can be achieved with a small number of features (3500). However, when the number of selected features increases, the classification performance decreases.

Figure 6 depicts the NB classification performance on the 20-Newsgroups dataset in terms of F1 measure, where it can be seen that the trend of the micro-F1 and macro-F1 performance is similar to that in Figure 5. Similar to the results of Reuters-21578 dataset, the proposed method outperforms other standard methods in micro-F1 and macro-F1. For instance, the best three micro-F1 and macro-F1 values (90%, 91%, and 92%) are reached by the MTF-MI method based on 4000 features. In contrast to the results in Figure 5, the performance of CHI method, as seen in Figure 6, is not competing with the performance of the proposed MTF-MI for features up to 3000. For example, the micro-F1 and macro-F1 values (66% and 65%, respectively) are reached by the CHI method on 3000 features, which are still less than the corresponding values of the proposed method (87% and 86%). Finally, the documents in 20-Newsgroups are almost uniformly distributed; therefore, the micro-F1 and macro-F1 performances of different schemes are noticed to be quite similar. In addition, the measure values increase as the feature number increases, which could be due to the similarity of some categories in the 20-Newsgroups dataset. Therefore, some terms are commonly present in more than one category, so when the number of selected features increases, it provides a better distinction between categories.

Figure 7 shows micro-F1 and macro-F1 classification performance on the WebKB dataset using NB classifier. Generally, the results in Figure 7 are similar to those in Figure 5 for standard weighting techniques, as the boundaries between categories are apparent. The proposed MTF-MI method outperforms other techniques in terms of micro-F1 and macro-F1, where the maximum micro-F1 value (86%) is achieved by MTF-MI on 3500 features. Moreover, similar to the results on Reuters-21578 and 20-Newsgroups datasets, the proposed MTF-MI has outperformed other methods with noticeable performance differences.

It can be concluded that the proposed method MTF-MI performs the highest on different corpora, which indicates

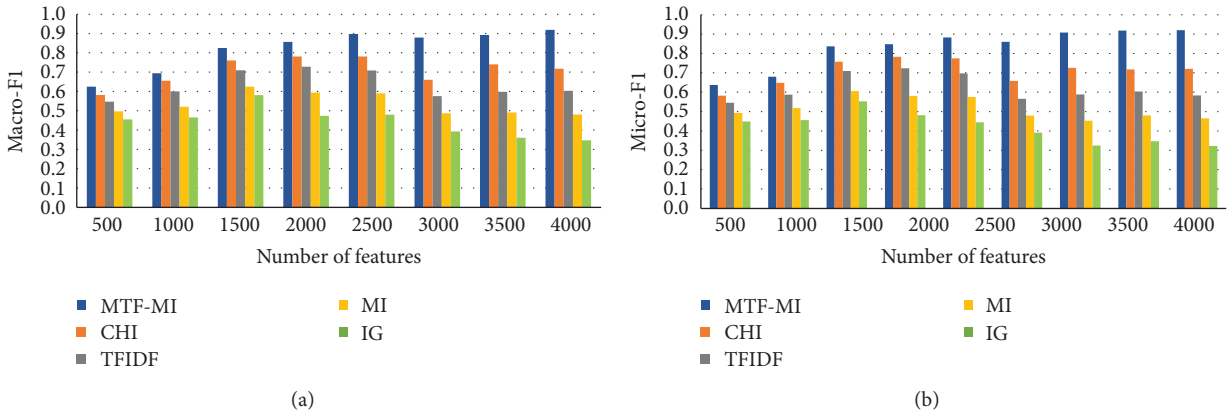

Figure 6: Macro-F1 and micro-F1 measures of multiclass text classification on the 20-Newsgroups corpus with different numbers of features.

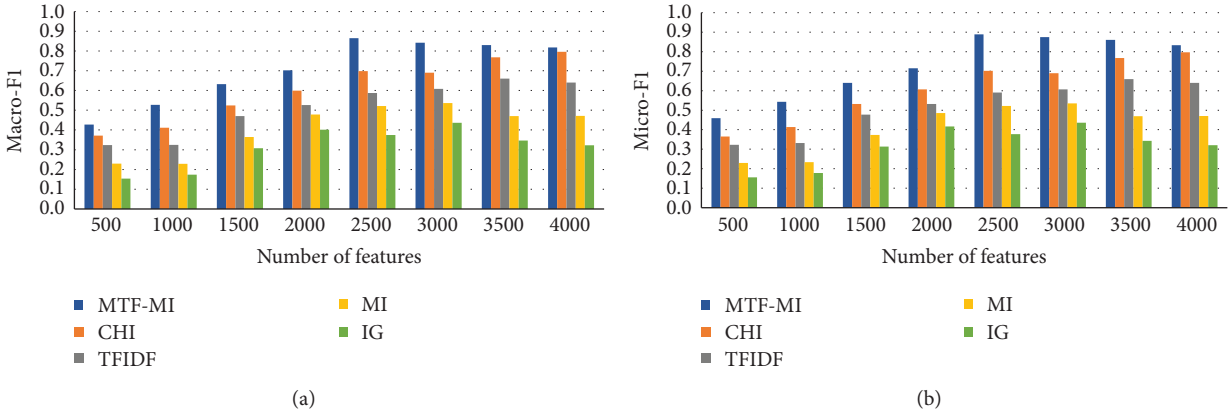

Figure 7: Macro-F1 and micro-F1 measures of multiclass text classification on the WebKB corpus with different numbers of features.

that the proposed approach is effective in selecting the features and representing the data as well as their generality. Based on the experimental results on different datasets, the performance of the proposed method is more effective for the three datasets, which means that the maximum term frequency factor introduced to the classical MI plays a big role to reach high performance. Therefore, it can be concluded that the proposed MTF-MI method is more effective than the classical state-of-the-art method.

6. Conclusion

This paper introduces MTF-MI, a distributed feature selection approach designed upon the MapReduce programming model. The proposed approach, based on mutual information method, has been implemented using Apache Hadoop, and it has been applied over three different large datasets. The performance of resulting classification models generated by MTF-MI has been systematically evaluated using Naïve Bayes classifier, implemented in Hadoop framework, over a cluster of five computers. The experimental study has proved that MTF-MI efficiently improves the selection of the relevant features while discarding the selection of irrelevant ones. The proposed approach is the best in average of F-measure compared to four state-of-the-art methods, namely, CHI, TF-IDF, MI, and IG. However, this method becomes less performed for a given threshold of selected features. Although the results vary within the datasets, the general insights provided here help highlight the importance of the combination of the feature selection techniques with the distributed aspect that is added through Hadoop framework usage for the prediction tasks on large textual datasets.

As part of this work, we have also compared the proposed approach with a sequential version of MTF-MI implemented on a single machine using java. Our results showed that the sequential version is unable to handle large datasets due to memory requirements. Meanwhile, our version is fully scalable and yields better memory usage when dealing with very large datasets. Despite the multiple advantages of parallelism, it can be hazardous if not used appropriately. When large and complex datasets are used, overparallelism can cause the distribution to ignore certain meaningful relationships between features, which can negatively affect the accuracy of the results.

Data Availability

20-Newsgroups dataset is from https://archive.ics.uci.edu/ml/datasets/Twenty+Newsgroups. Reuters-21578 dataset is from https://archive.ics.uci.edu/ml/datasets/reuters-21578+text+categorization+collection. WebKB dataset is from http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/webkb-data.gtar.gz. The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] A. Onan, "Sentiment analysis on product reviews based on weighted word embeddings and deep neural networks," *Concurrency and Computation: Practice and Experience*, vol. 10, Article ID e5909, 2020.

[2] A. Onan and S. Korukoğlu, "A feature selection model based on genetic rank aggregation for text sentiment classification," *Journal of Information Science*, vol. 43, no. 1, pp. 25–38, 2017.

[3] A. Onan, "Hybrid supervised clustering based ensemble scheme for text classification," *Kybernetes*, vol. 46, 2017.

[4] A. Onan, "Two-stage topic extraction model for bibliometric data analysis based on word embeddings and clustering," *IEEE Access*, vol. 7, pp. 145614–145633, 2019.

[5] A. Onan and M. A. Tocoglu, "A term weighted neural language model and stacked bidirectional lstm based framework for sarcasm identification," *IEEE Access*, vol. 9, pp. 7701–7722, 2021.

[6] V. P. Deshpande, R. F. Erbacher, and C. Harris, "An evaluation of naïve bayesian anti-spam filtering techniques," in *Proceedings of the 2007 IEEE SMC Information Assurance and Security Workshop*, pp. 333–340, IEEE, New York, NY, USA, June 2007.

[7] L. M. Q. Abualigah, *Feature Selection and Enhanced Krill Herd Algorithm for Text Document Clustering*, Springer, Berlin, Germany, 2019.

[8] A. Onan, "Classifier and feature set ensembles for web page classification," *Journal of Information Science*, vol. 42, no. 2, pp. 150–165, 2016.

[9] J. Zhang, Y. Xiong, and S. Min, "A new hybrid filter/wrapper algorithm for feature selection in classification," *Analytica Chimica Acta*, vol. 1080, pp. 43–54, 2019.

[10] X. Deng, Y. Li, J. Weng, and J. Zhang, "Feature selection for text classification: a review," *Multimedia Tools and Applications*, vol. 78, no. 3, pp. 3797–3816, 2019.

[11] J. Tang, S. Alelyani, and H. Liu, "Feature selection for classification: a review," *Data Classification: Algorithms and Applications*, vol. 37, 2014.

[12] H. Amazal, M. Ramdani, and M. Kissi, "A parallel global tfidf feature selection using hadoop for big data text classification," in *Advances on Smart and Soft Computing*, pp. 107–117, Springer, Berlin, Germany, 2020.

[13] A. P. Rodrigues and N. N. Chiplunkar, "A new big data approach for topic classification and sentiment analysis of twitter data," *Evolutionary Intelligence*, pp. 1–11, Springer, Berlin, Germany, 2019.

[14] B. Venkatesh and J. Anuradha, "A hybrid feature selection approach for handling a high-dimensional data," in *Innovations in Computer Science and Engineering*, pp. 365–373, Springer, Berlin, Germany, 2019.

[15] D. Glushkova, P. Jovanovic, and A. Abelló, "Mapreduce performance model for Hadoop 2.x," *Information Systems*, vol. 79, pp. 32–43, 2019.

[16] Z. Li, W. Lu, Z. Sun, and W. Xing, "A parallel feature selection method study for text classification," *Neural Computing and Applications*, vol. 28, no. 1, pp. 513–524, 2017.

[17] C. Reggiani, Y. A. Le Borgne, and G. Bontempi, "Feature selection in high-dimensional dataset using mapreduce," in *Benelux Conference on Artificial Intelligence*, pp. 101–115, Springer, Berlin, Germany, 2017.

[18] Z. Zhao, R. Zhang, J. Cox, D. Duling, and W. Sarle, "Massively parallel feature selection: an approach based on variance preservation," *Machine Learning*, vol. 92, no. 1, pp. 195–220, 2013.

[19] Z. Sun and Z. Li, "Data intensive parallel feature selection method study," in *Proceedings of the 2014 International Joint Conference on Neural Networks (IJCNN)*, pp. 2256–2262, IEEE, Beijing, China, July 2014.

[20] M. Kumar and S. Kumar Rath, "Classification of microarray using mapreduce based proximal support vector machine classifier," *Knowledge-Based Systems*, vol. 89, pp. 584–602, 2015.

[21] D. Peralta, S. Del Río, S. Ramírez-Gallego, I. Triguero, J. M. Benitez, and F. Herrera, "Evolutionary feature selection for big data classification: A mapreduce approach," *Mathematical Problems in Engineering*, vol. 2015, Article ID 246139, 11 pages, 2015.

[22] I. Tsamardinos, G. Borboudakis, P. Katsogridakis, P. Pratikakis, and V. Christophides, "A greedy feature selection algorithm for big data of high dimensionality," *Machine Learning*, vol. 108, no. 2, pp. 149–202, 2019.

[23] M. Rahmaninia and P. Moradi, "Osfsmi: online stream feature selection method based on mutual information," *Applied Soft Computing*, vol. 68, pp. 733–746, 2018.

[24] H. Lu, J. Chen, K. Yan, Q. Jin, Y. Xue, and Z. Gao, "A hybrid feature selection algorithm for gene expression data classification," *Neurocomputing*, vol. 256, pp. 56–62, 2017.

[25] C. Pascoal, M. R. Oliveira, A. Pacheco, and R. Valadas, "Theoretical evaluation of feature selection methods based on mutual information," *Neurocomputing*, vol. 226, pp. 168–181, 2017.

[26] M. Han and W. Ren, "Global mutual information-based feature selection approach using single-objective and multi-objective optimization," *Neurocomputing*, vol. 168, pp. 47–54, 2015.

[27] W. Gao, L. Hu, and P. Zhang, "Class-specific mutual information variation for feature selection," *Pattern Recognition*, vol. 79, pp. 328–339, 2018.

[28] W. Gao, L. Hu, P. Zhang, and J. He, "Feature selection considering the composition of feature relevancy," *Pattern Recognition Letters*, vol. 112, pp. 70–74, 2018.

[29] F. Macedo, M. Rosário Oliveira, A. Pacheco, and R. Valadas, "Theoretical foundations of forward feature selection methods based on mutual information," *Neurocomputing*, vol. 325, pp. 67–89, 2019.

[30] H. Amazal, M. Ramdani, and M. Kissi, "Towards a feature selection for multi-label text classification in big data," in *International Conference on Smart Applications and Data Analysis*, pp. 187–199, Springer, Berlin, Germany, 2020.

[31] B. Zhang, *Analysis and Research on Feature Selection Algorithm for Text Classification*, University of Science and Technology of China, Anhui, China, 2010.

[32] X. Tang, Y. Dai, and Y. Xiang, "Feature selection based on feature interactions with application to text categorization," *Expert Systems with Applications*, vol. 120, pp. 207–216, 2019.
[33] L. Abualigah, A. Diabat, S. Mirjalili, M. Abd Elaziz, and A. H. Gandomi, "The arithmetic optimization algorithm," *Computer Methods in Applied Mechanics and Engineering*, vol. 376, Article ID 113609, 2021.
[34] O. Aytuğ, "Sentiment analysis on twitter based on ensemble of psychological and linguistic feature sets," *Balkan Journal of Electrical and Computer Engineering*, vol. 6, no. 2, pp. 69–77, 2018.
[35] M. Jiang, Y. Liang, X. Feng et al., "Text classification based on deep belief network and softmax regression," *Neural Computing and Applications*, vol. 29, no. 1, pp. 61–70, 2018.
[36] L. M. Abualigah, A. T. Khader, and E. S. Hanandeh, "Hybrid clustering analysis using improved krill herd algorithm," *Applied Intelligence*, vol. 48, no. 11, pp. 4047–4071, 2018.
[37] G. Beatty, E. Kochis, and M. Bloodgood, "The use of unlabeled data versus labeled data for stopping active learning for text classification," in *Proceedings of the 2019 IEEE 13th International Conference on Semantic Computing (ICSC)*, pp. 287–294, IEEE, Newport Beach, CA, USA, February 2019.
[38] A. Onan, "An ensemble scheme based on language function analysis and feature engineering for text genre classification," *Journal of Information Science*, vol. 44, no. 1, pp. 28–47, 2018.
[39] H. Schütze, C. D. Manning, and P. Raghavan, *Introduction to Information Retrieval*, vol. 39, Cambridge University Press, Cambridge, UK, 2008.