Control and Cybernetics

vol. **49** (2020) No. 4

A survey of big data classification strategies*

by

Chitrakant Banchhor and N. Srinivasu

Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram, AP, India
Corresponding author: banchhorchitrakant@gmail.com

Abstract: Big data plays nowadays a major role in finance, industry, medicine, and various other fields. In this survey, 50 research papers are reviewed regarding different big data classification techniques presented and/or used in the respective studies. The classification techniques are categorized into machine learning, evolutionary intelligence, fuzzy-based approaches, deep learning and so on. The research gaps and the challenges of the big data classification, faced by the existing techniques are also listed and described, which should help the researchers in enhancing the effectiveness of their future works. The research papers are analyzed for different techniques with respect to software tools, datasets used, publication year, classification techniques, and the performance metrics. It can be concluded from the here presented survey that the most frequently used big data classification methods are based on the machine learning techniques and the apparently most commonly used dataset for big data classification is the UCI repository dataset. The most frequently used performance metrics are accuracy and execution time.

Keywords: big data, data mining, MapReduce, classification, machine learning, evolutionary intelligence, deep learning

1. Introduction

The advancements in technology have improved data acquisition in every area, including scientific and engineering activities, with special inroads into such domains as remote sensing, genomics, medical imaging, finance, but also the personal lives of the people, in association with the emergence of social media (Ulfarsson et al., 2016). The usually referred to data characteristics of the so-called "big data", which we do address here, are velocity, variety, veracity, value, and volume. The velocity is defined as the speed of processing the data. variety is defined as the multiplicity of kinds of data treated, like numbers, texts, photos, videos, and audios, veracity is defined as the degree of reliability

*Submitted: February 2020; Accepted: December 2020

that the data has to offer, value is defined as the capacity of processing of large amounts of trustworthy, reliable, and valuable data, whereas volume is defined as the extremely vast amounts of information generated in a unit of time. It is obvious that in some cases, if the variety, velocity, and volume of the data increase beyond a certain threshold, the usual techniques fail to store and adequately process the data. It is such data that are defined as big data. The statistical and geometric patterns for the respective datasets are being extracted by analyzing the characteristics of massive size (see, e.g., Suthaharan, 2014) . In addition to the challenges in storing and accessing the data, this massive growth of data leads to several other challenges in processing. Since the data collection is expensive, the systems and applications should use truly efficient algorithms for processing big data (Ulfarsson et al., 2016).

Big data (see Dessì et al., 2019; García-Gil et al., 2019) find important applications in various fields, like medicine, industry, and business. One of the major research problems is the effective data analytics, which is performed using data mining and machine learning approaches. As the data size keeps increasing, data mining becomes difficult to be effectively implemented with the current technologies and data mining software tools. The execution of the data mining process leads to high computational costs for large-scale datasets, thus making it necessary to analyze and process big data using more effective computing techniques (Tsai, Lin and Ke, 2016). The demand for smart data analytic approaches, in domains like image processing, data fusion, automatic classification, and multi-temporal processing increases due to the advent of big data. The parallelization techniques are used to significantly accelerate the computations that heavily depend on the amount of available data (Cavallaro et al., 2015; Hababeh et al., 2018). The problems, caused by dealing with large-scale datasets are handled, in particular, by the MapReduce framework (Tsai, Lin and Ke, 2016). The large-scale datasets are effectively handled by the MapReduce technique in combination with its distributed file system by providing a simple and robust environment.

The MapReduce framework is generally executed using a powerful parallel programming technique, called Hadoop. The MapReduce techniques consist of the map and reduce functions. The filtering and sorting are done using the *map* procedure, whereas the summary operation to generate the outcome is performed by the *reduce* function. The techniques, such as attribute reduction (Qian et al., 2015), and instance selection (López et al., 2015) are available in the MapReduce framework. The big data classification is built by assigning items in a collection for predicting the target class possibly accurately regarding each item (Kamal et al., 2017). The classification of big data is done using techniques like decision trees, genetic programming and genetic algorithms, Bayes networks, and so on (Arnaiz-González et al., 2017). Better generalization performance and higher training speed are provided by an extreme learning machine (ELM), compared to standard optimization approaches. However, big data

poses additional challenges to the ELM algorithms, due to its continuity and distributed blocks as they may entail large-scale training, which is hard to be finished by a common commodity machine in a truly limited time (Chen et al., 2016). Moreover, big data often feature imbalanced datasets, this issue being, for instance, overcome by Fuzzy Rule-Based Classification Systems (FRBCS) (Lopez et al., 2014), like that known as Chi-FRBCS-Big Data CS. The work reported in the literature also concerns the machine learning tools (Menaga and Revathi, 2020), fuzzy associative classifiers, k-Nearest Neighbour (k-NN) algorithm (Maillo et al., 2015), and the already mentioned Chi-FRBCS-BigData algorithm (Fernández et al., 2017) for big data classification.

The main aim of this paper is to consider the different techniques employed for big data classification. Based on the literature, this paper assigns the big data classification approaches to five categories, namely, machine learning-based approaches, evolutionary intelligence-based approaches, fuzzy-based approaches, deep learning-based approaches, and others.

The organization of the paper is as follows: Section 1 describes generally the big data classification; Section 2 describes the big data classification techniques, as they appear in the literature, Section 3 describes the challenges and the research gaps in the existing methods, Section 4 elucidates the results and discussion, and Section 5 provides the conclusion of the survey.

2. Description of big data classification methods

The specific research works that employ different classification schemes are characterized in this section. Fig. 1 shows the categorization of big data classification methodologies.

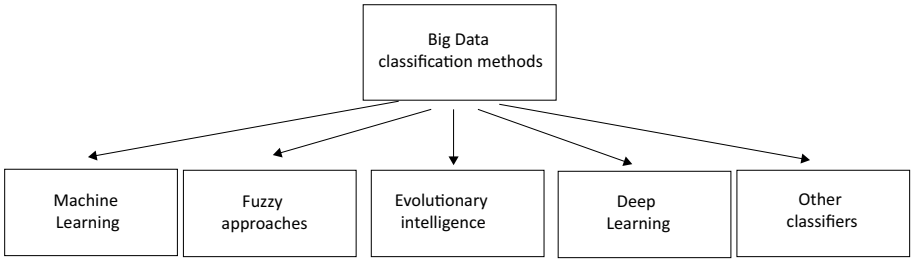

Figure 1. Categorization of big data classification techniques

2.1. Big data classification based on machine learning

Koliopoulos et al. (2015) designed a distributed Weka Spark, namely a distributed framework for Weka for data classification. This method integrated

the Spark's processing power and Weka's usability for providing a prototype that was distributed with big data workbench. The distributed Weka Spark achieved linear scaling that allowed for executing real-world scale workloads.

Scardapane, Wang and Panella (2016) developed an algorithm for a class of Recurrent Neural Network, Echo State Networks, which was a multiplier optimization procedure. In Echo State Networks, the communication between training patterns and the local exchanges between neighboring agents were not required.

Abawajy, Kelarevand and Chowdhury (2014) introduced a large iterative multitier ensemble (LIME) classifier that was developed for big data classification. These classifiers were easily generated and handled the big data classification effectively. The ensemble classifier generated an iterative system by automatically connecting several tiers simultaneously.

Xin et al. (2015) developed a MapReduce framework, known as Elastic Extreme Learning Machine (ElasticELM or E2LM). The learning ability and the training of large-scale dataset of the ELM were weak, and this was overcome by this method. The updated training data subset was used for the calculation of intermediate matrix multiplications. The output weights were obtained using these matrix multiplications.

Bhagat and Patil (2015) introduced a method for classifying multi-class imbalanced data. Initially, the binarization techniques decomposed the original dataset into subsets of binary classes. The imbalanced binary class was converted into balanced data using Synthetic Minority Over-sampling Technique (SMOTE) algorithm. Finally, the classification was done using the Random Forest classifier. This technique handled large datasets effectively.

Ulfarsson et al. (2016) developed an algorithm for the classification of 3-dimensional Magnetic Resonance Imaging (3-D MRI) images of the human brain. This algorithm was based on linear discriminant analysis (LDA). The covariance of the observed data was estimated by the noisy principal component analysis (nPCA) covariance model. The advantage of this method was that the dropped out variables did not contribute to the class separation. Anyhow, this method failed to tune the parameter fast enough.

Zhou, Wang and Wang (2012) developed a parallel Naïve Bayes classification algorithm based on MapReduce for big data classification. This method improved the performance of the original algorithm and processed the big datasets very effectively. In this method, computing resources were not adequately utilized.

Bechini, Marcelloni and Segatori (2016) designed a distributed rule-based classification method based on the MapReduce programming model. This method overcomes the time complexity and memory constraints faced by other algorithms and allows for managing the big data effectively. The Frequent Pattern (FP)-Growth algorithm's distributed version was used to mine the classification / association rules. After the classification / association rules were mined, the distributed rule pruning was performed and then the unlabeled patterns were classified using a set of surviving classification association rules. This method was, however, not implemented for a large number of datasets for investigating scalability and speedup.

Zhang et al. (2016) developed a version of k-means clustering for big data classification. In this method, the dataset was divided into several parts. The KNN classification thereby obtained was more efficient and accurate and could deal with big data. The main drawback was related to the processing time of the big data classification.

Marrón et al. (2017) introduced a method for big data classification, which is an integration of nearest neighbor, Hoeffding-trees, and gradient descent technique. In this method, the random feature function filter was used for additional predictive power. The simple gradient descent learner applied a random layer and they worked well without tuning the intensity parameter in real-time data analysis. However, this method required large amount of memory.

Triguero et al. (2016) designed an algorithm based on the Apache Spark to deal with imbalanced datasets. This method addressed the problems faced while dividing the data into multiple subsets. The small sample size effect was diminished by the in-memory operations. However, this method had problems with the extremely imbalanced datasets, where the positive class did not fit in the main memory of the computing nodes.

Lin et al. (2017) developed a heuristic bootstrap sampling method with an ensemble learning algorithm for big data classification. The bootstrap algorithm reduced the learning process and hence this approach was used in other data mining algorithms for imbalanced data, but it had accuracy below expectations.

Duan et al. (2018) designed an ELM-based technique on Spark parallel framework (SELM) for big data classification. The SELM algorithm consisted of three sub-algorithms, namely U matrix decomposition (U-PMD) algorithm, parallel V matrix decomposition (V-PMD) algorithm, and parallel hidden layer output matrix calculation (H-PMC) algorithm. The Spark provided good cache strategies, and fault tolerance.

Maillo, Triguero and Herrera (2015) developed a MapReduce-based approach along with k-NN for classification of big data. In this method, a large amount

of data was classified simultaneously. For the classification of data, the map phase was used to determine the k-NN in different data splits. Subsequently, the definitive neighbors were computed with a reduced phase. This method featured a low computational burden.

Zhai, Zhang and Wang (2017) designed an algorithm for large imbalanced data sets classification. The positive class instances of the learning phase were extended using this method. This algorithm had good scale-up and speed-up performance, but the datasets were not analyzed for the cases of multiple classes.

Chen et al. (2016) designed a MapReduce-based Extreme Learning Machine (MR-ELM) for big data classification. The distributed data blocks were trained in parallel by ELM training on the individual clusters and then the trained clusters were combined as a single-hidden layer feed forward neural network. The MR-ELM was used for the data that was distributed rather than the data that were located on one machine. The performance was not evaluated for distributed sample blocks.

Haque et al. (2014) developed a Hierarchical Stream Miner (HSMiner) using MapReduce-based approaches to address the scalability issue. This method solved the scalability issue for a large number of features. The MapReduce method along with HSMiner achieved significant speedup and scale-up. However, this method did not adapt to the changes in the data stream and the classes.

Read and Bifet (2015) designed a hybrid method for the classification of data along with additional predictive power. The hybrid algorithm developed was the combination of nearest neighbor, gradient descent methods, and Hoeffding trees along with a preprocessing approach using random feature functions. In this method, random weights and an ad-hoc choice of activation function were used for the classification of data. This method failed to adapt to the drifting concepts.

Xu et al. (2014) developed a parallel Support Vector Machine (SVM) technique, applied along with the MapReduce algorithm for email classification. The differences between traditional document classification and email foldering were evaluated. Moreover, this algorithm reduced the training time by optimization of the training data subsets from multiple participating nodes. However, this method did not use the annotated instances for the extraction of additional intelligence.

Maillo et al. (2017) designed an Iterative MapReduce solution for the k-NN algorithm using Spark. The k-NN algorithm was parallelized using Apache Spark with an iterative MapReduce process. This method had overcome the problem of runtime and memory consumption. However, it failed to process,

or adequately process, the data sets with missing values and a large number of features.

Singh et al. (2014) developed a quasi-real-time intrusion detection system. In this method, the machine learning approach was used for detecting Peer-to-Peer Botnet attacks. The distributed frameworks were build using Hive. The parallel processing power of Mahout was used to build Random Forest based Decision Tree model, which is applied to the problem of Peer-to-Peer Botnet detection in quasi-real-time.

Lakshmanaprabu et al. (2018) proposed a hierarchical framework for Social Internet of Things (SIoT) big data. This method used a Map-Reduce framework and a classifier model for the classification of data. The noise in the dataset was removed using the Gabor filter. The efficiency was improved using Hadoop Map Reduce method. However, the big data classification with the method proposed featured low accuracy.

Liu et al. (2013) applied a Naive Bayes classifier with the Hadoop framework for the classification of large datasets. The experiment was automated by integrating the Naive Bayes classifier on the Hadoop framework.

Varatharajan, Manogaran and Priyan (2018) developed an SVM model using a weighted kernel function method for the classification of the data. In this method, the inputs were the electrocardiogram (ECG) signals and the delay in ECG signals was removed using Finite Impulse Response (FIR) and Infinite Impulse Response (IIR) filters.

Ahlawat and Singh (2017) developed a machine learning technique involving the use of a decision tree for big data classification. The Map Reduce-based framework improved the efficiency of the traditional decision tree learning classifier. This method was developed for the diverse characteristics of unstructured big data.

2.2. Big data classification based on fuzzy approaches

López et al. (2015) designed a fuzzy rule-based classification system using Chi-FRBCS Big Data CS algorithm. The computational operations of the fuzzy model were done using the MapReduce framework. This method provided cost-sensitive learning techniques to address the data imbalance.

Fernández et al. (2017) introduced a MapReduce-based method for big data classification. This method analyzed the interrelation between the scarcity of the data and the number of labels of the fuzzy variables. The accuracy remained stable for high granularity level. In this method, however, the performance and the interpretability were not satisfying.

Ríoa et al. (2015) introduced a Chi-FRBCS-BigData algorithm for big data classification. This method used the linguistic fuzzy rule-based classification system for the fusion of rule bases. The map function distributed the model and then, the outputs were combined through the reduce function. It was hard to analyze the behaviour of this approach.

Banchhor and Srinivasu (2018) developed a Correlative Naïve Bayes (CNB) classifier along with fuzzy theory based operations for big data classification. Improved classifications were achieved using the MapReduce framework and the classifier. Initially, the database was transferred into the probabilistic index table, in which the rows corresponded to data items and columns to attributes.

Elkano et al. (2018) introduced the CHI-BD approach, which used the FRBCS technique for Big Data classification. This method was based on the CHI algorithm and it was meant to overcome the classification problems by introducing a distributed approach. The CHI-BD provided the same classification performance for execution of data irrespective of the number of mappers, but there was no linear effect regarding execution time, thus reducing the scale up effect.

Segatori, Marcelloni and Pedrycz (2018) introduced a distributed Fuzzy Decision Tree (FDT) learning scheme using the MapReduce programming model. However, this method was not implemented in the real time classification of data.

Bhukya and Gyani (2015) developed a fuzzy associative classification algorithm, also based on the MapReduce framework. This method extracted information from the distributed data that was stored in a distributed file system for the effective handling of the data.

Bakry, Safwat and Hegazy (2016) developed another MapReduce paradigm based approach for big data classification. The fuzzy and non-fuzzy techniques were used in the respective mapper. Although this method had better performance compared to the fuzzy rule-based classification system considered, it had low efficiency, high execution time, and low accuracy in its computations.

Lopez et al. (2014) introduced a linguistic fuzzy rule-based classification system, Chi-FRBCS-BigData for classification of large amounts of data. This method was, again, based on the MapReduce framework and two versions had been developed Chi-FRBCSBigData-Max and Chi-FRBCS-BigData-Ave. In this method, the variety and veracity of the big data were dealt with including the inherent uncertainty. On the other hand, though, this method required a high number of mappers.

Ludwig (2015) designed a specialized Fuzzy C-Means (FCM) algorithm for classification of big data. This method used scalable solutions and it outperformed the hard clustering algorithms. For the parallelization of the FCM algorithm, two MapReduce algorithms were used and the calculations of the centroids were performed before the membership matrix was established. However, this method was not used for the larger data set sizes at the scale of GBs of data.

Jin, Peng and Xie (2017) developed a classification method for big data using dynamic fuzzy inference approach named BigData-DFRI. In this method, the fuzzy rule interpolation (FRI) and fuzzy rule inference (CRI) were integrated. Here, a high number of mappers provided for faster results without degrading the performance. However, this method slowed down its performance while combining multiple data sources.

2.3. Big data classification based on evolutionary intelligence

Dagdia (2019) developed a distributed Dendritic Cell Algorithm, based, again, on the MapReduce framework, for data classification. This algorithm was distributed using a cluster of computing elements. The drawback of this method was not considering the input class data order and it was not tested on real-world applications.

Cao et al. (2016) applied a particle swarm optimization (PSO) and optimized Back-Propagation (BP) neural network for big data classification using the MapReduce framework. This method improved the accuracy and efficiency of the runtime of the BP neural network algorithm. The PSO algorithm improved the accuracy of the classification algorithm by the optimization of the BP neural network's thresholds and the values of the initial weights. The parallel processing was achieved by the MapReduce parallel programming model.

Triguero et al. (2015) introduced a parallel model for large-scale classification of data. The MapReduce scheme had distributed the algorithm in a cluster of computing elements. The undersampling process was sped up by the windowing approach. The windowing technique along with the MapReduce process reduced the building time to generate the resulting learned model. The accuracy of this approach can be further improved by adding hybrid undersampling techniques.

Lin et al. (2016) developed an approach based on the traditional Cat Swarm Optimization (CSO) algorithm for the classification of big data. This method had selected the features with high classification accuracy and solved the Travelling Salesman Problem (TSP).

Satish and Kavya (2014) introduced firefly algorithm and naïve Bayes classifier for purposes of big data classification. The feature space was provided by

the firefly algorithm and the classification of data was done through naïve Bayes classifier. These two processes were distributed effectively using the MapReduce framework. This method was validated by the Twitter data sets based on such metrics as accuracy, specificity, sensitivity, and time.

Fong, Wong and Vasilakos (2016) designed an Accelerated PSO for the classification of collected data streams that were assumed to represent an instance of Big Data. This method managed the high dimensionality and streaming format of the data feeds. The performance was evaluated for the actual big data that had a large degree of dimensionality. However, this method required more computational time.

Demidova, Nikulchev and Sokolova (2016) developed a modified PSO algorithm. In this method, the kernel functions of the particle were changed according to the best value. The data classification problems were solved with reduction of the time expenditure of the classifiers. The high quality of data classification was provided, in this context, by the SVM classifier. The SVM (Beno et al., 2014; Thomas and Rangachar, 2019) ensembles were based on the decorrelation maximization algorithm. However, this algorithm could not handle high volumes of unstructured data.

2.4. Big data classification based on Deep Learning

Shafiqand and Torunski (2017) designed a methodology for organizing logs, in which the Bayesian deep learning network based analysis was utilized for the detection of any possible faults. This method processed the logs on cloud platforms and scaled the big data effectively.

2.5. Big data classification based on other types of classifiers

Subramaniyaswamy et al. (2015) introduced a MapReduce based approach for the classification of unstructured data. In the distributed clusters, large volumes of data were processed in parallel as small chunks. The filtering, storage structure, and aggregation were done using the MapReduce framework. The data was parsed into tokens using sentiment analysis through natural language processing. However, this method did not deal with emoticon-based clustering.

Gao and Gao (2014) developed an effective Map-Reduce based information retrieval system, which processed information in a distributed way. This method provided low time complexity, better stability, and extensional performance.

Qian et al. (2015) designed a hierarchical attribute reduction algorithm for the classification of big data using MapReduce. The granular computing was introduced for concept ascension and then, the encoded decision table was defined. This algorithm processed big data efficiently on commodity computers.

Kamal et al. (2017) introduced De-Bruijn graph for the classification of data with the MapReduce framework. The graph-based approach handled genomic diversity easily. The optimal paths for the genome assembly were found using the De-Bruijn graph and they were the compact representation of k-mers. The sequences were classified based on their similarity using the purity of clustering and Jaccard similarity. This method performed faster when the elements of the classifier were extended. It provided better accuracy for metagenomic data clustering.

Arnaiz-González et al. (2017) developed a parallel implementation of the Democratic Instance Selection (DIS). In this version of the method, the DIS algorithm was designed using the MapReduce model and the implementation was done using the big data framework Spark. This method had better computational complexity and internal structure.

Patil and Sonavane (2017) introduced a classification method for multiclass imbalanced data sets. The balanced data were analyzed using various classifiers. The method used imbalanced data sets with improved oversampling that enhanced the classification. The method was implemented using two approaches, namely the non-clustered and the cluster based, advanced approach. This approach did not deal with unstructured data.

3. Research gaps and challenges

The challenges faced in the context of big data classification are as follows: In the big data, the major problem lies in the development and construction of the algorithms that recognize effectively and efficiently the data patterns for classification. These issues are commonly found in statistical learning, and statistical learning methodologies can be categorized into three kinds of learning, namely reinforcement, supervised, and unsupervised learning (Ulfarsson et al., 2016). The data from the internet sources require being put into some proper structure and further, the features vary for different fields, and thus, the preprocessing of the data is used, frequently with application of some standard normalization techniques before they are subject to classification. However, the traditional classifiers do not classify the continuous data streams (Haque et al., 2014). Most of the classification learning algorithms deal with small datasets and cannot handle parallelization, computational complexity, memory usage, and so on (Bechini, Marcelloni and Segatori, 2016).

The SVM-based MapReduce scheme helps in big data classification. The main problem of SVM lies in the selection of appropriate kernel parameters, which determines the classification accuracy. For effective classification, one of the important factors considered is computational overhead. The computational overhead should be possibly low for effective classification, this being achieved

using parallelization in classification algorithms (Ludwig, 2015). Each big data base contains differently featured datasets and thus, the training model developed for a particular domain or dataset usually fails to classify the data from other domains (Bakry, Safwat and Hegazy, 2016). One of the prime challenges in big data classification is the exponential growth of data volumes, which can be overcome by building parallel fuzzy systems (López et al., 2015).

One of the other major challenges in big data classification is the minimization of computational risk, mainly related to misclassification. The computational risk can be minimized by reducing the multidimensional problems and the classification performance can be improved by the maximization of training data.. Another significant challenge is designing a classifier that selects the feasible model parameters for big data classification. The big data classification can also be improved by designing the appropriate fitness functions, like those based on the posterior probability and probability index table. In machine learning and data mining, the extraction of knowledge with high efficiency is an important challenge. Knowledge extraction can be improved by including the elements of fuzzy theory in the classifier along with the MapReduce framework. The classification performance can be maximized using the assumption of existence of dependent features.

4. Results and discussion

This section provides the summary image of the survey here reported, based on different techniques for big data classification, publication year, adapted methods, datasets used, evaluation parameters, performance metrics, software tools, and accuracy.

4.1. Analysis based on publication year

This section contains the summary of the survey, based on the years of appearance of the respective works. This is shown in Fig. 2 for the total of 50 research papers that have been taken into consideration. It can be easily seen that the biggest number of papers included was published in the year 2016.

4.2. Analysis based on classification techniques

In this section we provide the overview of the classification methods, proposed and/or analyzed in the research papers surveyed. Figure 3 shows the distribution of the broad categories of classification methods employed in big data classification, conform to the content of the papers surveyed. Thus, 51% of the studies considered employed machine learning, 14% were based on evolutionary intelligence, 23% referred to the fuzzy-based approaches, 2% (one paper) employed deep learning, and the remaining 10% were based on other methods.

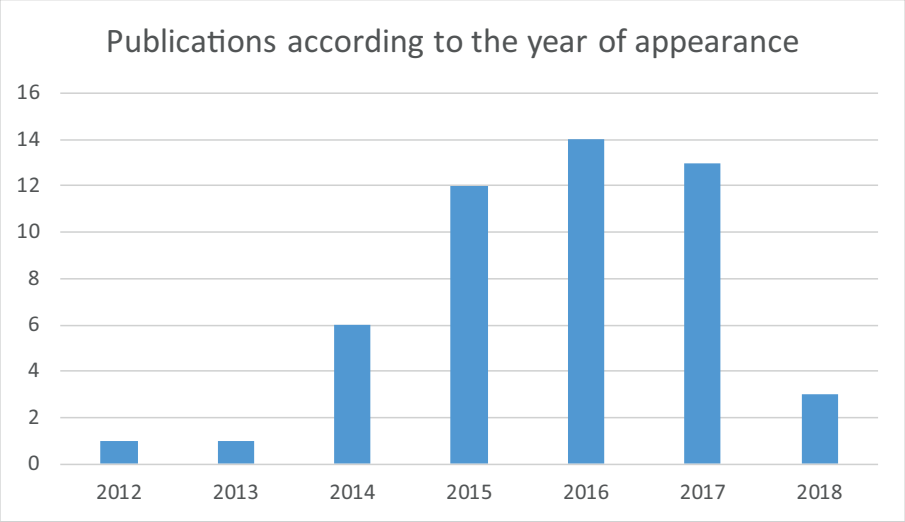

Figure 2. Statistics based on publication year

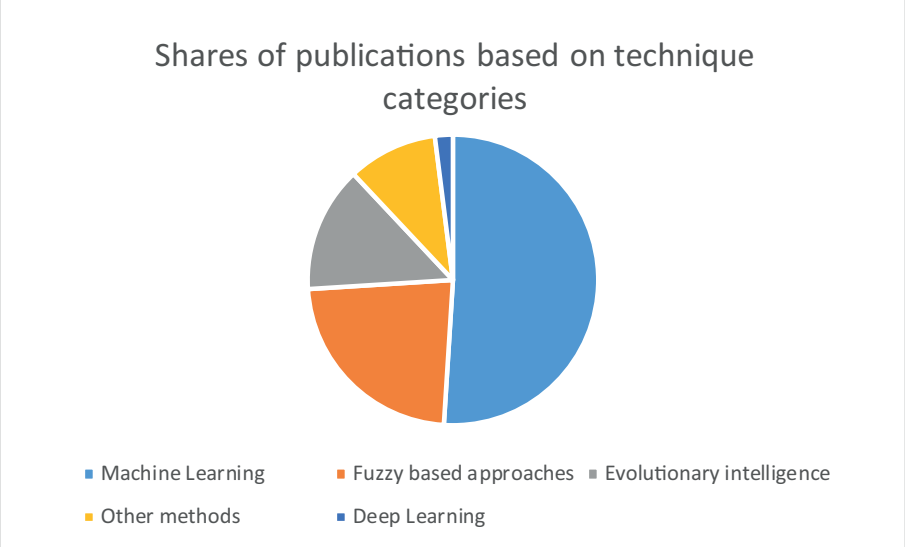

Figure 3. Statistics based on classification technique categories

4.3. Analysis based on software tools used

This section provides the overview of software tools used the in the research reported. Table 1 lists the software tools employed in the efforts aimed at the effective classification of data. The major software tools employed for the analysis of the research papers are Java platform, Hadoop framework, Apache Hadoop, Apache Spark, Knowledge Extraction based on Evolutionary Learning (KEEL) tools, Spark Mllib and Matlab. From the table, it appears that the Hadoop framework is the most commonly used software tool for big data classification.

Table 1. Analysis based on software tools

Implementation tools	*Corresponding references*
Java platform	Qian et al. (2015); Banchhor & Srinivasu (2018); Duan et al. (2018); Fong, Wong & Vasilakos (2016)
Hadoop framework	Subramaniyaswamy et al. (2015); Koliopoulos et al. (2015); Bhagat & Patil (2015); Fernández et al. (2017); Zhou, Wang & Wang (2012); Qian et al. (2015); Ríoa et al. (2015); Cao et al. (2016); Bechini, Marcelloni & Segatori (2016); Duan et al. (2018); Elkano et al. (2018); Zhai, Zhang & Wang (2017); Patil & Sonavane (2017); Bhukya & Gyani (2015); Chen et al. (2016); Haque et al. (2014); Lopez et al. (2014); Ludwig (2015); Triguero et al. (2015)
C++	Demidova, Nikulchev & Sokolova (2016)
Spark Mllib	Lin et al. (2017)
KEEL tools	López et al. (2015)
Matlab	Ulfarsson et al. (2016)
Weka tool	Abawajy, Kelarevand & Chowdhury (2014)
Apache hadoop	Ludwig (2015); Bakry, Safwat & Hegazy (2016); Maillo, Triguero & Herrera (2015); Triguero et al. (2016)
Apache Spark	Maillo et al. (2017); Arnaiz-González et al. (2017); Duan et al. (2018)

4.4. Analysis based on datasets employed

This section provides the overview of the datasets used in the surveyed research works. Table 2 shows the various data sets employed as instances for big data classification and their corresponding references. The most frequently used dataset is the UCI repository dataset. Other datasets used are Twitter Datasets, KDDCup1999, TanCorpMinTest, Scene UNderstanding (SUN) Database, KEEL repository, NARMA-10 dataset, Amazon Elastic Compute

Cloud (Amazon EC2), followed by more specific ones, i.e. the simulated datasets and the imaging genetics datasets, localization and skin segmentation datasets, FCUP repository, fixed-size log dataset, synthetic data sets, and other real-world databases than those already mentioned.

4.5. Analysis based on performance metrics

This section reviews the performance metrics used for assessing the methods meant to classify big data. Table 3 elaborates on the performance metrics used in the research works surveyed here. The most commonly used performance metric is accuracy and other performance metrics are sensitivity, area under the curve (AUC), classification time, G-mean, specificity, mean absolute error, mean square deviation, F-measure, and speed.

4.5.1. Analysis based on accuracy in terms of percentage ranges

In this section, the categorization according to the achieved accuracy values is presented. Table 4 shows the respective distribution of results, based on accuracy using 6 ranges, namely 30-40%, 50-60%, 60-70%, 70%-80%, 80%-90% and 90%-100%, respectively. It can be seen from the table that the methodologies proposed in Bhagat and Patil (2015), Banchhor and Srinivasu (2018),Read and Bifet (2015), Lakshmanaprabu et al. (2018), and Liu et al. (2013) attained accuracy in the highest of the ranges, 90%-100%, while the approaches, presented in Fernández et al. (2017), Ríoa et al. (2015), Marrón et al. (2017), and Lopez et al. (2014) achieved a somewhat lower values of accuracy, falling into the range of 80%-90%.

4.5.2. Analysis based on execution time

This section presents the results, reported in the publications surveyed, concerning the execution time for the particular big data classification techniques. Most of the research papers surveyed reported the execution time between 50s and 500s for the datasets analysed, as this can be seen in Table 5.

5. Conclusion

In this paper, an ample survey of papers, presenting big data classification methods is presented. The survey encompasses 50 research papers and the methods considered in them are categorized into machine learning, evolutionary intelligence, fuzzy-based approaches, deep learning, and so on. The merits and demerits of the surveyed approaches are shortly discussed. Moreover, the research gaps and the challenges faced by the big data classification techniques are suggested for the effective future research and development work. The surveyed works were collected from IEEE, Google scholar, and other sources.

Table 2. Analysis based on datasets

Datasets	**Corresponding references**
UCI repository dataset	Fernández et al. (2017); Zhou, Wang & Wang (2012); Qian et al. (2015); Ríoa et al. (2015); Bechini, Marcelloni & Segatori (2016); Marrón et al. (2017); Elkano et al. (2018); Zhai, Zhang & Wang (2017); Patil and Sonavane (2017); Bhukya & Gyani (2015); Chen et al. (2016); Haque et al. (2014); Lopez et al. (2014); Lin et al. (2016); Read & Bifet (2015); Jin, Peng & Xie (2017); Shafiqand & Torunski (2017); Maillo et al. (2017); Daggdia (2019); Arnaiz-González et al. (2017); Kamal et al. (2017); Demidova, Nikulchev & Sokolova (2016); Segatori, Marcelloni & Pedrycz (2018)
Twitter Datasets	Subramaniyaswamy et al. (2015); Satish & Kavya (2014); Lakshmanaprabu et al. (2018)
KDDCup1999	López et al. (2015); Triguero et al. (2015, 2016)
TanCorpMinTest	Gao & Gao (2014)
Amazon EC2	Koliopoulos et al. (2015)
NARMA-10 dataset	Scardapane, Wang & Panella (2016)
Simulated datasets and Imaging genetics datasets	Ulfarsson et al. (2016)
SUN Database	Cao et al. (2016)
Localization and skin segmentation datasets	Banchhor & Srinivasu (2018)
MNIST Database	Zhang et al. (2016); Duan et al. (2018)
PokerHand	Maillo et al. (2015); Bakry, Safwat & Hegazy (2016)
KEEL repository	Patil & Sonavane (2017)
FCUP repository	Chen et al. (2016)
Cover type dataset	Bakry, Safwat & Hegazy (2016); Ludwig (2015)
probabilistic numerical dataset (PND)	Fong, Wong & Vasilakos (2016)
Synthetic Data Sets	Demidova, Nikulchev & Sokolova (2016); Zhang et al. (2016)
Enron email set datasets	Xu et al. (2014)
Cornell University movie review dataset and Stanford SNAP Amazon movie review dataset	Liu et al. (2013)
Fixed size log dataset	Shafiqand & Torunski (2017)
Airlines dataset from MOA	Ahlawat & Singh (2017)
CAIDA dataset	Lakshmanaprabu et al. (2018)
UCSD (University of California San Diego) dataset	Singh et al. (2014)

Table 3. Analysis based on performance metrics

Performance metrics	*References*
Accuracy	López et al. (2015); Bhagat & Patil (2015); Fernández et al. (2017); Ríoa et al. (2015); Cao et al. (2016); Zhang et al. (2016); Marrón et al. (2017); Triguero et al. (2016); Banchhor & Srinivasu (2018); Kamal et al. (2017); Arnaiz-González et al. (2017); Maillo, Triguero & Herrera (2015); Haque et al. (2014); Bakry, Safwat & Hegazy (2016); Lopez et al. (2014); Fong, Wong & Vasilakos (2016); Read & Bifet (2015); Xu et al. (2014); Singh et al. (2014); Lakshmanaprabu et al. (2018); Liu et al. (2013); Ahlawat & Singh (2017)
Sensitivity	Banchhor & Srinivasu (2018); Lakshmanaprabu et al. (2018); Ahlawat & Singh (2017)
AUC	Abawajy, Kelarevand & Chowdhury(2014); Elkano et al. (2018); Triguero et al. (2015); Demidova, Nikulchev & Sokolova (2016)
Classification time	Triguero et al. (2015)
G-mean	Zhai, Zhang & Wang (2017); Triguero et al. (2015)
Specificity	Banchhor & Srinivasu (2018); Ahlawat & Singh (2017)
Mean absolute error	Varatharajan, Manogaran & Priyan (2018); Chen et al. (2016); Triguero et al. (2015)
Mean square deviation	Varatharajan, Manogaran & Priyan (2018)
F-measure	Bhagat & Patil (2015); Lin et al. (2017)
Speed	Gao & Gao (2014); Bechini, Marcelloni & Segatori (2016); Kamal et al. (2017); Arnaiz-González et al. (2017); Lakshmanaprabu et al. (2018)
Execution time	López et al. (2015); Ríoa et al. (2015); Bechini, Marcelloni & Segatori (2016); Zhang et al. (2016); Maillo, Triguero & Herrera (2015); Read & Bifet (2015); Maillo et al. (2017)
ROC	Patil & Sonavane (2017)

Table 4. Analysis based on accuracy values

Accuracy ranges	*References*
30%-40%	Fong, Wong & Vasilakos (2016)
50%-60%	Maillo, Triguero & Herrera (2015)
60%-70%	Zhang et al. (2016); Kamal et al. (2017); Ahlawat & Singh (2017)
70%-80%	Bakry, Safwat & Hegazy(2016)
80%-90%	Fernández et al. (2017); Ríoa et al. (2015); Marrón et al. (2017); Lopez et al. (2014)
90%-100%	Bhagat & Patil (2015); Banchhor & Srinivasu (2018); Read & Bifet (2015); Lakshmanaprabu et al. (2018); Liu et al. (2013)

Table 5. Analysis based on execution time

Execution time	*References*	*Datasets*
50s -500s	López et al. (2015); Read & Bifet (2015); Zhang et al. (2016); Arnaiz-González et al. (2017)	López et al. (2015): KDDCup1999, Read & Bifet (2015), Arnaiz-González et al. (2017): UCI repository dataset, Zhang et al. (2016): MNIST Database
600-6000s	Bechini, Marcelloni & Segatori (2016); Maillo, Triguero & Herrera (2015)	Bechini, Marcelloni & Segatori (2016): UCI repository dataset, Maillo, Triguero & Herrera (2015): PokerHand
>6000s	Ríoa et al. (2015); Singh et al. (2014)	Ríoa et al. (2015): UCI repository dataset, Singh et al. (2014): UCSD dataset

Simple statistics are then presented referring to software tools used, datasets, publication year, classification techniques, and performance metrics. From this survey, it can be concluded that the most frequently used big data classification methodologies come from the machine learning domain, the most commonly used dataset is the UCI repository dataset, while the most common metrics are accuracy and execution time.

References

Abawajy, J.H., Kelarevand, A. and Chowdhury, M. (2014) Large Iterative Multitier Ensemble Classifiers for Security of Big Data. *IEEE Transactions on Emerging Topics in Computing,* **2**(3), 352 – 363.

Ahlawat, K. and Singh, A. P. (2017) A Novel Hybrid Technique for Big Data Classification Using Decision Tree Learning. In: *Proceedings of the International Conference on Computational Intelligence, Communications, and Business Analytics*. Springer, 118-128.

Arnaiz-González, Á., González-Rogel, A., Díez-Pastor, J-F. and López-Nozal, C. (2017) MR-DIS: democratic instance selection for big data by MapReduce. *Progress in Artificial* Intelligence, **6**(3), 211–219.

Bakry, M.E., Safwat, S. and Hegazy, O. (2016) A Mapreduce Fuzzy Techniques of Big Data Classification. In: *Proceedings of SAI Computing Conference, London, UK*. Springer, 13-15.

Banchhor, C. and Srinivasu, N. (2018) FCNB: Fuzzy Correlative Naive Bayes Classifier with MapReduce Framework for Big Data Classification. *Journal of Intelligent Systems,* **29**(1).

Bechini, A., Marcelloni, F. and Segatori, A. (2016) A MapReduce Solution for Associative Classification of Big Data. *Information Sciences,* 332, 33-55.

Beno, M. M., Valarmathi I. R., Swamy S. M. and Rajakumar, B. R. (2014) Threshold prediction for segmenting tumour from brain MRI scans. *International Journal of Imaging Systems and Technology,* **24**(2), 129-137.

Bhagat, R.C. and Patil, S.S. (2015) Enhanced SMOTE Algorithm for Classification of Imbalanced Big-Data using Random Forest. In: *Proceedings of IEEE International Advance Computing Conference (IACC)*. IEEE.

Bhukya, R. and Gyani, B.J. (2015) Fuzzy Associative Classification Algorithm Based on MapReduce Framework. In: *Proceedings of the International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT)*. IEEE.

Cao, J., Cui, H., Shi, H. and Jiao, L. (2016) Big Data: A Parallel Particle Swarm Optimization-Back-Propagation Neural Network Algorithm Based on MapReduce. *PloS One,* **11**(6).

Cavallaro, G., Riedel, M., Richerzhagen, M., Benediktsson, J.A. and Plaza, A. (2015) On Understanding Big Data Impacts in Remotely Sensed Image Classification Using Support Vector Machine Methods. *IE-*

EE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, **8**(10), 4634-4646.

Chen, J., Chen, H., Wan, X. and Zheng, G. (2016) MR-ELM: a MapReduce-based framework for large-scale ELM training in big data era. *Neural Computing and Applications,* **27**(1), 101–110.

Dagdia, Z.C. (2019) A scalable and distributed dendritic cell algorithm for big data classification. *Journal of Swarm and Evolutionary Computation,* 50.

Demidova, L., Nikulchev, E. and Sokolova, Y. (2016) Big Data Classification Using the SVM Classifiers with the Modified Particle Swarm Optimization and the SVM Ensembles. *International Journal of Advanced Computer Science and Applications,* **7**(5).

Dessì, D., Fenu, G., Marras, M. and Recupero, D.R. (2019) Bridging learning analytics and Cognitive Computing for Big Data classification in micro-learning video collections. *Computers in Human Behavior,* 92, 468-477.

Duan, M., Li, K., Liao, X. and Li, K. (2018) A Parallel Multiclassification Algorithm for Big Data Using an Extreme Learning Machine. *IEEE Transactions on Neural Networks and Learning Systems,* **29**(6), 2337–2351.

Elkano, M., Galar, M., Sanz, J. and Bustince, H. (2018) CHI-BD: A Fuzzy Rule-Based Classification System for Big Data classification problems. *Fuzzy Sets and Systems*, 348, 75-101.

Fernández, A., Río, S., Bawakid, A. and Herrera, F. (2017) Fuzzy rule based classification systems for big data with MapReduce: granularity analysis. *Advances in Data Analysis and Classification,* **11**(4), 711–730.

Fong, S., Wong, R. and Vasilakos, A.V. (2016) Accelerated PSO Swarm Search Feature Selection for Data Stream Mining Big Data. *IEEE Transactions on Services Computing,* **9**(1), 33 – 45.

Gao, Sh. and Gao, K. (2014) Modelling on Classification and Retrieval Strategy in Map-Reduce Based IR System. In: *Proceedings of 2014 International Conference on Modelling, Identification & Control, Melbourne, Australia.* IEEE, 322-325.

García-Gil, D., Luengo, J., García, S. and Herrera, F. (2019) Enabling Smart Data: Noise filtering in Big Data classification. *Information Sciences,* 479, 135-152.

Hababeh, I., Gharaibeh, A., Nofal, S. and Khalil, I. (2018) An Integrated Methodology for Big Data Classification and Security for Improving Cloud Systems Data Mobility. *IEEE Access,* 7, 9153 – 9163.

Haque, A., Parker, B., Khan, L. and Thuraisingham, B. (2014) Evolving Big Data Stream Classification with MapReduce. In: *Proceedings of IEEE 7th International Conference on Cloud Computing.* IEEE, 570-577.

Jin, S., Peng, J. and Xie, D. (2017) Towards MapReduce Approach with Dynamic Fuzzy Inference/Interpolation for Big Data Classification Problems. In: *Proceedings of the IEEE 16th International Conference on Cog-*

*nitive Informatics & Cognitive Computing (ICCI*CC).* IEEE.

Kamal, S., Parvin, S., Ashour, A.S., Shi, F. and Dey, N. (2017) De-Bruijn graph with MapReduce framework towards metagenomic data classification. *International Journal of Information Technology,* **9**(1), 59–75.

Koliopoulos, A-K., Yiapanis, P., Tekiner, F., Nenadic, G. and Keane, J. (2015) A Parallel Distributed Weka Framework for Big Data Mining using Spark. In: *Proceedings of IEEE International Congress on Big Data.* IEEE.

Lakshmanaprabu, S.K., Shankar, K., Khanna, A., Gupta, D., Rodrigues, D.J.J. and Albuquerque, V.H.C.D. (2018) Effective Features to Classify Big Data Using Social Internet of Things. *IEEE Access,* 6, 24196-24204.

Lin, K-C., Zhang, K-Y., Huang, Y-H., Hung, J.C. and Yen, N. (2016) Feature selection based on an improved cat swarm optimization algorithm for big data classification. *The Journal of Supercomputing,* **72**(8), 3210–3221.

Lin, W., Wu, Z., Lin, L., Wen, A. and Li, J. (2017) An Ensemble Random Forest Algorithm for Insurance Big Data Analysis. *IEEE Access,* 5, 16568–16575.

Liu, B., Blasch, E., Chen, Y., Shen, D. and Chen, G. (2013) Scalable Sentiment Classification for Big Data Analysis Using Naıve Bayes Classifier. In: *Proceedings of the IEEE International Conference on Big Data.* IEEE.

Lopez, V., Rio, S., Benitez, J.M. and Herrera, F. (2014) On the use of MapReduce to build Linguistic Fuzzy Rule Based Classification Systems for Big Data. In: *Proceedings of IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), Beijing, China.* IEEE.

López, V., Río, S., Benítez, J.M. and Herrera, F. (2015) Cost sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data. *Fuzzy Sets and Systems,* 258, 5–38.

Ludwig, S.A. (2015) MapReduce-based fuzzy c-means clustering algorithm: implementation and scalability. *International Journal of Machine Learning and Cybernetics,* **6**(6), 923–934.

Maillo, J., Ramírez, S., Triguero, I. and Herrera, F. (2017) kNN-IS: An Iterative Spark-based design of the k-Nearest Neighbors Classifier for Big Data. *Knowledge-Based Systems,* 117, 3-15.

Maillo, J., Triguero, I. and Herrera, F. (2015) A MapReduce-based k-Nearest Neighbor Approach for Big Data Classification. *IEEE Trustcom / BigDataSE / ISPA, Helsinki, Finland.* IEEE.

Marrón, D., Read, J., Bifet, A.T. and Navarro, N. (2017) Data stream classification using random feature functions and novel method combinations. *The Journal of Systems and Software,* 127, 195-204.

Menaga, D. and Revathi, S. (2020) Deep Learning: A Recent Computing Platform for Multimedia Information Retrieval. In: *Deep Learning Techniques and Optimization Strategies in Big Data Analytics,* 124-141.

Patil, S.S. and Sonavane, S.P. (2017) Enriched Over-Sampling Techniques for Improving Classification of Imbalanced Big Data. In: *Proceedings of IEEE Third International Conference on Big Data Computing Service and Applications.* IEEE.

Qian, J., Lv, P., Yue, X., Liu, C. and Jing, Z. (2015) Hierarchical attribute reduction algorithms for big data using MapReduce. *Journal of Knowledge-Based Systems,* 73, 18-31.

Read, J. and Bifet, A. (2015) Data Stream Classification using Random Feature Functions and Novel Method Combinations. *IEEE Trustcom / BigDataSE / ISPA, Helsinki, Finland.* IEEE.

Ríoa, S.D., López, V., Benítez, J.M. and Herrera, F. (2015) A MapReduce Approach to Address Big Data Classification Problems Based on the Fusion of Linguistic Fuzzy Rules. *International Journal of Computational Intelligence Systems,* **8**(3), 422-437.

Satish, K.V.R and Kavya, N. P. (2014) Big Data Processing with harnessing Hadoop - MapReduce for Optimizing Analytical Workloads. In: *Proceedings of the International Conference on Contemporary Computing and Informatics* (IC3I). IEEE.

Scardapane, S., Wang, D. and Panella, M. (2016) A decentralized training algorithm for Echo State Networks in distributed big data applications. *Neural Networks,* 78, 5–74.

Segatori, A., Marcelloni, F. and Pedrycz, W. (2018) On Distributed Fuzzy Decision Trees for Big Data. *IEEE Transactions on Fuzzy Systems,* **26**(1), 174-192.

Shafiqand, M.O. and Torunski, E. (2017) Towards Map Reduce based Bayesian Deep Learning Network for Monitoring Big Data Applications. In: *Proceedings of the IEEE International Conference on Big Data (BIGDATA).* IEEE.

Singh, K., Guntuku, S.C., Thakur, A. and Hota, C. (2014) Big Data Analytics framework for Peer-to-Peer Botnet detection using Random Forests. *Information Sciences,* 278, 488-497.

Subramaniyaswamy, V., Vijayakumar, V., Logesh, R. and Indragandhi, V. (2015) Unstructured Data Analysis on Big Data using Map Reduce. In: *Proceedings of the 2nd International Symposium on Big Data and Cloud Computing (ISBCC'15). Procedia Computer Science,* 50, 456-465.

Suthaharan, S. (2014) Big data classification: Problems and challenges in network intrusion prediction with machine learning. *ACM SIGMETRICS Performance Evaluation Review,* **41**(4), 70-73.

Thomas, R. and Rangachar, M.J.S. (2019) Fractional Rider and Multi-Kernel-Based Spherical SVM for Low Resolution Face Recognition. *Multimedia Research,* **2**(2), 35-43.

Triguero, I., Galar, M., Vluymans, S., Cornelis, C., Bustince, H., Herrera, F. and Saeys, Y. (2015) Evolutionary Undersampling for Imbalanced Big Data Classification. In: *Proceedings of IEEE Congress on*

Evolutionary Computation (CEC). IEEE.

Triguero, I., Galar, M., Merino, D., Maillo, J., Bustince, H. and Herrera, F. (2016) Evolutionary Undersampling for Extremely Imbalanced Big Data Classification under Apache Spark. In: *Proceedings of IEEE Congress on Evolutionary Computation (CEC), Vancouver, BC, Canada*. IEEE.

Tsai, C-F., Lin, W-C. and Ke, S-W. (2016) Big data mining with parallel computing: A comparison of distributed and MapReduce methodologies. *Journal of Systems and Software,* 122, 83–92.

Ulfarsson, M.O., Palsson, F., Sigurdsson, J. and Sveinsson, J.R. (2016) Classification of Big Data With Application to Imaging Genetics. *Proceedings of the IEEE,* **104**(11), 2137-2154.

Varatharajan, R., Manogaran, G. and Priyan, M. K. (2018) A big data classification approach using LDA with an enhanced SVM method for ECG signals in cloud computing. *Multimedia Tools and Applications,* **77**(8), 10195–10215.

Xin, J., Wang, Z., Qu, L. and Wang, G. (2015) Elastic extreme learning machine for big data classification. *Neurocomputing,* 149, 464–471.

Xu, K., Wen, C., Yuan, Q., He, X. and Tie, J. (2014) A MapReduce based Parallel SVM for Email Classification. *Journal of Networks,* **9**(6), 1640-1647.

Zhai, J., Zhang, S. and Wang, C. (2017) The classification of imbalanced large data sets based on MapReduce and ensemble of ELM classifiers. *International Journal of Machine Learning and Cybernetics,* **8**(3), 1009–1017.

Zhang, S., Deng, Z., Cheng, D., Zong, M. and Zhu, X. (2016) Efficient kNN Classification Algorithm for Big Data. *Neurocomputing,* 195, 143-148.

Zhou, L., Wang, H. and Wang, W. (2012) Parallel Implementation of Classification Algorithms Based on Cloud Computing Environment. *TELKOMNIKA Indonesian Journal of Electrical Engineering,* **10**(5), 1087-1092.