International Journal of Computer Vision and Image Processing
Volume 7 • Issue 3 • July-September 2017

A Review of Infrastructures to Process Big Multimedia Data

Jaime Salvador, Universidad Central del Ecuador, Quito, Ecuador
Zoila Ruiz, Universidad Central del Ecuador, Quito, Ecuador
Jose Garcia-Rodriguez, University of Alicante, Alicante, Spain

ABSTRACT

In the last years, the volume of information is growing faster than ever before, moving from small to huge, structured to unstructured datasets like text, image, audio and video. The purpose of processing the data is aimed to extract relevant information on trends, challenges and opportunities; all these studies with large volumes of data. The increase in the power of parallel computing enabled the use of Machine Learning (ML) techniques to take advantage of the processing capabilities offered by new architectures on large volumes of data. For this reason, it is necessary to find mechanisms that allow classify and organize them to facilitate to the users the extraction of the required information. The processing of these data requires the use of classification techniques that will be reviewed. This work analyzes different studies carried out on the use of ML for processing large volumes of data (Big Multimedia Data) and proposes a classification, using as criteria, the hardware infrastructures used in works of machine learning parallel approaches applied to large volumes of data.

KEYWORDS

Big Data, GPU, Hadoop, Machine Learning, MapReduce, Multimedia

1. INTRODUCTION

Machine Learning tries to imitate human being intelligence using machines. Machine learning algorithms use data of any kind to train the model. Depending on the problem, the data may be of the order of gigas. For this reason an optimized storage system for large volumes of data (Big Data) is indispensable.

In recent years there has been an accelerated growth in the volume of information available on the network. Likewise, several alternatives have appeared for processing these large volumes of data (Big Data) and their storage. These alternatives are related to both: structured (numerical and alphanumerical data) and unstructured (text, images and videos) data. In the first case, some sort of *Database System* is needed, and in the second a sophisticated *File System* has to be used. As an example of the first case we can mention Apache HBase[1], and in the second Hadoop Distributed File System (HDFS)[2]. The complexity of the data demands the creation of new architectures that optimize the computation time and the necessary resources to extract valuable knowledge from the data (Singh & Kaur, 2016).

The accelerated growth of information of various types available on the network, has generated the need to extract information and process it in an efficient way. Traditional techniques are oriented to process information in clusters. With the evolution of the graphic processor unit (GPU) it appeared

DOI: 10.4018/IJCVIP.2017070105

alternatives to take full advantage of the multiprocessing capacity of this type of architectures. The most common programming frameworks area NVidia and OpenCL (Demidov, Ahnert, Rupp, & Gottschiling, 2013). In 2008, Khronos Group introduced the OpenCL (Open Computing Language) standard, which was a model for parallel programming. Subsequently appeared its main competitor, NVidia CUDA (Computer Unified Device Architecture). CUDA devices increase the performance of a system due to the high degree of parallelism they are able to manage (Kirk & Hwu, 2010).

This document reviews platforms, languages, and many other features of the most popular machine learning frameworks and offers a classification for someone who wants to begin in this field. In addition, an exhaustive review of works using machine learning techniques to deal with Big Data is done, including its most relevant features.

The remainder of this document is organized as follows. Section 2 introduces the Big Data concept summarizing its characteristics; Section 3 describes the techniques of machine learning and the most popular platforms. Next, an overview about Big Multimedia Data Processing is presented. Section 4 presents a summary table with several classification criteria. Finally, in section 5 some conclusions and opportunities for future work are presented.

2. BIG DATA

Big Data is present in all areas and sectors worldwide. However, it's complexity exceeds the processing power of traditional tools, requiring high-performance computing platforms to exploit the full power of Big Data (Shim, 2013). These requirements have undoubtedly become a real challenge. Many studies focus on the search of methodologies that allow lowering computational costs with an increase in the relevance of extracted information. The need to extract useful knowledge has required researchers to apply different machine learning techniques, to compare the results obtained and to analyze them according to the characteristics of the large data volumes (volume, velocity, veracity and variety, the 4V's) (Mujeeb & Naidu, 2015).

The techniques used by Machine Learning (ML) are focused on minimizing the effects of noise from digital images, videos, hyperspectral data, among others, extracting useful information in various areas of knowledge, such as civil engineering (Rashidi, 2016), medicine (Athinarayanan, 2016), remote Sensing (Torralba, 2008).

With the various repositories of images that have been generated over the last years, many computer vision algorithms try to solve problems related to finding matches for existing local image features in Big Data, grouping the characteristics and labeling them (Muja, 2009).

Actually, there are several information repositories related to a wide range of areas, these datasets can be used to test the performance of some algorithms. For example:

- Face database: CMU-MultiPIE
  This dataset contains around 750.000 images of people over the span of five months. This dataset contains more than 305 GB of data.
- Classification with multiples classes, digit recognition: THE MNIST DATABASE
  This dataset contains data of handwritten digits, it has a training set of 60,000 examples, and a test set of 10000 examples. It can be found at http://yann.lecun.com/exdb/mnist/

3. MACHINE LEARNING FOR BIG MULTIMEDIA DATA

This section reviews Machine Learning and Big Multimedia Data processing platforms concepts. The first part presents a classification of the machine learning techniques to later summarize some platforms oriented to the implementation of the algorithms.

3.1. Machine Learning

The main goal of machine learning is to create systems that learn or extract knowledge from data and use that knowledge to predict future situations, states or trends (Landset, Khoshgoftaar, Richter, & Hasanin, 2015).

Machine Learning algorithms can be grouped as follows:

1. Supervised Learning
   - Classification algorithms

     A system is able to learn from a set of sample data (labeled data), from which a set of classification rules called model is built. These rules are used to predict a class or category for new information (Nguyen & Armitage, 2008). In this case the model predicts the value of a categorical variable.

     In short, a classification algorithm trains a model that can predict a categorical value.
   - Regression algorithms

     The goal of this type of algorithms is to predict a numerical label using unlabeled observations. For the observations, it is necessary to known the numerical value of the label (Guller, 2015).

     In short, a regression algorithm trains a model than can predict a numerical value.
2. Unsupervised Learning
   - Clustering algorithms

     It consists on the formation of groups (called clusters) of instances that share common characteristics without any prior knowledge (Nguyen & Armitage, 2008).
   - Recommendations algorithms

     It consists of predicting patterns of preferences and the use of those preferences to make recommendations to the users (Owen, Anil, Dunning, & Friedman, 2012).
   - Dimensionality Reduction Algorithms

     It consists on the reduction in the number of variables (attributes) of a dataset without affecting the predictive capacity of the model (Guller, 2015).

     Machine Learning techniques are currently used to extract relevant information from different types of data. Using these techniques provides, in many cases, better results in the process of sorting unstructured data (images, videos), especially if the information comes from multiple sources (Sarath, 2014).

     Scalability is an important aspect to consider in a learning method. Such capacity is defined as the ability of a system to adapt to the increasing demands in terms of information processing. To support this process onto large volumes of data, the platforms incorporate different forms of scaling (Singh & Reddy, 2015):
   - Horizontal scaling, involves the distribution of the process over several nodes (computers).
   - Vertical scaling, involves adding more resources to a single node (computer).

     Vertical scaling was associated with graphics processing, but since GPU became a general-purpose processing units, vertical scaling could be applied to any kind of problem.

3.1.1. Machine Learning Libraries

In this section, we present the description of several libraries that implement some of the algorithms included in the previous classification. The use of these libraries is proposed using as criteria the integration with new systems but not and end user tool criteria.

Apache Mahout[3] provides implementations for many of the most popular machine learning algorithms. It contains implementation of clustering algorithms, classification, and many others. To enable clustering, it uses Apache Hadoop (Aridhi & Mephu, 2016).

Some algorithms included in Apache Mahout are:

- Classification: Naïve Bayes, Hidden Markov Models, Logistic Regression, Random Forest.
- Clustering: K-Means, Canopy, Fuzzy k-Means, Streaming KMeans, Spectral Clustering.
- Recommendations: user based, Item-to-item, item-to-ALS (Alternating Least Squares).
- Dimensionality reduction: Singular Value Decomposition, Stochastic SVD.

MLlib[4] is part of the Apache Spark project that implements several machine learning algorithms. Among the groups of algorithms that it implements we can find classification, regression, clustering and dimensionality reduction algorithms (Meng, y otros, 2016).

Some algorithms included in MLlib are:

- Classification: logistic regression, naïve Bayes, decision trees, random forest, linear SVMs, and others.
- Clustering: K-Means, Gaussian mixtures (GMMs), Power Iteration Clustering (PIC), Latent Dirichlet Allocation (LDA), Bisecting k-means, Streaming k-means.
- Regression: Generalized Linear Regression (GLR).
- Recommendations: Alternating Least Squares (ALS).
- Dimensionality reduction: Singular Value Decomposition (SVD), Principal Component Analysis (PCA).

FlinkML[5] is the Flink machine learning library. Its goal is to provide scalable machine learning algorithms. It provides tools to help the design of machine learning systems (Aridhi & Mephu, 2016).

Some algorithms included in FlinkML are:

- Classification: Multiple linear regression, optimization framework, SVM, and others.
- Clustering: K-Nearest neighbors join.
- Recommendations: Alternating Least Squares (ALS)

Table 1 describes the libraries presented above. For each one, the supported programming language is detailed:

As shown above, the most used languages are Java and Scala, which shows a tendency when implementing systems that integrate machine learning techniques.

Table 2 describes each library and the support for scaling when working with large volumes of data:

In the table, we can see that the scaling techniques are combined with others to obtain platforms with better performance.

Table 1. Machine learning tool by language

Tool	Java	Scala	Python	R
Apache Mahout	x	x		
Spark MLLib	x	x	x	x
FlinkML	x	x		

Table 2. Machine learning tool scaling

Tool	H+V	Multicore	GPU	Cluster
Apache Mahout	x	x		x
Spark MLLib	x	x	x	x
FlinkML	x	x		x

3.2. Platforms for Big Multimedia Data processing

Big Multimedia Data is defined as a large and complex collection of data (images, audio and video), which are difficult to process by a relational database system. The typical size of the data, in this type of problems, is of the order of tera or peta bytes and they are in constant growth (Jiang, Chen, Qiao, Weng, & Li, 2015).

There are many platforms to work with Big Data, among the most popular we can mention Hadoop, Spark (two open source projects from the Apache Foundation) and MapReduce.

MapReduce (Dean & Ghemawat, 2004) is a programming model oriented to the process of large volumes of data (Hashem, y otros, 2016). Problems addressed using MapReduce should be problems that can be separated into small tasks to be processed in parallel (Kiran & Ravi Prakash, 2013). Many different implementations of the original MapReduce framework are possible (Dean & Ghemawat, 2004), for example this programming model is adopted by several specific implementations (platforms) like Apache Hadoop and Apache Spark.

Apache Hadoop[6] is an open source project sponsored by the *Apache Software Fundation*. It allows distributed processing of large volumes of data in a cluster (Jackson, Vijayakumar, Quadir, & Bharathi, 2015). It consists of three fundamental components (Holmes, 2015): HDFS (storage component), YARN (resource planning component) and MapReduce.

Apache Spark[7] is a cluster computing system based on the MapReduce concept. Spark supports interactive computing and its main objective is to provide high performance while maintaining resources and calculations in memory (Landset, Khoshgoftaar, Richter, & Hasanin, 2015).

H_2O[8] is an open source framework that provides libraries for parallel processing, information analysis and machine learning along with data processing and evaluation tools (Landset, Khoshgoftaar, Richter, & Hasanin, 2015).

Apache Storm[9] is an open source system for real time distributed computing. An application created with Storm is designed as a directed acyclic graph (DAG) *topology*.

Apache Flink[10] is an open source platform for batch and stream processing

Table 3 compares the platform with the type of processing supported and the storage system used.

As can be seen, all the platforms support all varieties of storage. However, in large implementations the local storage (or pseudo-cluster) is not an option to consider.

Table 3. Platform scaling

Tool	Scaling				Storage		
	HV	Multicore	GPU	Cluster	IM[11]	Local	DFS[12]
Apache Hadoop	x	x		x		x	x
Apache Spark	x	x	x	x	x	x	x
H2O	x	x	x[13]	x	x	x	x
Apache Storm	x	x		x	x	x	x
Apache Flink	x	x		x	x	x	x

4. MACHINE LEARNING FOR BIG MULTIMEDIA DATA REVIEW

This section proposes a classification of research works that use Big Multimedia Data platforms to apply machine learning techniques based on three criteria: language (programming language supported), scaling (scalability) and storage (supported storage type).

The combination of these three factors allows the proper selection of a platform to integrate machine learning techniques with Big Data. Scaling is related to the type of storage. In the case of using horizontal scaling, a distributed file system must be used.

Table 4 shows the classification based on the above-mentioned criteria.

Table 4. Language-Scaling-Storage LSS Classification

Article	Language	Scaling	Storage
Al-Jarrah et al., 2015		V	DFS, IM
Aridhi & Mephu, 2016	Scala, Java	Cluster	DFS, IM
Armbrust et al., 2015	Scala, Java, Python	Cluster	cal, DFS, IM
Bertolucci et al., 2016	Java	Cores, Cluster	DFS, IM
Borthakur, 2008	Java	Cluster	Local, DFS
Castillo et al., 2010		Cluster	
Catanzaro et al., 2008		Cluster, GPU	DFS, IM
Crawford et al., 2015	Scala, Java, Python		Loca,IM
Nagina, 2016	Java	Cluster	Local, DFS
Fan & Bifet, 2013	Java	Cluser, GPU	DFS
Gandomi & Haider, 2015		Vertical	
Ghemawat et al., 2003		Cluster	DFS
Hafez et al., 2016	Scala, Java, Python	Cluster	DFS, IM
Hashem et al., 2016	Java	V, Cluster, GPU	DFS, IM
He et al., 2014		V, Cluster	DFS, IM
Hodge et al., 2016	Scala, Java	Cluster	DFS
Issa & Figueira, 2012	Java, Python	H, Cluster	DFS, IM
Jackson et al., 2015	Java, Python	Cluster	DFS
Jain & Bhatnagar, 2016	Java	Cluster	DFS
Jiang et al., 2015	Java, Python	Cluster, GPU	DFS, IM
Kacfah Emani et al., 2015	Scala, Java	Cluster	DFS, IM
Kiran & Ravi Prakash, 2013	Java	Cluster	DFS, IM
Kraska et al., 2013	Scala	Cluster	DFS, IM
Landset et al., 2015	Scala, Java		DFS, IM
Meng et al, 2016	Scala, Java	Cluster	DFS
Modha & Spangler, 2003		Cluster	DFS
Naimur Rahman et al., 2016	Java	Cluster	DFS
Namiot, 2015	Scala, Java	Cluster	DFS
Norman et al., 2015		Cluster	DFS
Pääkkönen, 2016	Scala, Java, Python	Cluster	DFS
Ramírez-Gallego et al., 2015	Scala	Core, Cluster	DFS
Saecker & Markl, 2013		V, Cluster, GPU	DFS
Salloum et al., 2016	Scala, Java	Cluster, Paralell	DFS
Saraladevi et al., 2015	Java	Cluster	DFS
Seminario & Wilson, 2012	Java	Cluster	cal, IM, DFS
Singh & Reddy, 2015	Scala, Java		cal, IM, DFS
Singh & Kaur, 2016	Java	H, V, Cluster	DFS
Walunj & Sadafale, 2013	Java		Local, DFS
Zaharia et al., 2010	Scala, Java	Cluster	DFS

4.1. Language

The supported programming language is an important aspect when selecting a platform. Taking into account that this work is oriented to platforms that are not end user (tools like Weka[14], RapidMiner[15], etc. are not considered), the following programming languages are considered: Java, Scala, Python, R.

The most common languages in this type of implementations are Java and Scala. However over time appeared layers of software that abstract access to certain technologies. With these news technologies is possible to use the platforms described in this document with different programming languages.

4.2. Scaling

Scaling focuses on the possibility of including more process nodes (horizontal scaling) or include parallel processing within the same node (vertical scaling) by using of graphics cards. This document consider horizontal, vertical, cluster and GPU scaling.

As can be seen in, and in the above tables, all the platforms scale horizontal by using a DFS that in most cases corresponds to Hadoop. However, platforms that work with data in memory are becoming more popular (like Spark).

4.3. Storage

Depending on the amount of information and the strategy used to process the information, it is possible to decide the type of storage: local, DFS, in-memory. Each type involves the implementation of hardware infrastructure to be used. For example, in the case of a cluster implementation, it is necessary to have adequate equipment that will constitute the cluster nodes.

For proof of concepts it is acceptable to use a pseudo-cluster that simulates a cluster environment on a single machine.

5. CONCLUSION

In this document, a brief description on the machine learning techniques was carried out, orienting them to the processing of large volumes of data. Then, some of the platforms that implement the algorithms described for the processing of multimedia data were analyzed. Most of the reviewed articles use techniques, languages and scaling described in this document.

In general, we can conclude that all these techniques are scalable in one way or another. It is possible to start from a local architecture (or pseudo-cluster) for a proof of concept test and progressively scale to more complex architectures like a cluster, evidencing the need for distributed storage (DFS). The programming language should not be a factor when selecting a platform, since nowadays there are interfaces that allow the use of the mentioned platforms in a growing variety of languages.

General purpose datasets are a good option for testing some aspects of each algorithm and platform, but the final decision depends on the type of data related with the problem.

As future work, a comparison of efficiency and scaling of different platforms in multimedia data could be done, comparisons related with: medical images, facial and pattern recognition.

Finally, streaming processing could be incorporated into the study, which is a fundamental part of some of the platforms described in this document.

REFERENCES

Al-Jarrah, O. Y., Yoo, P. D., Muhaidat, S., Karagiannidis, G. K., & Taha, K. (2015). Efficient Machine Learning for Big Data: A Review. *Big Data Research*, *2*(3), 87–93. doi:10.1016/j.bdr.2015.04.001

Aridhi, S., & Mephu, E. (2016). Big Graph Mining: Frameworks and Techniques. *Big Data Research*, *6*, 1–10. doi:10.1016/j.bdr.2016.07.002

Armbrust, M., Ghodsi, A., Zaharia, M., Xin, R. S., Lian, C., Huai, Y., & Franklin, M. J. et al. (2015). Spark SQL: Relational Data Processing in Spark Michael. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data - SIGMOD '15* (pp. 1383-1394). doi:10.1145/2723372.2742797

Athinarayanan, S. &. (2016). Classification of cervical cancer cells in PAP smear screening test. *ICTACT Journal on Image and Video Processing*, *6*(4), 1234-1238.

Bertolucci, M., Carlini, E., Dazzi, P., Lulli, A., & Ricci, L. (2016). Static and dynamic big data partitioning on apache spark. *Advances in Parallel Computing*, *27*, 489–498.

Borthakur, D. (2008). HDFS architecture guide. *Hadoop Apache Project*.

Castillo, S. J., Fernández, J. R., & Sotos, L. G. (2010). Algorithms of Machine Learning for K-Clustering. *Trends in Practical Applications of Agents and Multiagent Systems*, *71*, 443-452.

Catanzaro, B., Catanzaro, B., Keutzer, K., & Keutzer, K. (2008). Fast Support Vector Machine Training and Classification on Graphics Processors. In Proceedings of the 25th international conference on *Machine Learning* (pp. 104–111).

Crawford, M., Khoshgoftaar, T. M., Prusa, J. D., Richter, A. N., & Al Najada, H. (2015). Survey of review spam detection using machine learning techniques. *Journal of Big Data*, *2*(1), 23. doi:10.1186/s40537-015-0029-9

Dean, J., & Ghemawat, S. (2004). MapReduce: Simplied Data Processing on Large Clusters. In *Proceedings of 6th Symposium on Operating Systems Design and Implementation* (pp. 137-149).

Demidov, D., Ahnert, K., Rupp, K., & Gottschiling, P. (2013). ViennaCL---Linear Algebra Library for Multi- and Many-Core Architectures. *SIAM Journal on Scientific Computing*, 38(5), 412–439.

Fan, W., & Bifet, A. (2013). Mining Big Data: Current Status, and Forecast to the Future. *ACM SIGKDD Explorations Newsletter*, *14*(2), 1–5. doi:10.1145/2481244.2481246

Gandomi, A., & Haider, M. (2015). Beyond the hype: Big data concepts, methods, and analytics. *International Journal of Information Management*, *35*(2), 137–144. doi:10.1016/j.ijinfomgt.2014.10.007

Ghemawat, S., Gobioff, H., & Leung, S.-t. (2003). Google File System.

Guller, M. (2015). *Big Data Analytics with Spark.*

Hafez, M. M., Shehab, M. E., El Fakharany, E., & Hegazy, A. G. (2016). Effective Selection of Machine Learning Algorithms for Big Data Analytics Using Apache Spark. In *Proceedings of the International Conference on Intelligent Systems.*

Hashem, I. A., Anuar, N. B., Gani, A., Yaqoob, I., Xia, F., & Khan, S. U. (2016). MapReduce: Review and open challenges. *Scientometrics*, *109*(1), 1–34. doi:10.1007/s11192-016-1945-y

He, Q., Li, N., Luo, W. J., & Shi, Z. Z. (2014). A survey of machine learning for big data processing. *Moshi Shibie yu Rengong Zhineng [Pattern Recognition and Artificial Intelligence]*, *27*, 327-336.

Hodge, V. J., Keefe, S. O., & Austin, J. (2016). Hadoop neural network for parallel and distributed feature selection. *Neural Networks*, *78*, 24–35. doi:10.1016/j.neunet.2015.08.011 PMID:26403824

Holmes, A. (2015). *Hadoop in Practice* (S. Edi, Ed.). Manning.

Issa, J., & Figueira, S. (2012). Hadoop and memcached: Performance and power characterization and analysis. *Journal of Cloud Computing: Advances, Systems and Applications*, *1(1)*, 10.

Jackson, J. C., Vijayakumar, V., Quadir, M. A., & Bharathi, C. (2015). Survey on programming models and environments for cluster, cloud, and grid computing that defends big data. *Procedia Computer Science*, *50*, 517–523. doi:10.1016/j.procs.2015.04.025

Jain, A., & Bhatnagar, V. (2016). Crime Data Analysis Using Pig with Hadoop. *Physics Procedia*, *78*, 571–578.

Jiang, H., Chen, Y., Qiao, Z., Weng, T. H., & Li, K. C. (2015). Scaling up MapReduce-based Big Data Processing on Multi-GPU systems. *Cluster Computing*, *18*(1), 369–383. doi:10.1007/s10586-014-0400-1

Kacfah Emani, C., Cullot, N., & Nicolle, C. (2015). Understandable Big Data: A survey. *Computer Science Review*, *17*, 70–81. doi:10.1016/j.cosrev.2015.05.002

Kiran, M. a., & Ravi Prakash, G. (2013). Verification and Validation of MapReduce Program Model for Parallel Support Vector Machine Algorithm on Hadoop Cluster. In *Proceedings of the International Conference on Advanced Computing and Communication Systems (ICACCS)* (Vol. 4, pp. 317-325). doi:10.1109/ICACCS.2013.6938728

Kirk, D., & Hwu, W.-M. W. (2010). *Programming Massively Parallel Processors: A Hands-on Approach.*

Kraska, T., Talwalkar, A., Duchi, J., Griffith, R., Franklin, M., & Jordan, M. (2013). MLbase: A Distributed Machine-learning System. In *Proceedings of the 6th Biennial Conference on Innovative Data Systems Research (CIDR'13).*

Landset, S., Khoshgoftaar, T. M., Richter, A. N., & Hasanin, T. (2015). A survey of open source tools for machine learning with big data in the Hadoop ecosystem. *Journal of Big Data*, *2*(1), 24. doi:10.1186/s40537-015-0032-1

Meng, X., Bradley, J., Street, S., Francisco, S., Sparks, E., Berkeley, U. C., & Hall, S. et al. (2016). MLlib. *Machine Learning in Apache Spark.*, *17*, 1–7.

Mishra, N. K. (2016). An overview of melanoma detection in dermoscopy images using image processing and machine learning.

Modha, D. S., & Spangler, W. S. (2003). Feature Weighting in k -Means Clustering. *Machine Learning*, *52*(3), 217–237. doi:10.1023/A:1024016609528

Muja, M. &. (2009). Fast approximate nearest neighbors with automatic algorithm configuration. *VISAPP*, *1*(2), 331-340.

Mujeeb, S., & Naidu, L. (2015). A Relative Study on Big Data Applications and Techniques. *International Journal of Engineering and Innovative Technology*, *4*(10), 133-138.

Nagina, D. S. (2016). Scheduling algorithms in big data: a survey. *International Journal Of Engineering And Computer Science*, *5*, 17737–17743.

Naimur Rahman, M., Esmailpour, A., & Zhao, J. (2016). Machine learning with big data an efficient electricity generation forecasting system. *Big Data Research*, *5*, 9–15. doi:10.1016/j.bdr.2016.02.002

Namiot, D. (2015). On big data stream processing. *International Journal of Open Information Technologies*, *3(8)*, 48–51.

Nguyen, T. T., & Armitage, G. (2008). A survey of techniques for internet traffic classification using machine learning. *IEEE Communications Surveys & Tutorials*, *10*(4), 56-76.

Norman, S., Martin, R., & Franczyk, B. (2015). Evaluating New Approaches of Big Data Analytics Frameworks. In Proceedings of the International Conference on Business Information Systems, *LNBIP* (Vol. *208*, pp. 28–37). doi:10.1007/978-3-319-19027-3_3

Owen, S., Anil, R., Dunning, T., & Friedman, E. (2012). *Mahout in Action*. Manning.

Pääkkönen, P. (2016). Feasibility analysis of AsterixDB and Spark streaming with Cassandra for stream-based processing. *Journal of Big Data*, *3*(1), 6. doi:10.1186/s40537-016-0041-8

Ramírez-Gallego, S., Mouriño-Talín, H., Martínez-Rego, D., Bolón-Canedo, V., Benitez, J. M., Alonso-Betanzos, A., & Herrera, F. (2015). Un Framework de Selección de Características basado en la Teoría de la Información para Big Data sobre Apache Spark.

Rashidi, A. S. (2016). An analogy between various machine-learning techniques for detecting construction materials in digital images. *KSCE Journal of Civil Engineering, 20*(4), 1178-1188.

Saecker, M., & Markl, V. (2013). Big data analytics on modern hardware architectures: A technology survey. In *Business Intelligence, LNBIP* (Vol. 138, pp. 125-149.

Salloum, S., Dautov, R., Chen, X., Peng, P. X., & Huang, J. Z. (2016). Big data analytics on Apache Spark. *International Journal of Data Science and Analytics, 1*(3-4), 145–164. doi:10.1007/s41060-016-0027-9

Saraladevi, B., Pazhaniraja, N., Paul, P. V., Basha, M. S., & Dhavachelvan, P. (2015). Big data and Hadoop-A study in security perspective. *Procedia Computer Science, 50*, 596–601. doi:10.1016/j.procs.2015.04.091

Sarath, T. N. (2014). A Study on Hyperspectral Remote Sensing. In *Proceedings of the International Conference on Information and Communication Technologies* (pp. 5-8).

Seminario, C. E., & Wilson, D. C. (2012). Case study evaluation of mahout as a recommender platform. In *Proceedings of the CEUR Workshop* (pp. 45-50).

Shim, K. (2013). MapReduce algorithms for big data analysis. *LNCS, 7813*, 44–48. doi:10.1007/978-3-642-37134-9_3

Singh, D., & Reddy, C. K. (2015). A survey on platforms for big data analytics. *Journal of Big Data, 2*(1), 8. doi:10.1186/s40537-014-0008-6 PMID:26191487

Singh, R., & Kaur, P. J. (2016). Analyzing performance of Apache Tez and MapReduce with hadoop multinode cluster on Amazon cloud. *Journal of Big Data, 3*(1), 19. doi:10.1186/s40537-016-0051-6

Torralba, A. F., Fergus, R., & Freeman, W. T. (2008). 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 30*(11), 1958–1970. doi:10.1109/TPAMI.2008.128 PMID:18787244

Walunj, S. G., & Sadafale, K. (2013). An Online Recommendation System for E-commerce Based on Apache Mahout Framework. In *Proceedings of the 2013 Annual Conference on Computers and People Research*, 153-158. doi:10.1145/2487294.2487328

Zaharia, M., Chowdhury, M., Franklin, M. J., Shenker, S., & Stoica, I. (2010). Spark: Cluster Computing with Working Sets. In *Proceedings of the 2nd USENIX conference on Hot topics in Cloud Computing HotCloud'10*.

ENDNOTES

1 https://hbase.apache.org/
2 http://hadoop.apache.org/
3 http://mahout.apache.org
4 http://spark.apache.org/mllib/
5 https://ci.apache.org/projects/flink/flink-docs-release-1.2/dev/libs/ml/index.html
6 http://hadoop.apache.org
7 http://spark.apache.org
8 http://www.h2o.ai/h2o
9 http://storm.apache.org
10 https://flink.apache.org
11 In-memory
12 Distributed File System
13 Through Deep Water (http://www.h2o.ai/deep-water/) in beta state
14 http://www.cs.waikato.ac.nz/ml/weka
15 https://rapidminer.com

Jaime Salvador received his Master's Degree in Artificial Intelligence from the Universidad de Sevilla in 2014. He is currently a Professor at Facultad de Ingeniería, Ciencias Físicas y Matemática of the Universidad Central del Ecuador. His main research interests include parallel computing in GPU, Machine Learning.

Zoila Ruiz received her Master's Degree in Artificial Intelligence from the Universidad de Sevilla in 2014. She is currently a Professor at Facultad de Ingeniería, Ciencias Físicas y Matemática of the Universidad Central del Ecuador. His main research interests include Machine Learning, Agent based systems, and Geographic Information Systems.

Jose Garcia-Rodriguez received his PhD degree, with specialization in Computer Vision and Neural Networks, from the University of Alicante (Spain). He is currently Associate Professor at the Department of Computer Technology of the University of Alicante. His research areas of interest include: computer vision, computational intelligence, machine learning, pattern recognition, robotics, man-machine interfaces, ambient intelligence, computational chemistry, and parallel and multicore architectures. He has authored +100 publications in journals and top conferences and revised papers for several journals like Journal of Machine Learning Research, Computational intelligence, Neurocomputing, Neural Networks, Applied Softcomputing, Image Vision and Computing, Journal of Computer Mathematics, IET on Image Processing, SPIE Optical Engineering and many others, chairing sessions in the last decade for WCCI/IJCNN and participating in program committees of several conferences including IJCNN, ICRA, ICANN, IWANN, IWINAC KES, ICDP and many others. He is also member of European Networks of Excellence and COST actions like Eucog, HIPEAC, AAPELE or I&VL and director or the GPU Research Center at University of Alicante and PhD program in Computer Science.