CLASSIFICATION ALGORITHMS FOR BIG DATA ANALYSIS, A MAP REDUCE APPROACH

V. A. Ayma [a, *], R. S. Ferreira [a], P. Happ [a], D. Oliveira [a], R. Feitosa [a, b], G. Costa [a], A. Plaza [c], P. Gamba [d]

[a] Dept. of Electrical Engineering, Pontifical Catholic University of Rio de Janeiro, Brazil – (vaaymaq, rsilva, patrick, raul, gilson)@ele.puc-rio.br
[b] Dept. of Computer and Systems, Rio de Janeiro State University, Brazil
[c] Dept. of Technology of Computers and Communications, University of Extremadura, Spain - aplaza@unex.es
[d] Dept. of Electronics, University of Pavia, Italy - paolo.gamba@unipv.it

KEY WORDS: Big Data, MapReduce Framework, Hadoop, Classification Algorithms, Cloud Computing

ABSTRACT:

Since many years ago, the scientific community is concerned about how to increase the accuracy of different classification methods, and major achievements have been made so far. Besides this issue, the increasing amount of data that is being generated every day by remote sensors raises more challenges to be overcome. In this work, a tool within the scope of *InterIMAGE Cloud Platform* (*ICP*), which is an open-source, distributed framework for automatic image interpretation, is presented. The tool, named *ICP: Data Mining Package*, is able to perform supervised classification procedures on huge amounts of data, usually referred as *big data*, on a distributed infrastructure using Hadoop MapReduce. The tool has four classification algorithms implemented, taken from WEKA's machine learning library, namely: Decision Trees, Naïve Bayes, Random Forest and Support Vector Machines (SVM). The results of an experimental analysis using a SVM classifier on data sets of different sizes for different cluster configurations demonstrates the potential of the tool, as well as aspects that affect its performance.

1. INTRODUCTION

The amount of data generated in all fields of science is increasing extremely fast (Sagiroglu et al., 2013) (Zaslavsky et al., 2012) (Suthaharan, 2014) (Kishor, 2013). MapReduce frameworks (Dean et al., 2004), such as Hadoop (Apache Hadoop, 2014), are becoming a common and reliable choice to tackle the so called *big data* challenge.

Due to its nature and complexity, the analysis of *big data* raises new issues and challenges (Li et al., 2014) (Suthaharan, 2014). Although many machine learning approaches have been proposed so far to analyse small to medium size data sets, in a supervised or unsupervised way, just few of them have been properly adapted to handle large data sets (Yadav et al., 2013) (Dhillon et al., 2014) (Pakize et al., 2014). An overview of some data mining approaches for very large data sets can be found in (He et al., 2010) (Bekkerman et al., 2012) (Nandakumar et al., 2014).

There are two main steps in the supervised classification process. The first is the training step where the classification model is built. The second is the classification itself, which applies the trained model to assign unknown data to one out of a given set of class labels. Although the training step is the one that draws more scientific attention (Liu et al., 2013) (Dai et al., 2014) (Kiran et al., 2013) (Han et al., 2013), it usually relies on a small representative data set that does not represent an issue for *big data* applications. Thus, the *big data* challenge affects mostly the classification step.

This work introduces *the ICP: Data Mining Package*, an open-source, MapReduce-based tool for the supervised classification of large amounts of data. The remaining of the paper is organized as follows: Section 2 presents a brief overview of Hadoop; the tool is presented in Section 3; a case study is presented in Section 4 and, finally, the conclusions are discussed in Section 5.

2. HADOOP OVERVIEW

Apache Hadoop is an open-source implementation of the MapReduce framework, proposed by Google (Intel IT Center, 2012). It allows the distributed processing of datasets in the order of petabytes across hundreds or thousands of commodity computers connected to a network (Kiran et al., 2013). As presented in (Dean et al., 2004), it has been commonly used to run parallel applications for big data processing and analysis (Pakize et al., 2014) (Liu et al., 2013). The next two sections present Hadoop's two main components: HDFS and MapReduce.

2.1 Hadoop Distributed File System

The Hadoop Distributed File System (HDFS) is the storage component of Hadoop. It is designed to reliably store very large data sets on clusters, and to stream those data at high throughput to user applications (Shvachko et al., 2010). HDFS stores file system metadata and application data separately. By default, it stores three independent copies of each data block (*replication*) to ensure reliability, availability and performance (Kiran et al., 2013).

[*] Corresponding author

doi:10.5194/isprsarchives-XL-3-W2-17-2015

2.2 Hadoop MapReduce

Haddop MapReduce is a parallel programming technique for distributed processing, implemented on top of HDFS (Grolinger et al., 2014). The Hadoop MapReduce engine consists of a *JobTracker* and several *TaskTrackers*. When a MapReduce job is executed, the JobTracker splits it into smaller tasks (map and reduce) handled by the TaskTrackers. In the Map step, the master node takes the input, divides it into smaller sub-problems and distributes them to worker nodes. Each worker node processes a sub-problem and writes its results as key/value pairs. In the Reduce step, the values with the same key are grouped and processed by the same machine to form the final output (Kiran et al., 2013).

2.3 Pig

Pig is a framework for executing data flows in parallel on Hadoop. It has two components: a language and an engine. *Pig's* language, called Pig Latin, makes it easier for non-technical users to interact with MapReduce by providing a high-level language that is also extensible (Apache PIG, 2014) (Olston et al., 2008). Pig Latin can be extended through the use of User Defined Functions (*UDFs*), which can be written in Java, Jython, Python, JavaScript, Ruby and Groovy. Through *UDFs*, users can create custom functions that meet their specific needs. Pig's engine takes a Pig Latin script and compiles it automatically in MapReduce jobs.

3. ICP: DATA MINING PACKAGE

InterIMAGE Cloud Platform (*ICP*) is an open source, distributed framework for automatic interpretation of remote sensing and medical image data built on top of Hadoop (Ferreira et al., 2014). *ICP: Data Mining Package* is one of the tools within the scope of this framework, which is an open-source software tool implemented in Java and freely available in http://www.lvc.ele.puc-rio.br/wp/?p=1831. Up to now, it embodies four classification algorithms taken from the *WEKA* (Machine Learning Group at the University Waikato, 2014) java library: Naïve Bayes Classifier, Decision Trees, Random Forest and Support Vector Machines (SVM).

The parallel procedure works as follows. The data to be classified, henceforth called *big data* set is stored on HDFS. The training set is stored on an auxiliary storage system. When the execution starts, each HDFS block is processed by a different map task. The map task, firstly, reads the training data set and trains the classifier. After that, the trained classification model is used to classify the *big data set*. The multiple executions of the training step (for each map) should not impact the computational performance substantially because the amount of training data is small compared to the *big data set,* which accounts for most of the processing time.

A brief example of a *Pig Latin* script is presented in Table 1. In this script, a SVM classification process is performed using a given training and testing set, and saving the result in a defined output file.

Pig Latin script that executes a SVM classification

```
REGISTER …/pathTO/weka.jar;
REGISTER …/pathTO/interimage-pig-datamining.jar;
DEFINE II_SVMClassifier
br.puc_rio.ele.lvc.interimage.datamining.udf.SVMClassifier
('…/pathTo/trainSet.csv', 'configurationOptions');

dataTest = LOAD '…/pathTo/testSet.csv' USING
org.apache.pig.piggybank.storage.CSVExcelStorage(',',
'YES_MULTILINE', 'NOCHANGE',
'SKIP_INPUT_HEADER') AS (Att_1:float, …, Att_n:float);

classes = FOREACH dataTest GENERATE
II_SVMClassifier(Att_1, …, Att_n) AS csfrOutcome;

STORE classes INTO '…/pathTO/output.csv' USING
org.apache.pig.piggybank.storage.CSVExcelStorage(',',
'YES_MULTILINE', 'NOCHANGE');
```

Table 1. Pig Latin script that perform a SVM classification.

4. EXPERIMENTS AND RESULTS

This section reports some experiments conducted upon *ICP: Data Mining Package*. These experiments were carried out on the Mortar platform (Mortar Data, 2014) (Amazon Web Services, 2014), a cloud-computing, open-source framework for organizing, developing, testing, and deploying big data processing applications based on Hadoop. This platform relies on Amazon Elastic MapReduce (Amazon EMR, 2014), which uses the Hadoop framework to distribute the data and processing across a resizable Amazon Elastic Compute Cloud cluster (Amazon EC2, 2014), and on Amazon Simple Storage Service (Amazon S3, 2014).

On Mortar one can work directly with *Pig* on Hadoop and configure the number of cluster nodes on which the *Pig Latin* script will be executed in a simple and flexible way. The next sections will present the datasets used in this work.

4.1 Urban Hyperspectral Data Set

The tests reported henceforward were performed on Pavia hyperspectral data set (Hypercomp Research Group, 2014). It consists of a hyperspectral image collected by the ROSIS optical sensor over the University of Pavia, Italy. The image contains 610×340 pixels at 1.3 meters per pixel resolution over 103 spectral bands (from 0.43 to 0.86 μm).

The data set has nine ground truth classes of interest, comprising urban, soil and vegetation classes. In our experiments 3921 pixels were selected for training and 42776 pixels for testing, as shown in Figure 1(b) and Figure 1(c) respectively; and a Pavia hyperspectral false color composition image is presented in Figure 1(a). The classes' distribution within each data set is presented in Table 2.

The size of the Pavia hyperspectral ground truth is approximately 20Mb. Synthetic data sets were built from it, with 100, 200 and 500 times the original data set size, yielding data files with around 2Gb, 4Gb and 10Gb respectively. Only these three datasets were considered in the experiments since the original dataset is too small for Hadoop.

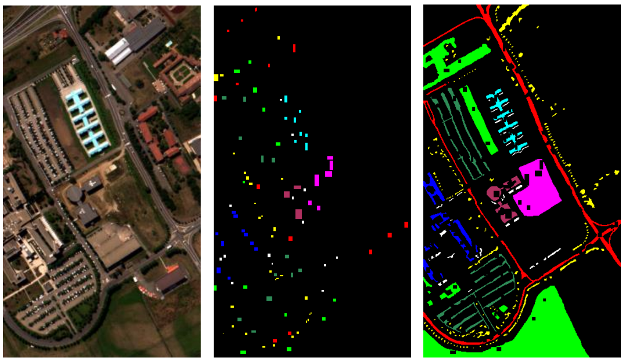
(a) (b) (c)

Figure 1. (a) Pavia hyperspectral image (false color composition, R-band 45,G-band 27, B-band 11); (b) Training data set; (c) Test data set.

Classes	Training Set Points	Testing Set Points	Color in Figure 1(c)
Asphalt	548	6304	🟥
Meadows	540	18146	🟩
Gravel	392	1815	🟦
Trees	524	2912	🟨
Metal Sheets	265	1113	🟦
Bare Soil	532	4572	🟪
Bitumen	375	981	🟫
Self-blocking Bricks	514	3364	🟩
Shadow	231	795	⬜

Table 2. Classes considered in the experimental analysis.

4.2 Experimental Results

The SVM classification algorithm was used to evaluate the tool. WEKA uses the Jhon Platts sequential minimal optimization algorithm for training the SVM (Platt, 1998). In the experiences, a multi-class pairwise (one versus one) SVM classification with a polynomial function kernel was performed, with a complexity parameter C = 1.0 and exponent value γ = 1.0 over a 5-fold cross validation procedure.

The SVM had as inputs, the first nine principal components computed from the 103 bands of the Pavia hyperspectral image. Figure 2 shows the outcome and the overall accuracy. The classification algorithm was applied on the 2GB, 4GB and 10GB data sets, in a local mode configuration (used as baseline) and in clusters with 10, 20 and 50 nodes on the Mortar platform. Each node in the cluster had 4 64-bit virtual cores, 15GB of RAM and a high-performance network.

Table 3 presents the execution times for each test data set running on the local mode and on the 10-, 20- and 50-node cluster configurations. For the local mode, the observed execution time grows almost linearly with the size of the input data. The results also show that the execution times drop consistently from the local mode to the other configurations.

Data Set	Execution Time (seconds)			
	1 Node	10 Nodes	20 Nodes	50 Nodes
2 GB	968	121	112	97
4 GB	2040	146	122	111
10 GB	4827	285	185	126

Table 3. Execution time for each configuration.

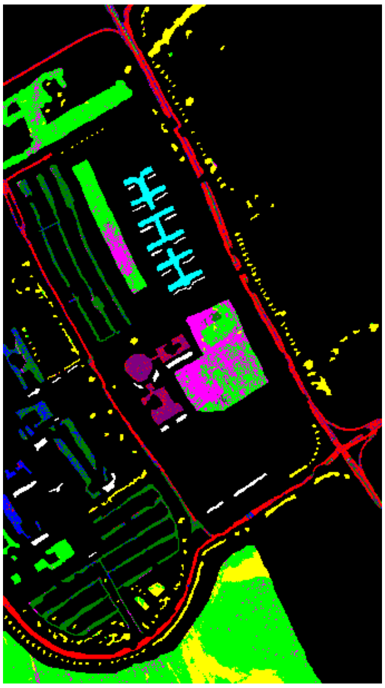

Figure 2. SVM classification outcome image on the Pavia Hypersectral data set (Overal Accuracy: 78,26%).

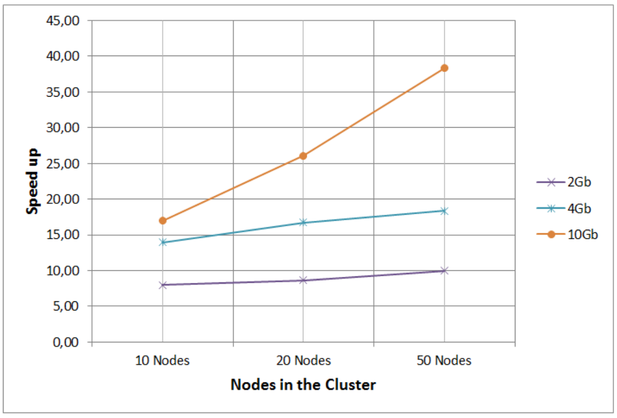

Figure 3. Speedups for each data set.

Figure 3 shows the speedup achieved by each cluster configuration for each image. It can be seen that, as the size of the data set increases, each cluster configuration achieves better speedups. This is because, for larger data sets, Hadoop can profit more from the available cores producing a higher parallelization. This also explains why larger data sets show higher speedup gains from one cluster configuration to the other. For the 4GB image, the speedups were 13.97, 16.72 and 18.38, respectively. Finally, for the 10GB image, the speedups were 16.94, 26.09 and 38.31.

On the other hand, as the cluster size increases, the classification for smaller data sets might underutilize the cluster resources. This result indicates that there is a threshold above which increasing the number of nodes does not produce substantial performance gains. For the 2GB image, for example, the speedups were 8.0, 8.64 and 9.98, for the 10-, 20- and 50-node clusters, respectively.

5. CONCLUSIONS

In this paper, *ICP: Data Mining Package* is presented, a tool able to perform classification processes on huge amounts of data, exploiting the benefits of working on clusters with the Hadoop framework.

An experimental analysis indicated that the speedup achieved by the tool increases with the amount of data being processed. Additionally, the results showed that increasing the number of nodes in the cluster does not necessarily provide a corresponding reduction of execution times. Thus, the proper cluster configuration depends not only on the operations to be executed but also on the amount of input data; there must be a balance between the amount of data to be processed and the number of nodes to be used to achieve the best performance.

ACKNOWLEDGEMENTS

The authors acknowledge the support provided by CNPq (Conselho Nacional de Desenvolvimento e Pesquisa), CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) and FP7 (Seventh Framework Programme) in the scope of the TOLOMEO Project.

REFERENCES

Amazon Web Services, 2014. *Amazon EC2*. http://aws.amazon.com/ec2/ (3 Nov. 2014).

Amazon Web Services, 2014. *Amazon EMR*. http://aws.amazon.com/elasticmapreduce/ (3 Nov. 2014).

Amazon Web Services, 2014. *Amazon S3*. http://aws.amazon.com/s3/ (3 Nov. 2014).

Amazon Web Services, 2014. *AWS Case Study: Mortar Data*. http://aws.amazon.com/solutions/case-studies/mortar-data/ (3 Nov. 2014).

Apache Hadoop, 2014. Welcome to Apache™ Hadoop®. http://hadoop.apache.org (3 Nov. 2014).

Apache PIG, 2014. *Welcome to Apache Pig!*. http://pig.apache.org/ (7 Nov. 2014).

Bekkerman, R., Bilenko, M., and Langford, J., 2012. *Scaling up Machine Learning: Parallel and Distributed Approaches.* Cambridge University Press.

Dai, W., and Ji, W., 2014. A MapReduce Implementation of C4.5 Decision Tree Algorithm. *International Journal of Database Theory and Application*, 7(1), pp. 49-60.

Dean, J., and Ghemawat, S., 2004. MapReduce: Simplified Data Processing on Large Clusters. *Proceedings of the 6th Conference on Symposium on Opearting Systems Design and Implementation*, 6, pp. 137-149.

Dhillon, S., and Kaur, K., 2014. Comparative Study of Classification Algorithms for Web Usage Mining. *International Journal of Advanced Research in Computer Science and Software Engineering,* 4(7), pp. 137-140.

Ferreira, R., Oliveira, D., Happ, P., Costa, G., Feitosa, R., and Bentes, C., 2014. InterIMAGE 2: The Architecture of an Open Source, High Performance Framework for Automatic, Knowledge-Based Image Interpretation. *International Geographic Object-Based Image Analysis Conference.* Thessaloniki, Greece.

Grolinger, K., Hayes, M., Higashino, W., L'Heureaux, A., Allison, D., and Capretz, M., 2014. Challenges for MapReduce in Big Data. *IEEE 10th World Congress on Services*, pp. 182-189.

Han, J., Liu, Y., and Sun, X., 2013. A Scalable Random Forest Algorithm Based on MapReduce. *4th IEEE International Conference on Software Engineering and Service Science*, pp. 849-852.

He, Q., Zhuang, F., Li, J., and Shi, Z., 2010. Parallel Implementation of Classification Algorithms Based on MapReduce. *International Conference on Rough Set and Knowledge Technology*, pp. 655-662.

Hypercomp Research Group., 2014. New Digital Repository for Remotely Sensed Hyperspectral Imagery with Unmixing Based Retrieval Functionality. http://www.hypercomp.es/repository/ (3 Nov. 2014)

Intel IT Center., 2012. *Apache Hadoop Community Spotlight: Apache MapReduce* http://www.intel.com/content/www/us/en/big-data/hadoop-spotlight-apache-mapreduce-paper.html (3 Nov. 2014).

Kiran, M., Kumar, A., Mukherjee, S., and Prakash, R., 2013. Verification and Validation of MapReduce Program Model for Parallel Support Vector Machine. *International Journal of Computer Science Issues*, 10(3), pp. 317-325.

Kishor, D., 2013. Big Data: The New Challenges in Data Mining. *International Journal of Innovative Research in Computer Science & Technology,* 1(2), pp. 39-42.

Li, J., Xu, Z., Jiang, Y., and Zhang, R., 2014. The Overview of Big Data Storage and Management. *International Conference on Cognitive Informatics and Cognitive Computing*, pp. 510-513.

Liu, B., Blasch, E., Chen, Y., Shen, D., and Chen, G., 2013. Scalable Sentiment Classification for Big Data Analysis Using Naïve Bayes Classifier. *IEEE International Conference on Big Data*, pp. 99-104.

Machine Learning Group at the University Waikato, 2014. *Weka 3: Data Mining Software in Java*. http://www.cs.waikato.ac.nz/~ml/weka/index.html (3 Nov. 2014)

Mortar Data, 2014. *Mortar*. https://www.mortardata.com/ (3 Nov. 2014).

Nandakumar, A., and Yambem, N., 2014. A Survey on Data Mining Algorithms on Apache Hadoop Platform. *International Journal of Emerging Technology and Advanced Engineering*, 4(1), pp. 563-565.

Olston, C., Reed, B., Srivastava, U., Kumar, R., and Tomkins, A., 2008. Pig Latin: A Not-So-Foreign Language for Data

Processing. *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pp. 1099-1110.

Pakize, S., and Gandomi, A., 2014. Comparative Study of Classification Algorithms Based On MapReduce Model. *International Journal of Innovative Research in Advanced Engineering,* 1(7), pp. 215-254.

Platt, J., 1998. Fast Training of Support Vector Machines using Sequential Minimal Optimization. *Advances in Kernel Methods.* MIT Press, pp. 185-208.

Sagiroglu, S., and Sinanc, D., 2013. Big Data: A Review. *International Conference on Collaboration Technologies and Systems (CTS)*, pp. 42-47.

Shvachko, K., Kuang, H., Radia, S., and Chansler, R., 2010. The Hadoop Distributed File System*.* *IEEE 26th Symposium on Mass Storage Systems and Technologies*, pp. 1-10.

Suthaharan, S., 2014. Big Data Classification: Problems and Challenges in Network Intrusion Prediction with Machine Learning. *ACM SIGMETRICS Performance Evaluation Review,* 41(4), pp. 70-73.

Yadav, C., Wang, S., and Kumar, M., 2013. Algorithm and Approaches to Handle Large Data - A Survey. *International Journal of Computer Science and Network,* 2(3), pp. 37-41.

Zaslavsky, A., Perera, C., and Georgakopoulos, D., 2012. Sensing as a Service and Big Data. *Proceedings of the International Conference on Advances in Cloud Computing (ACC)*, pp. 21-29.