LARGE SCALE THEMATIC MAPPING BY SUPERVISED MACHINE LEARNING ON 'BIG DATA' DISTRIBUTED CLUSTER COMPUTING FRAMEWORKS

J.Lozano, N.Aginako, M.Quartulli, I.Olaizola

Vicomtech-IK4
Digital TV and Multimedia Services
Mikeletegi 57, 20009 Donostia, Spain

E.Zulueta, P. Iriondo

University of Basque Country
Sys. Eng. and Automation Department
Alameda de Urquijo, 48013 Bilbao, Spain

1. INTRODUCTION

The Petabyte-scale data volumes in Earth Observation (EO) archives are not efficiently manageable with serial processes running on large isolated servers. Distributed storage and processing based on 'big data' cloud computing frameworks needs to be considered as a part of the solution.

This contribution describes a parallelized data processing approach for EO image analysis that is based on the MapReduce [1] paradigm and implemented on the Apache Spark [2] framework. Existing algorithms for e.g. thematic mapping need to be re-defined in order to exploit distributed execution capabilities to run on large coverage data.

The Apache Spark engine, originally widely used in text analytics, can be exploited for processing, filtering and sorting image-oriented data across multiple servers (the "map" operation), and grouping and summarizing the obtained results (the "reduce" operation). A large variety of machine learning algorithms can be defined in terms of map- and reduce-operations, thus making them available on computing clusters via Spark.

The thematic mapping approach presented in [3] is based on a serial implementation of a probabilistic k-Nearest Neighbor supervised classification approach that produces high quality results. The algorithm in itself, as it is often the case with machine learning, is inherently parallelizable [4], yet it needs to be revised in order to manage big volumes of data efficiently in terms of performance. Since the algorithm is coded in a high level scripting language, the processing time needed for the classification of a 25 Megapixel image is of about a minute. If these values are extrapolated to regional extensions, unacceptable running times are obtained as a result. As a concrete example, the generation of metric thematic maps of the Basque country in the north of Spain would require analyzing about 150 Megapixels of data, hence obtaining running times of the order of 20 hours. While the parallelization and distribution of analysis processes can provide evident advantages, porting classical machine learning algorithms intended for limited data volumes to the domain of large scale remote sensing data coverages requires a significant effort. The adoption of a parallelization approach based on the MapReduce paradigm can be beneficial in this respect.

In this contribution, we present a methodology for the parallelization of machine learning algorithms on local and cloud-based cluster computing environments for the efficient analysis of large geospatial EO coverages.

2. METHODOLOGY

In the image in figure 1, we present a functional schema of the proposed idea. First we select the geographical area to be analyzed and the data sources. This geographical area is divided into tiles with limited size, to be able to easily distribute the analysis through the framework. Primitive feature extraction, being completely independent for each separate tile, can be expressed using map-like functions that will be executed in each node of the framework. Supervision can be effectively run on a small training subset of the original large scale dataset. The resulting trained classifier can then be distributed to the different processing nodes and evaluated onto the extracted primitive feature in a further perfectly parallel operation. A merging phase (perhaps to a lower resolution

version) can be expressed by a reduce operation collecting the distributed data in a serialized file-based representation on disk.

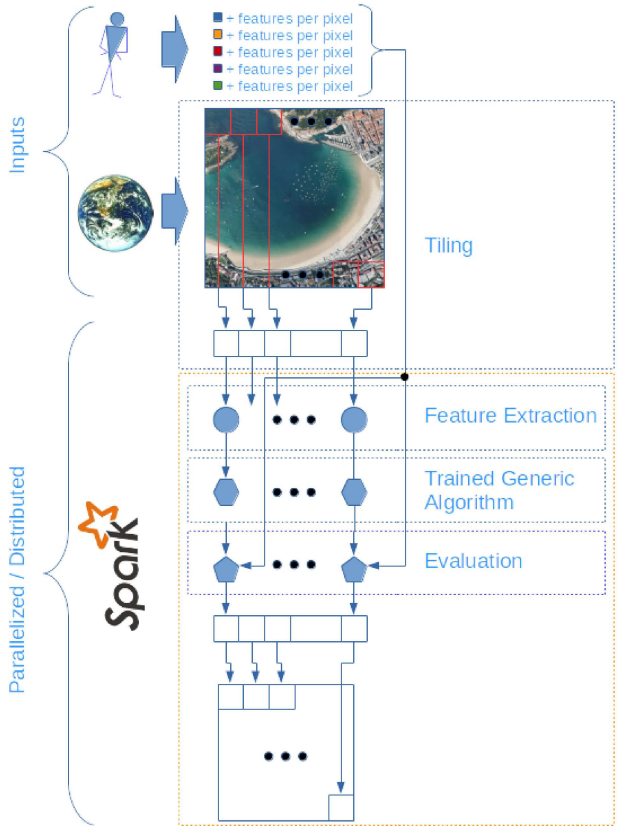

Fig. 1. High level architectural diagram of the implemented processing flow. Tiling of large scale raster coverages allows distributing independent primitive feature extraction and supervised classification based on a model trained on limited data on the nodes of an in-memory big data cluster computing framework of the kind of Apache Spark.

3. EXPERIMENTAL APPROACH

The dominant MapReduce implementation as of 2015 is the open source Apache Hadoop framework, that relies heavily on disk serialization of intermediate results. We choose to implement our prototype on Apache Spark, an alternative implementation that is optimized for in-memory, iterative processing of the kind often encountered in machine learning. The implemented prototype is instantiated and run on the Amazon Elastic Computing Cluster infrastructure-as-a-service, in order to be able to better evaluate optimal horizontal and vertical scaling options for the system. Its availability also allows us to experimentally evaluate optimal data partitioning paramets for a specific data analysis configuration.

The implemented system can be seen as a distributed catalogue for a tile-based partitioned coverage. Each tile is described as a completely autonomous entity in terms of its metadata as well as by a raster data cube that corresponds to a original acquisition on which image analysis procedures can be run.

Training data provided by a human supervisor can be used to instantiate a classifier object that we can distribute to cluster nodes for data evaluation and for the final creation of classified tiles.

Finally, all tiles can be merged in a unique layer to be compared with ground truth to obtain statistical measures of classification performance [5].

As mentioned in previous section, the modularity offered by Apache Spark allows us to easily apply different clustering and (trained) classification algorithms with a standardized interface.

The experimental results presented in this abstract correspond to a Gaussian naive Bayesian supervised classification algorithm [5]: given a class variable y and a dependent feature vector x_1 through x_n, the Bayes' theorem

$$P(y|x_1, ..., x_n) = \frac{P(y)P(x_1, ..., x_n|y)}{P(x_1, ..., x_n)}$$

can be restated using the "naive" independence assumption $P(x_i|y, x_1, ..., x_n) = P(x_i|y)$ as

$$P(y|x_1, ..., x_n) = \frac{P(y)\Pi_{i=1}^{n} P(x_i|y)}{P(x_1, ..., x_n)} .$$

Since $P(x_1, ..., x_n)$ is constant given a specific input, the classification rule becomes

$$P(y|x_1, ..., x_n) = P(y) \prod_{i=1}^{n} P(x_i|y)$$

$$\Downarrow$$

$$\hat{y} = \arg\max_y P(y)\Pi_{i=1}^{n} P(x_i|y)$$

where the likelihood of the features $P(x_i|y)$ is assumed to be a Gaussian $P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2})$.

4. EVALUATION

To be able to evaluate the proposed solution and compare it to previous developments [3] and to an existing ground truth map, we have limited the area to be analyzed to a 25 Megapixel optical airborne scene from Donostia - San Sebastian, from the Open Data Euskadi repository (http://opendata.euskadi.net/).

In this area, we have defined six principal classes of interest: Beach, Buildings, Paths, Rocks, Sea and Vegetation. The evaluation process has been carried out in terms of both classification quality and system performance. The evaluation of the classification quality is based on classical statistical measures, like Precision, Recall, F1 and Accuracy.

With respect to a similar yet fully serial system based on a probabilistic k-Nearest Neighbor classification scheme in [3], the obtained results (figure 2) show a slight decrement in F1 and Precision values, with moderate improvements in the Recall. Accuracy increased slightly for almost each class.

System performance is evaluated in terms of processing time. This processing step includes feature extraction from the images to evaluate and its evaluation in corresponded classification algorithm model.

Typical processing times for the presented prototype entailed 62.68 seconds to complete the classification task for a single CPU running sequentially. The presented prototype has been run with different tile sizes to evaluate the performance of the distributed in-memory system. Figure 3b shows the obtained result for the distributed system with different tile sizes in comparison with the single processor result. 128 by 128 pixel sized tiles show the best performance in the considered configuration of the distributed system corresponding to an improvement of a factor of more than two for four workers.

If the option of a migration to cloud computing is considered, the possibility to design an architecture with more nodes, than the notebook can simulate, will lead into the decrement of needed time drastically.

5. CONCLUSIONS

The problem of large scale thematic mapping is a central one for the exploitation of modern EO acquisition systems and catalogs.

To this end, we have introduced a simple yet efficient architecture and presented an implementation of a pre-operational prototype capable of distributing supervised and unsupervised thematic mapping processes onto modern in-memory cluster computing frameworks of the kind of Apache Spark.

Fig. 3. Resulting thematic layer obtained after processed tile collection from the distributed prototype implemented based on the Apache Spark framework and

Experimental evaluation of the classification performance has been complemented by perliminary processing performance evaluations that have highlighted the influence of tile size in the effectiveness of the approach based on a distributed in-memory framework.

While preliminary results on real data appear convincing, for the final contribution a thorough quantitative evaluation of the learning performance will be conducted on the available ground truth data corresponding to the processed scene.

Furthermore, the preliminary processing performance tests will be complemented by horizontal and vertical scalability experiments.

6. REFERENCES

[1] Jeffrey Dean and Sanjay Ghemawat, "Mapreduce: Simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, Jan. 2008.

[2] Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, and Ion Stoica, "Spark: Cluster computing with working sets," in *Proceedings of the 2Nd USENIX Conference on Hot Topics in Cloud Com-*

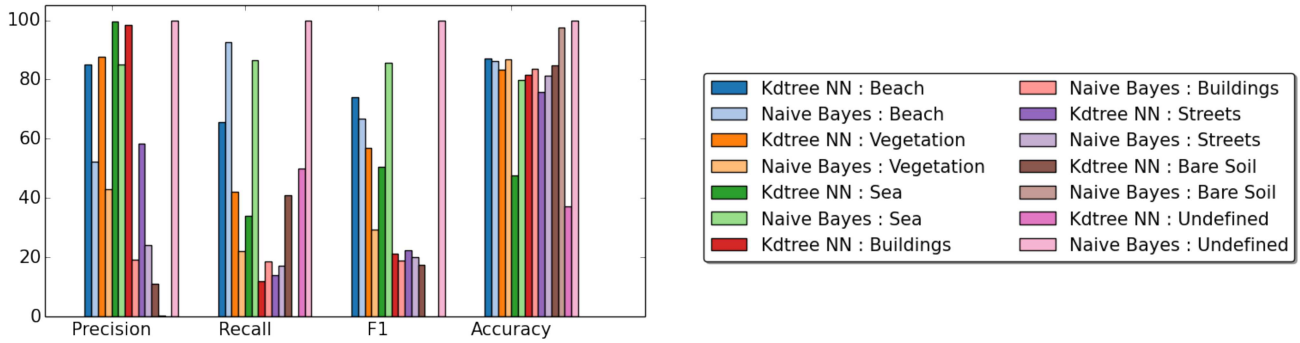

Fig. 2. Statistical measures comparative between developed prototype and previous prototype in [3].

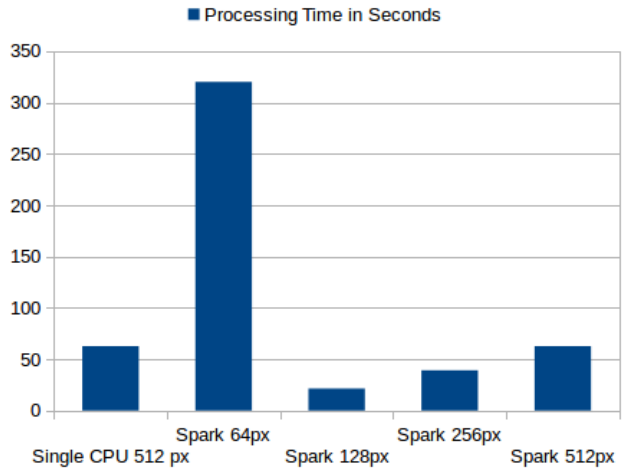

Fig. 4. Processing time comparative Apache Spark framework vs standard CPU

puting, Berkeley, CA, USA, 2010, HotCloud'10, pp. 10–10, USENIX Association.

[3] Javier Lozano, Naiara Aginako, Marco Quartulli, Igor G.Olaizola, and Ekaitz Zulueta, "Scalable machine learning for fast thematic mapping in web server," in *Proceedings of the 2014 conference on Big Data from Space (BiDS '14)*. 2014, pp. 38–41, Publications Office of the European Union.

[4] Ujjwal Maulik and Anasua Sarkar, "Efficient parallel algorithm for pixel classification in remote sensing imagery," *GeoInformatica*, vol. 16, no. 2, pp. 391–407, 2012.

[5] Harry Zhang, "The optimality of naive bayes," in *Proceedings of the Seventeenth International Florida Artificial Intelligence Research Society Conference (FLAIRS 2004)*, 2004.