A Complete Data Science Work-flow For Insurance Field

Mohammed Ghesmoune
University of Paris 13
LIPN-UMR 7030 - CNRS
99, av. J-B Clément,
93430 Villetaneuse, France
ghesmoune@lipn.univ-paris13.fr

Mustapha Lebbah
University of Paris 13
LIPN-UMR 7030 - CNRS
99, av. J-B Clément,
93430 Villetaneuse, France
lebbah@lipn.univ-paris13.fr

Hanane Azzag
University of Paris 13
LIPN-UMR 7030 - CNRS
99, av. J-B Clément,
93430 Villetaneuse, France
azzag@lipn.univ-paris13.fr

Salima Benbernou
University of Paris Descartes
45 rue des Saints Pères,
LIPADE, Paris Cedex 06, France
salima.benbernou@parisdescartes.fr

Mourad Ouziri
University of Paris Descartes
45 rue des Saints Pères,
LIPADE, Paris Cedex 06, France
Mourad.Ouziri@parisdescartes.fr

Tarn Duong
University of Paris 13, Sorbonne Paris City
LIPN-UMR 7030 - CNRS
99, av. J-B Clément,
93430 Villetaneuse, France
duong@lipn.univ-paris13.fr

***Abstract*—In recent years, "Big Data" has become a new ubiquitous term. Big Data is transforming science, engineering, medicine, health-care, finance, business, and ultimately our society itself. Learning from Big Data has become a significant challenge and requires development of new types of algorithms. Most machine learning algorithms can not easily scale up to Big Data. MapReduce is a simplified programming model for processing large datasets in a distributed and parallel manner. In this paper, we present our work carried in a big data project[1] which is dedicated to the insurance sector. This allows us to validate our method on real-world data for insurance. We present the complete pipeline or work-flow going from data collection to visualization, passing by data fusion, data analysis, clustering, and prediction tasks. The insurance dataset is enriched with data collected from heterogeneous sources. A predictive and analysis system is proposed by combining the clustering result with decision trees. We use the topological approach, especially the SOM method, for its interest in being able to cluster and visualize the data at the same time. We make the source code of our SOM-MapReduce algorithm, written with Spark using the MapReduce paradigm, publicly available[2].**

***Keywords*-Data fusion, RDF, Semantic, Entity resolution, Big Data, Map-Reduce, Spark, Data clustering, SOM, Prediction, Insurance data, Visualization.**

I. Introduction

Many challenges arising from the Big Data fusion include how to integrate data from multiple and heterogeneous data sources, how to identify the meaning between entities from different sources [1], how to handle the inconsistent naming styles in different data sources, and how to resolve the conflicting data types for the same entity. The Linked Data paradigm allows us to describe a recommended best practice for displaying, sharing and connecting data, information and knowledge on the Semantic Web using URIs, the RDF model of data, and ontologies. RDF[3] is a conceptual description of information modeling that is implemented in Web resources, using a variety of syntax notations and data serialization formats (XML, n-triple, turtle).

In this paper, we consider clustering multi-dimensional data. Clustering is a key in a variety of areas: machine learning, data mining, pattern recognition, social network etc. It is difficult to store and analyze a large volume of data on a single machine with a sequential algorithm [2]. In such situations, the MapReduce (MR) programming paradigm is used to overcome this problem [3]. The MR programming model was designed to simplify the processing of large files on a parallel system through user-defined Map and Reduce functions [4].

A. Applied data science

In a recent work, we have presented a Big Data work-flow based on the streaming approach, where data are processed continuously in real time [5]. In this paper, we will use the batch approach for data processing and clustering. More precisely, we are concerned with designing clustering algorithm named Self-Organizing Map (SOM, [6]) using MapReduce. We use the emerged open-source implementation named Spark [7]. We design a complete distributed SOM clustering solution using Spark and MapReduce paradigm. We demonstrate the utility of SOM-MR as a method for unsupervised learning to explore insurance Big Data. We make the source code of our SOM-MR algorithm, written with Spark using the MapReduce paradigm as well as our project on the Spark-Notebook platform publicly available[4]. We present the complete pipeline or work-flow going from data collection to visualization, passing by data fusion using RDF, data

[1] http://ns209168.ovh.net/squarepredict/

[2] https://github.com/Spark-clustering-notebook/coliseum

[3] https://www.w3.org/RDF/

[4] https://github.com/Spark-clustering-notebook/coliseum

978-1-5386-5035-6/18/$31.00 ©2018 IEEE

analysis, clustering, and prediction tasks. A predictive and analysis system is proposed by combining the clustering result with decision trees.

The remainder of this paper is organized as follows: Section II summarizes the architecture of our platform. Section III is dedicated to related works. Section IV outlines the data fusion from different heterogeneous sources. Section V presents our SOM MapReduce using the Spark open source platform. Section VI provides the experimental evaluation on both insurance dataset and the comparisons between two manners to design MapReduce function. Finally, Section VII concludes this paper.

II. Architecture of the Big data framework

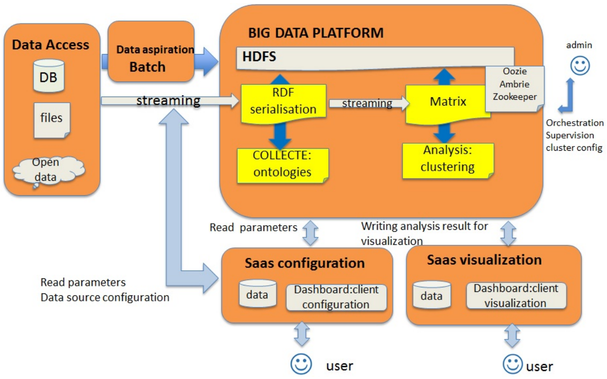

Figure 1: Big data platform

In this section, we describe the Big data platform developed from the collection to visualization step, and is depicted in Fig. 1. The application we targeted is insurance.

1) **Data sets**. The data in our platform are collected from heterogeneous sources including proprietary (housing insurance contracts), and different open data sets such as the French national institution of statistics INSEE[5] that contains information related to census household and housing surveys (i.e., type of heating, proportions of housing type in the local area etc.), the ONDRP[6] which is a department of the National Institute of Advanced Studies of Security and Justice, which contains information related to crime and delinquency (i.e., home invasions, average of armed burglaries against individuals in their homes, etc.), as well as the well known open data base Dbpedia amongst others. The data have different format (RDF, CSV, etc.).
2) **Data aspiration batch.** The data are collected through a classical ETL as a batch and considered and waved to the platform, and then transformed in appropriate format (RDF) and finally stored in HDFS.
3) **SaaS Configuration.** The component is a software which is a Service that provides a dashboard to help a user to process a configuration on the data and transfer the data to be represented into RDF in the platform.
4) **RDF serialization and ontologies**. In order to provide a semantic reasoning by inferring new hidden data, all data are represented in RDF. Consequently, RDF data are processed in serialized n-triples format *subject-predicate-object*. Moreover, the semantic links are built to connect RDF data of each data source with the concepts of an OWL ontology. The fusion process uses those connections to infer semantic relations (subsumption, equivalence, disjointness, etc.) across the data-sources and identify duplicates of the same real world entities (the owl sameAs relationships).
5) **Clustering and Analysis** The aim of clustering is to separate the data set (waved from the collection process in a matrix format) into a small number of groups where the members within a cluster are similar to each other, and members from different clusters are different to each other. Clusters are useful for data reduction, analyzing and understanding the deep structure of the data set. Since we are merging heterogeneous data sets from different sources, clustering provides an analytically tool to quantify the new information created by this newly merged data set, with respect to the individual data sets.
6) **Visualization** Visualizations are effective in indicating the directions in which the analysis should proceed as they can present key aspects of the data set in a single graphical summary which would be not evident in a numerical form.

III. Related work

We categorize the related work as follows.

Big data fusion

Data integration has been much studied in the last decade in the database community [8], [9]. Moreover, some tools that have been developed for RDF query evaluation, such as Jena[7] or Sesame[8], are not suited to Big Data since they require the loading of previously established data in memory before evaluating them. It is then necessary to develop a SPARQL query execution engine adapted to Big Data with the help of MapReduce [10]–[13].

Big data clustering

An attractive way to assist the analysts in data exploration is to base on unsupervised approaches allowing clustering and mapping high-dimensional data in a low-dimensional space. The self-organizing maps (SOM) [6] can be used as a clustering method that addresses these issues. MapReduce is the most popular programming paradigm suited for data

[5] http://www.insee.fr/fr/bases-de-donnees/default.asp?page=open-data/open-data-utilisation.htm

[6] http://www.inhesj.fr/fr/ondrp

[7] https://jena.apache.org/

[8] http://rdf4j.org/

already stored on a distributed file system, which offers data replication as well as the ability to execute computations locally on each data node. The work [14] proposed a parallel and distributed implementation of *k*-means in MapReduce. MR-DBSCAN [15] is a scalable MapReduce-based DBSCAN algorithm. Many works have been proposed to scale-up the EM algorithm and the parallel implementation of EM proposed in [16] is coded in Spark.

IV. Big data fusion

In this section, based on our work in [17], we present two aspects when Big Data fusion is processed: the entity resolution approach based on inference mechanisms to manage the ambiguity of real world entities for linking data at the semantic and URI levels, and a query evaluation based on entity resolution results in order to include implicit data into query results using the MapReduce paradigm to deal with huge volumes of data.

A. Entity resolution approach

Each data source uses its own OWL ontology (as a conceptual model) and identifies the resource using internal URIs (as an entity identification). Therefore, the same entities may be described using different or equivalent concepts (semantics) identified by different URIs among different data sources. The entity resolution rules are applied on the RDF data using a resolution algorithm. We propose a MapReduce algorithm that triggers entity resolution rules in a parallel manner on distributed small pieces of data. The algorithm reconciles pairs of entity fragments matching a functional key that appear in the antecedent of the resolution rules.

B. MapReduce based Query evaluation

We present in this section a MapReduce query evaluation approach to compute a complete query result by including implicit data. We propose a query rewriting algorithm based on the MapReduce paradigm in order to enrich a user query by adding more RDF patterns that explicitly refer to the implicit data. This is processed in two steps. In the first step, a query plan composed of MapReduce jobs is generated for the query. In the second step, the generated query plan is evaluated in a Hadoop framework to produce the results.
The user query is rewritten using the inference rules, including the entity resolution as *SameAs* relationship rules. The inference rules are of the form: *antecedent* ⇒ *goal*. The list of inference rules contains the RDFS, the OWL and the axiom rules defined by the user. The inference rules are applied by a backward reasoning algorithm. For a given query, the algorithm generates (1) a MapReduce plan by applying inference rules to enrich query patterns and (2) the MapReduce jobs. For each query pattern, the algorithm generates new sub-patterns corresponding to the antecedent of the rules whose goal matches the pattern.

Finally, the enriched data are now ready to be translated to the analysis and clustering component. The enriched data are transformed in appropriate format (matrix) to the clustering component.

V. Spark-MapReduce and SOM

To handle the huge amount of data, it is necessary to use distributed architecture.

In the SOM algorithm we identified theses atomic MapReduce parts:

- Assign each observation $\mathbf{x}_i$ to the best match unit using expression 1

$$\phi(\mathbf{x}_i) = \arg\min_r \|\mathbf{x}_i - \mathbf{w}_r\|^2 \tag{1}$$

- Accumulate denominator and numerator for each cell $c \in C$
- Update weight vectors $\mathbf{w}_c$ (eq. 2)

$$\mathbf{w}_c = \frac{\sum_{\mathbf{x}_i \in \mathcal{A}} \mathcal{K}^T(\delta(c, \phi(\mathbf{x}_i))\mathbf{x}_i}{\sum_{\mathbf{x}_i \in \mathcal{A}} \mathcal{K}^T(\delta(c, \phi(\mathbf{x}_i)))} \tag{2}$$

A. SOM MapReduce-based method

In the proposed method, map outputs are merged in one value, so the key of the output is not used. The Map value of the output is a matrix and a neighborhood vector. The matrix is constituted by rows of data vectors $\mathbf{x}_i$ who are themselves multiplied by the neighborhood factors $\mathcal{K}^T(\delta(c, \phi(\mathbf{x}_i)))$. All those neighborhood factors are stored in the neighborhood vector. So the size of the output matrix is the number of prototypes multiplied by the size of the data vectors $(k \times n)$. The size of the neighborhood vector is the number of prototypes. The reduce function just sums all matrices and all neighborhood vectors together. The new model matrix is computed by dividing the sum of matrices and the sum of the neighborhood vectors. We denote $\mathcal{H}(k \times n)$ as neighborhood matrix, which elements are defined as follows: $\mathcal{H}_{i,j} = \mathcal{K}^T(\delta(i,j))$ We also consider that $\mathcal{H}_{:,j}$ $(\mathcal{H}_{i,:})$ denotes the column j (the row i) of the matrix $\mathcal{H}$. The Reduce function accumulates each data vector assigned to each prototype and counts them. The prototype matrix $\mathcal{W}$ is the accumulation divided by the denominator. Thus Map and Reduce functions are defined as follows:

$$\begin{aligned} MapNumerator(\mathbf{x}_i) &= \mathcal{H}_{:,\phi(\mathbf{x}_i)} \times \mathbf{x}_i \\ MapDenom(\mathbf{x}_i) &= \mathcal{H}_{:,\phi(\mathbf{x}_i)} \\ \mathbf{R}educe() &= \frac{\sum_{\mathbf{x}_i \in \mathcal{A}} MapNumerator(\mathbf{x}_i)}{\sum_{\mathbf{x}_i \in \mathcal{A}} MapDenom(\mathbf{x}_i)} \end{aligned}$$

For more details, please refer to [18].

VI. Experimental evaluations

A. Application for synthetic datasets

We implemented our algorithms https://github.com/TugdualSarazin/spark-clustering SOM MapReduce in Spark 0.7.3 and we compared them on a amazon EC2 cluster of

24 xlarge computers. Each computer has 4 cores and 15GB of RAM, so the total capacity of the cluster is of 96 cores and 360 GB of RAM.

B. Application for insurance field

Classification and regression trees (CART) are a useful technique for creating easily interpretable decision rules, see [19]. In the following we present an analysis combining an unsupervised learning with a supervised method. The SOM-MR algorithm is used as an unsupervised method while the regression trees are used to explain the clusters produced by the SOM-MR approach.

1) Exploratory data analysis of SOM-MR clusters: The 2012 insurances payouts data consists of a sample of 2,130,114 contracts enriched with open data from the INSEE and ONDRP. The SOM-MR clustering was carried out on these data, resulting in 100 clusters. The goal of is to construct a decision tree model of these enriched data in determining the payouts made for water damage (DDE) claims `charge_dde` and for the payouts made for fire damage (INC) claims `charge_inc` within each of the SOM-MR clusters.

As `charge_inc` and `charge_dde` are continuous variables, a regression tree analysis is appropriate. For each of the SOM-MR clusters, a regression tree with the response variable being `charge_inc` or `charge_dde`, and the covariates being the other variables. For the fire damages claims, these regressions trees are displayed in Figure 2 in decreasing order of the total sum of payouts per cluster. In each tree, the node labels contain two values: inside the lozenge is the total payouts, and below it is the number of claims. At each binary split, the left and right branches indicate the rule applied to the splitting variable. All the decision trees begin with the decision `nbsin_inc` < 0.5 which separates all the contracts with/without any damage claims at the root node. All the contracts without any claims becomes a terminal node, as they also do not contribute to the payouts. All the claims are then decomposed with further decision rules based on the covariates. For example, if we focus on the SOM-MR NumCluster=9 in the middle panel in Figure 2, we observe that the terminal node (labeled internally 55) has a payout total greater than €180K from only 8 contracts. A similar analysis can be carried out with the `nbsin_dde` decision trees.

2) Analysis of the insurance big data using SOM-MR: To further analyze clusters, we use the following 3 indicators: rate of claims, payouts per contract, and loss per contract.

$$Rate\_of\_claims = \frac{Number\_of\_claims}{Number\_of\_contracts} \quad (3)$$

$$Payout\_per\_claim = \frac{Sum\_of\_claim\_amounts}{Number\_of\_claims} \quad (4)$$

$$Loss\_per\_contract = Rate\_of\_claims \times Payout\_per\_claim \quad (5)$$

Regarding these indicators, especially the maximum and minimum values, the insurance company can focus its analysis on the corresponding clusters. Thus, a model based on the features of assigned data can be defined. Using this model, the insurance company can predict the payouts for a new customer within a cluster and so propose more personalized insurance contracts for its customers.

3) Supervised learning of SOM-MR clusters: In the previous section, we examined decision trees for the exploratory analysis of the SOM-MR clusters. In this section, we examine decision trees for the prediction of these SOM-MR cluster labels in a supervised learning context. The response variable is the SOM-MR cluster label `NumCluster`, which is a categorical variable, a classification tree is appropriate.

The categorical department variable `dept` (94 levels) causes a combinatorial explosion when used in conjunction with the categorical response `NumCluster` (20 levels) in a decision tree. The `dept` variable is not well-suited as an ordinal variable. Longitude and latitude are better adapted as, say `longitude` < 2.50, has a geographical meaning. So `dept` was replaced by the longitude and latitude of the prefecture of each department (`longitude`, `latitude`). We compute three decision trees: one with both the INSEE and ONDRP variables (Figure 3), one with the INSEE variables only, and one where the commune level INSEE variables are replaced by their departmental means in order to be comparable to the ONDRP variables.

For the decision tree with both added INSEE and ONDRP variables in Figure 3, the geographical variables `longitude` and `latitude` are important as `dept` previously, though the ONDRP crime variables are more important here than the INSEE housing variables. Each leaf node has a color-coded label which denotes the estimated cluster label obtained by following this decision tree. For each leaf node is annotated with the percentage of these contracts whose original SOM-MR cluster label coincide with the estimated label and the number of contracts contained in the node (n).

Removing the ONDRP variables is too drastic in order to assess the influence of the INSEE variables. The ONDRP variables are available at the departmental level whereas the INSEE variables at the commune (INSEE code) level. In terms of finding groups of similar values, it is more likely to occur for the more aggregated ONDRP variables than the lower level INSEE ones. To remedy this, we replace the commune level INSEE variables with their mean aggregated at the department level with the suffix `_MOYD`.

4) Validation of SOM-MR clusters: A comparison of the distributions of the summary statistics is easier with the graphical bar charts in Figure 4. Within each cluster, there are four bars: the violet are the 2012 data with the true SOM-

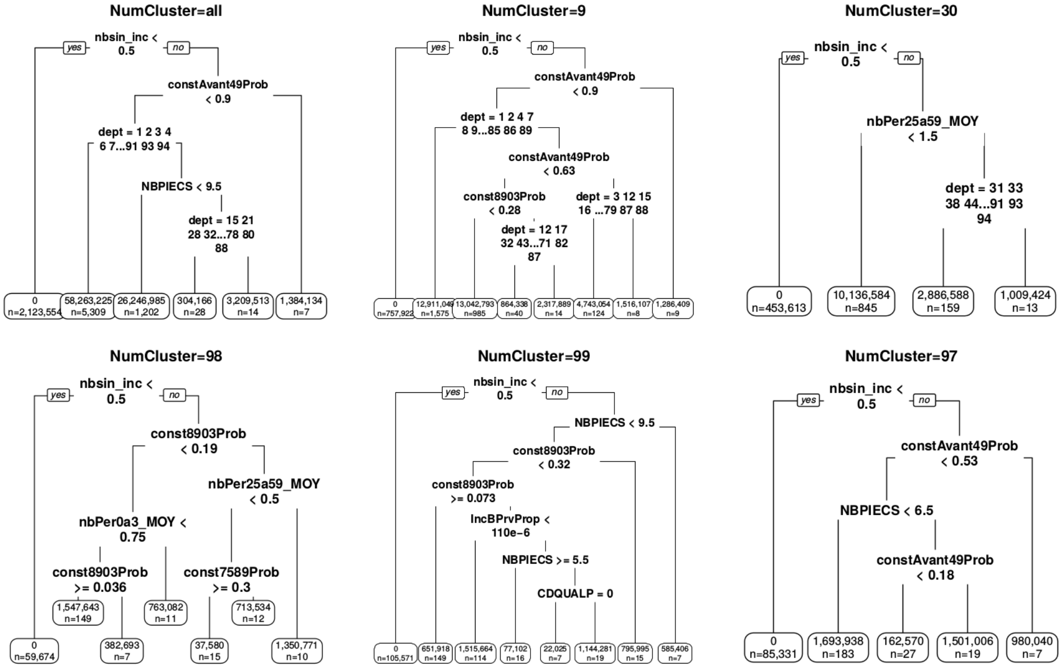

Figure 2: Decision trees for the enriched insurance data for `charge_inc` (fire damages) payouts, sorted by SOM cluster payouts: NumCluster = all, 9, 30, 98, 99, 97.

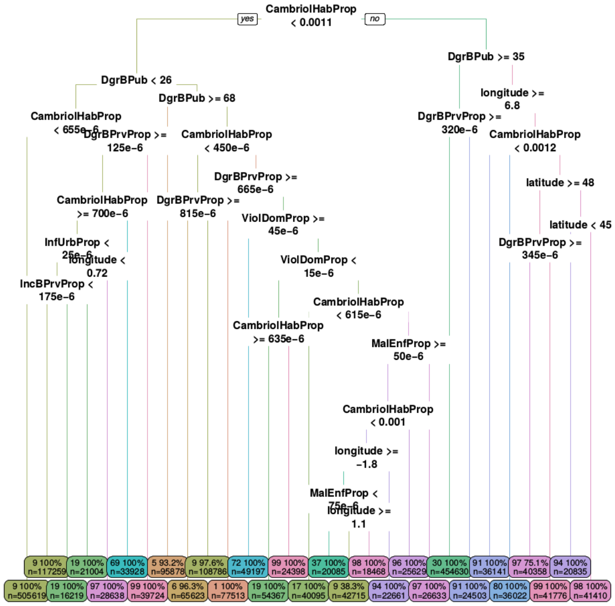

Figure 3: Decision tree for predicting the SOM-MR cluster labels of the enriched insurance data: INSEE and ONDRP variables.

MR clusters, the turquoise is the 2012 data re-classified using the decision tree, the green is for 2011, and the orange is for 2010. For 2010, 2011 there are 1.60 and 1.68 million contracts, €64.1M and €66.7M of fire damages, and €12.2M and €9.11M of water damages. In comparison for 2012, there are 2.1 million contracts, €89.4M of fire and €19.3M of water damages. So we can expect that the heights of the bars for the former to be around 20%, 20% and 40–50% lower than the latter. Taking these into account, the match between the number of contracts is good, especially for the highest bars in clusters 9 and 30. For the fire damages, cluster 30 is proportionally over-represented for 2010, 2011, but nonetheless does not exceed the 2012 level. For the water damages, clusters 9, 30 for 2011 appears to be under-represented and cluster 9 is over-represented for 2010, in comparison to 2012. Overall the SOM-MR cluster labels from 2012 are validated for clustering the 2010, 2011 data in terms of the number of contracts and fire damages, but less so for the water damages.

VII. Conclusion

In this paper, we have presented a complete data science work-flow through a real application for insurance field. We have learned a lot from this experience by showing that Big Data should be handled by different specialized communities from the database, knowledge reasoning and machine learning fields. We have implemented a platform including a set of models, algorithms, benchmarks for collecting the heterogeneous data, processing the fusion, the analysis, the clustering and finally the visualization.

Experimental evaluation demonstrated the effectiveness and efficiency of the presented Big Data work-flow. The

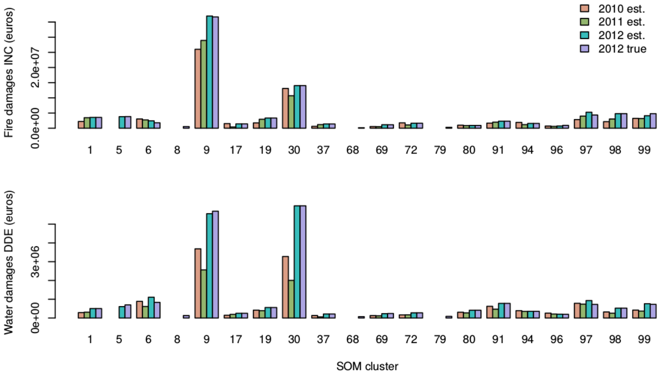

Figure 4: Validation of SOM cluster labels from decision tree for enriched insurance 2012 data with departmental mean of INSEE variables. Validation data are contracts from 2010 (orange), 2011 (green) and 2012 (turquoise). The 2012 data with true SOM clusters are violet. INC is fire damages claims (euros), DDE is water damages claims (euros).

utility of the work-flow as a suite of tools for data analytics has been demonstrated for insurance dataset.

We plan in the future to extend SOM-MR to deal with binary, categorical, and mixed data, and also to make our algorithm as autonomous as possible. Also, we envisage to set up a Lambda Architecture [20] where the SOM-MR algorithm will serve as an offline layer.

References

[1] H. Wache, T. Voegele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S. Hübner, "Ontology-based integration of information-a survey of existing approaches," in *IJCAI-01 workshop: ontologies and information sharing*, vol. 2001. Citeseer, 2001, pp. 108–117.

[2] H.-P. Kriegel, P. Kröger, and A. Zimek, "Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering," *ACM Trans. Knowl. Discov. Data*, vol. 3, no. 1, pp. 1:1–1:58, Mar. 2009.

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

[4] H. J. Karloff, S. Suri, and S. Vassilvitskii, "A model of computation for mapreduce," in *SODA'10*, 2010, pp. 938–948.

[5] M. Ghesmoune, H. Azzag, S. Benbernou, M. Lebbah, T. Duong, and M. Ouziri, "Big data: from collection to visualization," *Machine Learning*, pp. 1–26, 2017. [Online]. Available: http://dx.doi.org/10.1007/s10994-016-5622-4

[6] T. Kohonen, *Self-organizing Maps*. Springer Berlin, 2001.

[7] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: cluster computing with working sets," in *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*, ser. HotCloud'10. Berkeley, CA, USA: USENIX Association, 2010, pp. 10–10.

[8] C. A. Knoblock, P. Szekely, J. L. Ambite, S. Gupta, A. Goel, M. Muslea, K. Lerman, M. Taheriyan, and P. Mallick, "Semi-automatically mapping structured sources into the semantic web," in *Proceedings of the Extended Semantic Web Conference*, Crete, Greece, 2012.

[9] S. Endrullis, A. Thor, and E. Rahm, "WETSUIT: an efficient mashup tool for searching and fusing web entities," *PVLDB*, vol. 5, no. 12, pp. 1970–1973, 2012. [Online]. Available: http://vldb.org/pvldb/vol5/p1970_stefanendrullis_vldb2012.pdf

[10] J. Du, H. Wang, Y. Ni, and Y. Yu, "Hadooprdf: A scalable semantic data analytical engine," in *Intelligent Computing Theories and Applications - 8th International Conference, ICIC 2012, Huangshan, China, July 25-29, 2012. Proceedings*, 2012, pp. 633–641. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-31576-3_80

[11] N. Papailiou, D. Tsoumakos, I. Konstantinou, P. Karras, and N. Koziris, "H_2rdf+: an efficient data management system for big RDF graphs," in *International Conference on Management of Data, SIGMOD 2014, Snowbird, UT, USA, June 22-27, 2014*, 2014, pp. 909–912. [Online]. Available: http://doi.acm.org/10.1145/2588555.2594535

[12] J. Subercaze, C. Gravier, J. Chevalier, and F. Laforest, "Inferray: fast in-memory rdf inference," *Proceedings of the VLDB Endowment*, vol. 9, no. 6, pp. 468–479, 2016.

[13] H. Halpin, P. J. Hayes, J. P. McCusker, D. L. McGuinness, and H. S. Thompson, "When owl: sameas isnt the same: An analysis of identity in linked data," in *International Semantic Web Conference*. Springer, 2010, pp. 305–320.

[14] W. Zhao, H. Ma, and Q. He, "Parallel k-means clustering based on mapreduce," in *Cloud computing*. Springer, 2009, pp. 674–679.

[15] Y. He, H. Tan, W. Luo, S. Feng, and J. Fan, "Mr-dbscan: a scalable mapreduce-based dbscan algorithm for heavily skewed data," *Frontiers of Computer Science*, vol. 8, no. 1, pp. 83–99, 2014.

[16] H. Cui, J. Wei, and W. Dai, "Parallel implementation of expectation-maximization for fast convergence."

[17] S. Benbernou, X. Huang, and M. Ouziri, "Fusion of big rdf data: A semantic entity resolution and query rewriting-based inference approach," in *International Conference on Web Information Systems Engineering*. Springer, 2015, pp. 300–307.

[18] T. Sarazin, H. Azzag, and M. Lebbah, "SOM clustering using spark-mapreduce," in *2014 IEEE International Parallel & Distributed Processing Symposium Workshops, Phoenix, AZ, USA, May 19-23, 2014*, 2014, pp. 1727–1734. [Online]. Available: http://dx.doi.org/10.1109/IPDPSW.2014.192

[19] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, ser. Springer series in statistics. Springer-Verlag New York, 2009.

[20] N. Marz and J. Warren, *Big Data: Principles and best practices of scalable realtime data systems*. Manning Publications Co., 2015.