Merging Logical Analysis of Data Models

Osama Elfar and Soumaya Yacout
Mathematical and Industrial Engineering Department
École Polytechnique de Montréal
Montréal, Québec, Canada H3C 3A7
osama.elfar@polymtl.ca, soumaya.yacout@polymtl.ca

Hany Osman
Systems Engineering Department
King Fahd University of Petroleum & Minerals, KFUPM
Dhahran, 31261, Kingdom of Saudi Arabia
hanyosman@kfupm.edu.sa

Said Berriah
Research and Development Department
R2 Intelligent Technologies
Montréal, Québec, Canada, H2Y 3X7
said.berriah@r2.ca

Abstract

Logical Analysis of Data (LAD) is a classification algorithm known for its high accuracy and interpretable results, but it has relatively long computational time, which makes it unsuitable to treat big data. In this paper, we propose a platform for improving the computational efficiency of LAD especially in large scale and big data applications. We develop an approach based on a merging operation of two different LAD models. The proposed approach is ideal to develop incremental learning platform based on LAD. In addition, it enables LAD to be performed in parallel computing environments. The suggested operation determines the intersection between each pair of patterns from the two models. Each intersection is a new pattern in the new merged model and its characteristics are inherited from its parent patterns. Numerical experiments are conducted by using an industrial data set. The experiments demonstrate the trade-off between accuracy and computational efficiency of the proposed approach and the classical implementation of LAD with a sole run on the whole data. The proposed approach resulted in a maximum accuracy reduction of 5% and computational time reduction of 87% from the accuracy and computational time of the classical implementation of LAD, respectively.

Keywords

Logical Analysis of Data; Merging LAD Models; Incremental Learning; Big Data Analysis.

1. Introduction

Machine learning is one of the artificial intelligent (AI) applications that give the ability to learn and improve systems automatically. Algorithms of machine learning are categorized into supervised and unsupervised machine learning algorithms. Supervised machine learning is to learn and gain structural information from labeled training data. As such algorithms produce models or functions which able to provide correct predictions about new unlabeled data. Many supervised machine learning algorithms have been developed for classification problems, such as support vector machines, decision trees and neural network (Safavian and Landgrebe 1991, Scholkopf and Smola 2001, Hagan et al. 1996). Although these algorithms show high accuracy in the literature, but they don't have powerful explanation and interpretability of results. Logical Analysis of Data (LAD) is developed and used as a classification method that has competitive accuracy and powerful abilities of interpretability by providing patterns that contain structural information by analyzing the training data (Boros et al. 1997, Boros et al. 2000, Hammer and Bonates 2006). LAD is used as a

powerful classification method in various fields such as medical, services, and manufacturing (Hammer and Bonates 2006, Mortada et al. 2011, Bennane and Yacout 2012, Mortada et al. 2013, Ragab et al. 2015, Ragab et al. 2016, Shaban et al. 2017, Jocelyn et al. 2017, Ragab et al. 2018).

Developing scalable machine learning algorithms which can handle large datasets has attracted the researchers in the machine learning community. A taxonomy was proposed in (Zhou et al. 2017) for methods/platforms for machine learning on big data. In this taxonomy, studies are categorized based on whether any parallelism is considered in the algorithms/platforms or not. Methods in the non-parallelism category aim at providing much faster optimization algorithms that can deal with big data without any parallelism and run more efficiently, with significantly better time complexity and/or space complexity. The parallelism category reflects most of state of the art in scalable machine learning methods (Zhang et al. 2017). To deal with big data that is characterized by huge dimensions in terms of number of features and sample size, methods in this category exploit data geometry in the input or algorithm/model space or both at the same time (Al-Jarrah et al. 2014). Specifically, parallelism methods that make machine learning algorithms more scalable are classified into two sub-categories: (i) data parallelism: partitioning input data vertically, horizontally, or even arbitrarily into manageable pieces, and then computing on all data subsets simultaneously, and (ii) model/parameter parallelism: developing parallelized versions of learning algorithms by first dividing the learning model/parameters and then performing computations on each division concurrently. (Zhou et al. 2017)

Existing machine learning algorithms could use big data processing techniques to achieve scalability. Such efforts could be classified into two categories. In the first category, a general middleware layer that re-implements existing machine learning algorithms is developed (Zhou et al. 2017). For example, Spark MLlib (Sankar and Karau 2015) and Mahout (Owen et al. 2011) are two representative opensource packages that support many scalable machine learning algorithms. Many common learning algorithms, including classification, regression, clustering, collaborative filtering and dimensionality reduction, are included in both Spark MLlib and Mahout. These packages are independent layer that separate front-end algorithm from back-end execution engine, so it can be used on many big data engines, for example Mahout supports Hadoop, Spark and H2O as its big data engines. In the second category, existing learning algorithms are employed to run directly on big data platform as Hadoop or Spark. Many machine learning algorithms can be implemented by MapReduce, including linear regression, k-means, logistic regression, Naive Bayes, etc. (Zhou et al. 2017).

The incremental learning algorithms are developed for learning from streaming data and update the model according to the new extracted knowledge. In (Polikar et al. 2001), an incremental learning algorithm is defined as one that includes the following criteria: (1) Learning new knowledge from the new data. (2) No needing access to the data used to train the existing model. (3) Not suffering from catastrophic forgetting (preserve previously acquired knowledge). (4) It should be able to accommodate new classes when it is available in the new data.

However, current proposed algorithms for implementing LAD are not capable of offering solutions that satisfy the requirements of big data and incremental learning of streaming data. There is no method in the literature for implementing LAD on large scale data sets and for learning incrementally from streaming data. Therefore, we propose an algorithm to merge different LAD models which trained on different data subsets. This algorithm shall be used later as a reducer function in MapReduce paradigm and to learn incrementally from streaming data. In this paper, we explain our proposal and demonstrate that the proposed merging algorithm has competitive accuracy and reduces the computational time of processing large data sets even when it is run in a sequential paradigm. The accuracy of the merging algorithm is demonstrated by using a real industrial data set. Using this algorithm in parallel environment and for incremental learning will be accomplished in future research work.

2. Logical Analysis of Data (LAD)

LAD is a supervised data mining approach which was first proposed in (Crama et al. 1988). It has been considered as an effective classification technique that relies on extracting patterns from two-class binary datasets. These patterns are used as decision rules that classify the data into distinct classes (Moreira 2000).

LAD decision making approach consists of three main steps: data binarization, pattern generation, and theory formation as illustrated in Fig. 1. The step of data binarization intends to convert numerical and nominal data to binary data. Pattern generation is the essential step that identifies the patterns of different classes from the binarized dataset. Theory formation is the final step that uses the generated patterns to create a model called discriminant function. This

model is called LAD model and is used as a classifier for new data. The characteristics of the generated patterns have an essential impact on the accuracy of LAD decision model (Boros et al. 2000). The positive (negative) pattern is a set of literals which are true in at least one positive (negative) observation and false in all negative (positive) observations in the training data set. A literal is a Boolean variable x or its negation $\bar{x}$. Each binary attribute b_i can be represented by the corresponding literal x_i or its negation $\overline{x_i}$, where x_i is used for $b_i = 1$ and $\overline{x_i}$ for $b_i = 0$. The degree of a pattern is the number of its literals. (Boros et al. 2000)

There are four types of patterns; prime, spanned, strong and maximal. A prime pattern is the pattern such that if any of its literals is removed it will be no longer a pattern. A spanned pattern is the pattern that has the maximum number of literals without reducing the number of covered observations. A strong pattern is the pattern which its covered observations is not a subset of another pattern. A maximal pattern is the pattern that has the maximum coverage among all patterns that cover a certain observation in the training data set. (Hammer et al. 2004)

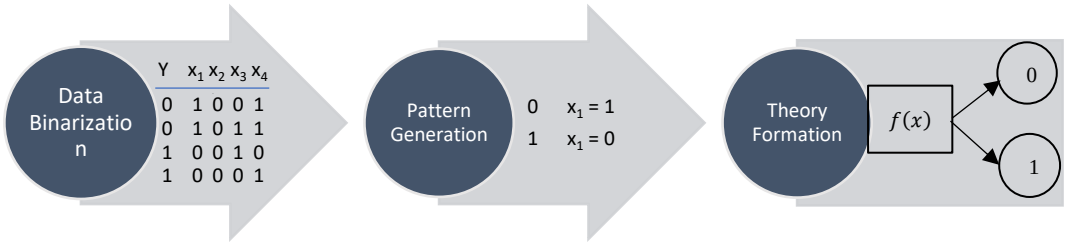

Figure 1. Framework of LAD.

For applying LAD in multi-class ($k \geq 3$) classification problems, where *k* is the number of classes, a decomposition technique is used to divide the multi-class problem into many two-class problems. This technique can be implemented in two different ways; One versus All (OvA) and One versus One (OvO) (Tax and Duin 2002).

OvA approach divides the multi-class problem into k different binary (two class) classification problems. Each problem considers one class i as the positive class and all the remaining $(k-1)$ classes as the negative class. A LAD classifier f_i, where$i \in [1:k]$ is constructed for each problem. By combining all k classifiers, the following multi-class OvA model for a new event/observation x is resulted:

$$\mathrm{f}(x) = \arg\max_i f_i(x), \qquad (1)$$

This model (equation 1) classify the new event x as class i which has the largest value $f_i(x)$. (Tax and Duin 2002).

OvO approach divides the multi-class model into $k(k-1)$ binary classification problems by considering each possible class pairs as an individual binary classification problem. Each problem consider class $i \in [1:k]$ as a positive class and $j \in [1:k]$ as a negative class where i ≠ j. By combining all classifiers f_{ij} , the following multi-class OvO model for a new event x is resulted

$$\mathrm{f(x)} = \arg\max_i \sum_{i \neq j} f_{ij}(x) \qquad (2)$$

This model (equation 2) classifies the new event x as class i which has the largest value $\sum_{i \neq j} f_{ij}(x)$ (Tax and Duin 2002).

Pattern generation step is very expensive in terms of computational time. The challenge is to generate patterns that enhance the accuracy of LAD decision model in such a way that minimizes the computational time while handling big data sets. Many approaches are used to generate patterns from the training data set. Most of these approaches are based on enumeration, heuristic algorithms or integer programming techniques. In this research, we use cbmLAD software (Dexin, 2017) which is based on Ant Colony heuristic technique to provide maximal patterns.

3. Merging of LAD models

3.1. Notation

In the case of two-classes data set, the training set is $\Omega = \Omega^+ + \Omega^-$ with n features. The data set is partitioned into two sub sets $\Omega_1 = \Omega_1^+ + \Omega_1^-$ and $\Omega_2 = \Omega_2^+ + \Omega_2^-$, where $\Omega = \Omega_1 \cup \Omega_2$. Both of Ω_1 and Ω_2 have same n features of Ω. If LAD is applied on each subset Ω_1 and Ω_2 separately, two patterns sets will be produced independently; $\Pi_1 = \{P_{11}, \dots, P_{1r_1}\}$ and $\Pi_2 = \{P_{21}, \dots, P_{2r_2}\}$, where r_1 and r_2 are the numbers of patterns generated from applying LAD on Ω_1 and Ω_2, respectively. Each pattern set has positive and negative patterns. Each P_{ij} is an n – dimensional hypercube that covers at least one observation from one of the sets Ω_i^+ or Ω_i^- and no observations from the other set Ω_i^- or Ω_i^+, respectively. i is a subscript to identify the data subset and j is a subscript to identify the pattern. Partitioning the data set Ω into Ω_1 *and* Ω_2 results in generating pure patterns, but for the sake of simplicity in implementing the merging operation, we will relax the purity condition of the generated pattern to allow for generating non-pure patterns. The union of patterns $(P_{i1} \cup \dots \cup P_{ir_i})$ in each set of patterns Π_i covers all the n – dimensional space of data subset Ω_i.

Each P_{ij} has characteristics which come from the observations covered by it. In the merging operation discussed in section 3.2, these characteristics will be transferred to the new merged patterns directly or indirectly. We illustrate these characteristics as follows; (1) C_{ij} is the class (the sign positive or negative) which is assigned to pattern j, (2) Cov_{ij} is the number of observations from Ω_i covered by pattern j, (3) Cov_{ij}^+ is the number of positive observations from Ω_i^+ covered by pattern j, (4) Cov_i^- is the number of negative observations from Ω_i^- covered by pattern j, (5) χ_{ij} is the homogeneity of pattern P_{ij}; $\chi_{ij} = Cov_{ij}^+/Cov_{ij}$ if C_{ij} is positive or $\chi_{ij} = Cov_{ij}^-/Cov_{ij}$ if C_{ij} is negative, (6) ω_{ij} is the weight of the pattern P_{ij}; $\omega_{ij} = Cov_{ij}^+/\sum_{k=1}^{r_i} Cov_{ik}^+$ if C_{ij} is positive and $\omega_{ij} = Cov_{ij}^-/\sum_{k=1}^{r_i} Cov_{ik}^-$ if C_{ij} is negative, (7) V_{ij} is n–volume of the hypercube (pattern) P_{ij}, (8) V_{ij}^r is the relative n–volume of the hypercube; $V_{ij}^r = V_{ij}/\sum_{C_{ik}=C_{ij}} V_{ik}$, (9) D_{ij}^r is the relative density of the pattern; $D_{ij}^r = \omega_{ij}/V_{ij}^r$ and (10) D_{ij} is the density; $D_{ij} = \omega_{ij}/V_{ij}$. We use the weight and the relative volume to determine the relative density. The relative density is used to reduce the impact of unbalanced data with the number of observations or the space of data that is reserved by each class.

3.2. Merging Operation of two LAD models

The proposed merging operation of two LAD models is explained in this section. The operation merges two patterns sets; Π_1 and Π_2 which are extracted from applying LAD on two different subsets of data Ω_1 and Ω_2 into one merged interpretable set of patterns Π_m as illustrated in Fig. 2.

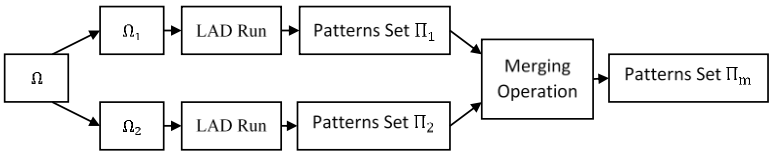

Figure 2. Schematic of running LAD on two different data subsets and merging results into one result.

The operation consists of two main steps; determining the intersection between each pair of patterns and computing the characteristics of the new intersecting patterns.

Step 1: Determine intersecting patterns

For each pair of patterns (P_{1i}, P_{2j}), with $P_{1i} \in \Pi_1$, $P_{2j} \in \Pi_2$, $i \in [1, \dots, r_1]$ and $j \in [1, \dots, r_2]$, the intersection pattern $P_{mk} = P_{1i} \cap P_{2j}$ is computed and added to the merged patterns set Π_m. The steps of finding this intersection is explained in the following paragraphs. If there is no intersection or there is just partial intersection between a pattern P_{1i} with any pattern from Π_2, the pattern P_{1i} or the remaining part of it will be added to the merged patterns set Π_m. By completing this process, the merging patterns set Π_m will cover the n – dimensional space of data subsets Ω_1 and Ω_2.

The algorithm proposed in (Andrzejak et al. 2013) is used in our work to determine the intersection between each two patterns. Andrzejak et al. 2013 propsoed a merging algorithm to determine the intersection between two decision trees after converting them to sets of boxes. This algorithm uses a line sweep technique to be run on $O(n(|\Pi_1| + |\Pi_2|)\, log(|\Pi_1| + |\Pi_2|) + |S|)$. Where *n* is the number of features, and *S* is the pairs of patterns that are intersected.

For each feature t, the algorithm determines a list E_t to contain the boundaries of feature t for each pattern in Π_1 and each pattern in Π_2. Each entry in E_t is annotated by its origin information (e.g. its pattern, pattern set Π_1 or Π_2, starting or ending boundary).

After sorting E_t list, the algorithm starts to sweep the entries and update an active patterns list by entering a new pattern to it if the entry is a starting boundary and remove the pattern from it if the entry is an ending boundary on feature *t*. During the line sweeping process, a list S_t is being developed to contain the pairs of patterns from Π_1 and Π_2 that overlap on the attribute t. The intersection list $S = S_1 \cap \ldots \cap S_n$ contains the intersected pairs of patterns which are overlapped on all n features. Merging patterns set Π_m is determined from the intersections of each pair of patterns in the list *S*.

Step 2: Computing the characteristics of the new intersecting patterns

In this step, the characteristics of each pattern in the merging patterns set Π_m will be determined based on its parent patterns from Π_1 and Π_2. Let $P_{m(j,k)} \in \Pi_m$ is a merged pattern. $P_{1j} \in \Pi_1$ *and* $P_{2k} \in \Pi_2$ are the parent patterns where $P_{1j} \cap P_{2k} = P_{m(j,k)}$. The n – volume of the merged pattern is the n – volume of the intersection between the parent patterns. There will be two cases; the first case is that the parent patterns have the same class and the second one is that they have different classes. In the first case, the class of the merged pattern will be the same class of its parent patterns. To determine the coverage of the merged pattern, we introduce the following assumption; the observations that are covered by any pattern have a uniform distribution on the space bounded by this pattern. According to this assumption, any portion of a pattern covers a number of observations is computed as shown in equation (3). The coverage of the merged pattern $P_{m(j,k)}$ will be determined using the same way. The coverage of the intersected portion of each parent pattern (P_{1j} and P_{2k}) will be determined and then will be added together as shown in equation (4). In the second case, there is a confliction between the parent patterns P_{1j} and P_{2k}. The class of the parent pattern that has the higher relative density will be assigned to the merged pattern $P_{m(j,k)}$.

$$\mathrm{Cov}_{\mathrm{portion}} = \frac{V_{portion}}{V_{pattern}} \times \mathrm{Cov}_{\mathrm{pattern}} \qquad (3)$$

$$\mathrm{Cov}_{m(\mathrm{j,k})} = \frac{\mathrm{V}_{m(\mathrm{j,k})}}{\mathrm{V}_{1\mathrm{j}}} \times Cov_{1\mathrm{j}} + \frac{\mathrm{V}_{m(\mathrm{j,k})}}{V_{2k}} \times Cov_{2k} \qquad (4)$$

Simple numerical example

For more explanation of the merging operation, we generated a two-classes data set $\Omega = \Omega^+ + \Omega^-$ with two features (X and Y) to explain the proposed operation. We partitioned the data set into two data subsets $\Omega_1 = \Omega_1^+ + \Omega_1^-$ and $\Omega_2 = \Omega_2^+ + \Omega_2^-$, where $\Omega = \Omega_1 \cup \Omega_2$. The generated data set and the two data subsets are illustrated in Fig. 3. Each data subset was processed by LAD separately. The two LAD processes provided two patterns sets $\Pi_1 = \{P_{11}, P_{12}\}$ and $\Pi_2 = \{P_{21}, P_{22}\}$ from Ω_1 and Ω_2 respectively as shown in Fig. 4.a and Fig. 4.b. The characteristics of the patterns are determined and illustrated in table 1. The generated patterns by classical LAD is showed in Fig. 4.d.

The first step of the merging operation is to determine the intersections between each pair of patterns from Π_1 and Π_2. These intersections are the new patterns in the merging set. As shown in Fig. 4.c, there are three new patterns are generated from three intersections; $P_{m(1,1)} = P_{11} \cap P_{21}$, $P_{m(1,2)} = P_{11} \cap P_{22}$ and $P_{m(2,2)} = P_{12} \cap P_{22}$. It is obvious that no intersection exists between the pair (P_{12}, P_{21}); $P_{12} \cap P_{21} = \emptyset$. In case of the pattern $P_{m(1,1)}$, there is no confliction between the parent patterns P_{11} and P_{21}. The coverage $Cov_{m(1,1)}$ is computed by using equation (4):

$$Cov_{m(j,k)} = \frac{V_{m(1,1)}}{V_{11}} \times Cov_{11} + \frac{V_{m(1,1)}}{V_{21}} \times Cov_{21} = \frac{0.5}{0.5} \times 70 + \frac{0.5}{0.6} \times 30 = 88.3$$

Feature		Value
# of attributes		2
# of classes		2
# of observations of Ω	Ω^+	100
	Ω^-	100
# of observations of Ω_1	Ω_1^+	70
	Ω_1^-	30
# of observations of Ω_2	Ω_2^+	30
	Ω_2^-	70

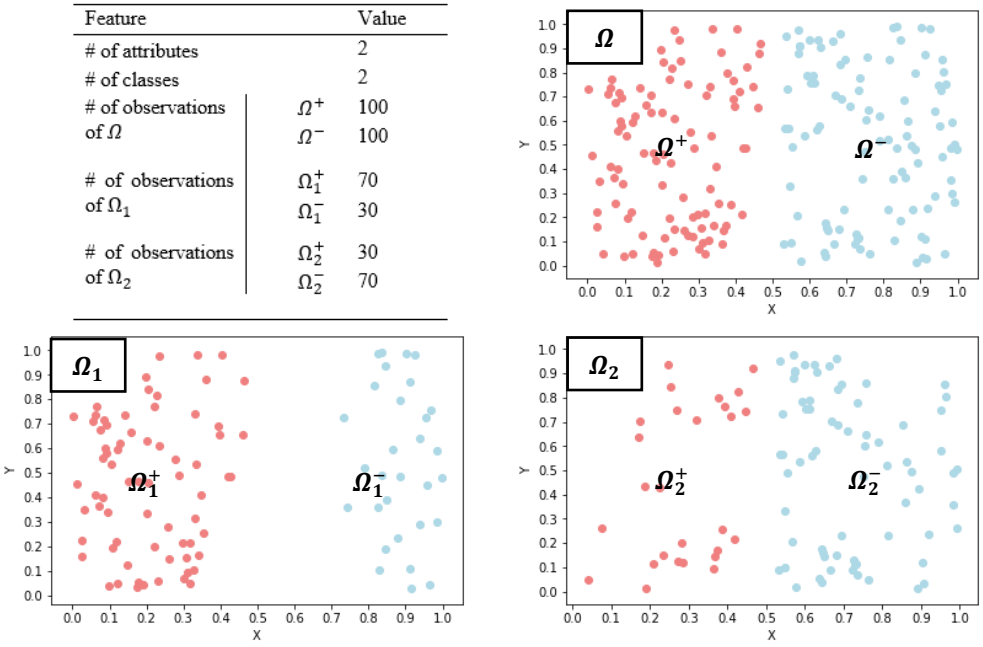

Figure 3. Generated data set Ω and two data subsets Ω_1 and Ω_2

In case of pattern $P_{m(1,2)}$, a confliction between the parent patterns exists. As the relative densities of the parent patterns P_{11} and P_{22} are equal as shown in table 1, we relax and use the density D in this case instead of the relative density D^r. So, the negative class of pattern P_{22} is assigned to the merging pattern $P_{m(1,2)}$ as the density of P_{22} ($D_{22} = 2$) is higher than the density of P_{11} ($D_{11} = 2$). The coverage $Cov_{m(1,2)}$ is computed as the previous pattern. The last merging pattern $P_{m(2,2)}$ doesn't have confliction between its parent patterns P_{12} and P_{22}. The characteristics of the all three merging patterns are computed and illustrated in table 2.

Table 1. Characteristics of the patterns in Π_1 and Π_2 sets.

Pattern	Class	Cov	Cov^+	Cov^-	χ	ω	V	V^r	D^r	D
P_{11}	Positive	70	70	0	1	1	0.6	1	1	1.66
P_{12}	Negative	30	0	30	1	1	0.4	1	1	2.5
P_{21}	Positive	30	30	0	1	1	0.5	1	1	2
P_{22}	Negative	70	0	70	1	1	0.5	1	1	2

Table 2. Characteristics of the merging patterns.

Pattern	Confliction	Class	Cov	Cov^+	Cov^-	χ	ω	V	V^r	D^r	D
$P_{m(1,1)}$	No	Positive	88.3	88.3	0	1	1	0.5	1	1	2
$P_{m(1,2)}$	Yes	Negative	25.7	11.7	14	0.54	0.14	0.1	0.2	0.7	1.4
$P_{m(2,2)}$	No	Negative	86	0	86	1	0.86	0.4	0.8	1.075	2.15

3.3. Merging Operation for incremental learning and big data

In case of the incremental learning, streaming data are loaded in chunks. The first chunk of the streaming data will be analyzed by LAD to generate first patterns set Π_1 and the first model (classifier). Then the second chunk will be analyzed by LAD to generate the second patterns set Π_2. By using the merging operation of LAD discussed in the previous section, the first two sets of patterns will be merged to produce the first merging patterns set Π_{m1} and update

the LAD model using this set. Each new chunk of the streaming data will generate a new set of patterns which will be merged with the current one as illustrated in Fig. 5.

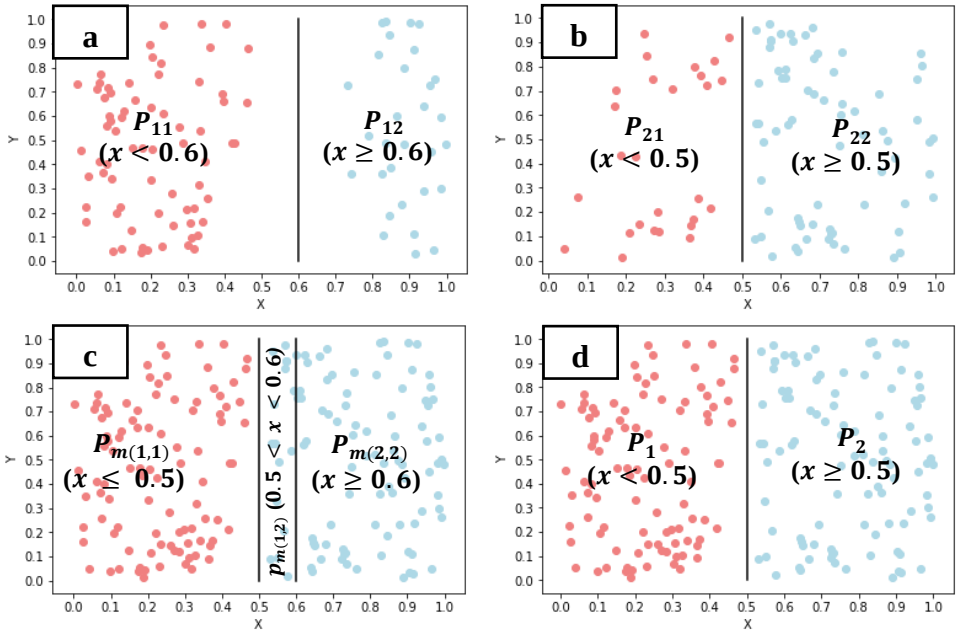

Figure 4. a: patterns set Π_1, b: patterns set Π_2, c: merging patterns set Π_m and d: the generated patterns by classical LAD

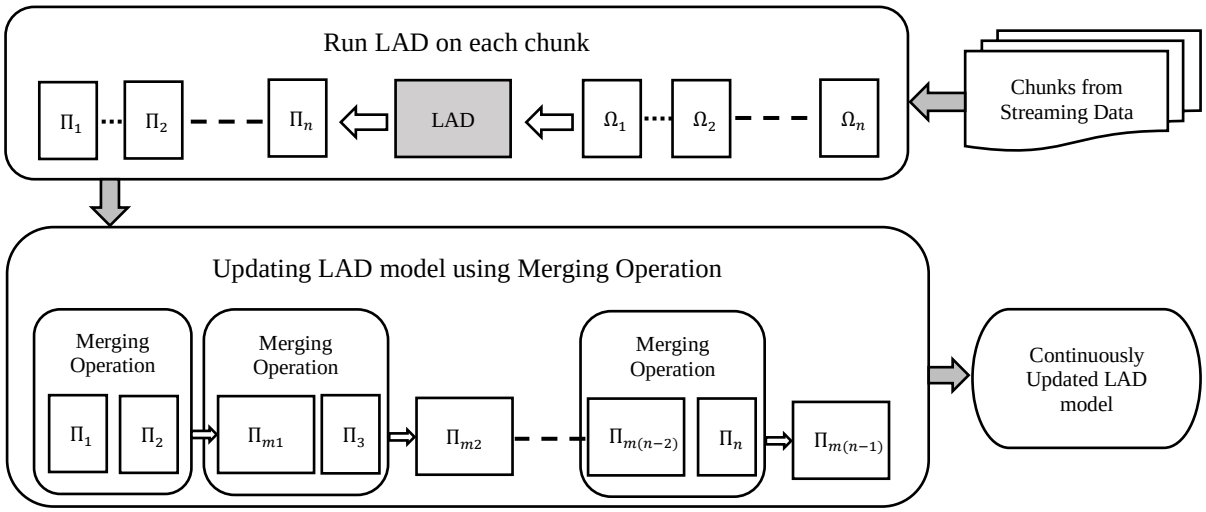

Figure 5. Framework of Incremental LAD using the merging operation

In case of big data set, data set Ω will be divided into n data subsets $[\Omega_1, \dots, \Omega_n]$. Each subset will be analyzed by LAD individually and provide a corresponding set of patterns. Sets of patterns $[\Pi_1, \dots, \Pi_n]$ will be generated, then will be merged together on a pairwise mechanism as illustrated in Fig. 6 till it ends with one general set of patterns. This framework is ideal to be used in distributed processing environments.

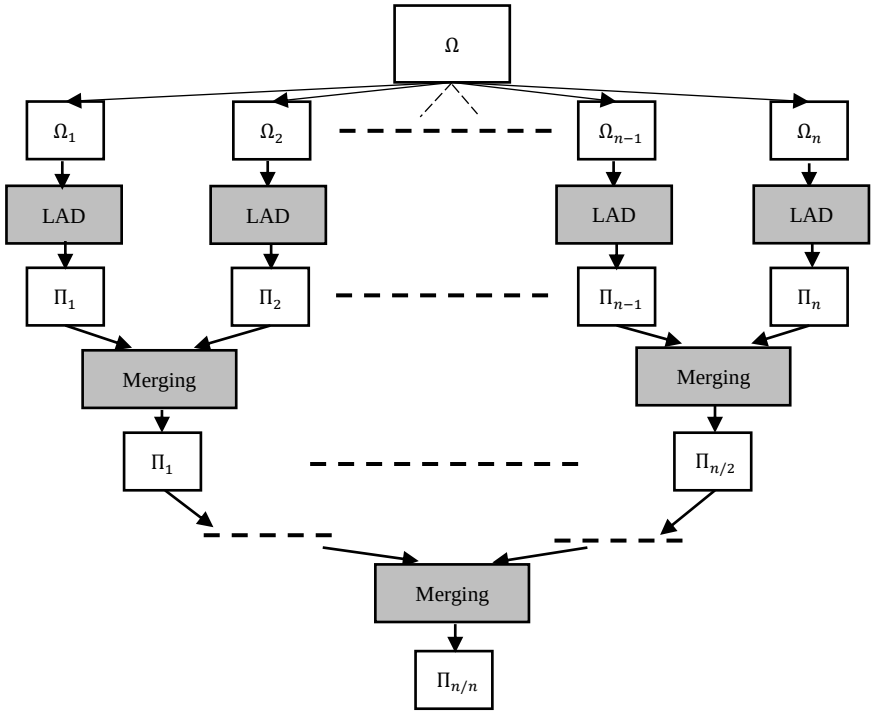

Figure 6. Framework of processing big data set Ω with LAD using Merging Operation

4. Numerical Experiments

We conducted a numerical experiment to evaluate the performance of the merging operation using OvO multi-class LAD. We expand the merging operation to deal with OvO multi-class LAD by running it $k(k-1)/2$ times; one for each two pairs of classes, where k is the number of classes. The numerical experiment is performed as follows.

4.1. Design of Experiment

In our experiment, we use an industrial dataset which is provided by a research industrial partner in the chemical field. The data includes six different operation modes. The object is to learn from the historical data to detect which mode is currently online. The data set is summarized in Table 3. The merging operation is implemented sequentially in this research work. The parallel implementation is still in the developing state. The *K*-Fold method is used with $k = 5$ to determine the average results from five different training data subsets and five corresponding testing data subsets. In our analysis, we compare between running classic LAD using cbmLAD software (Dexin, 2017) on the whole training data subsets and our proposed merging operation of LAD. For the merging operation, we load the training data subset to cbmLAD chunk by chunk. Every chunk has 5×10^3 observations. Each chunk is processed separately by cbmLAD and patterns sets are gathering in patterns' sets for the next step. The merging operation runs on pairs of sets of patterns and provides merging patterns sets which are merged together until we have one final merging set as illustrated in Fig. 6. So, the merging operation is repeated $n\ log\ n$ times, where, n is the number of sets which are provided to run LAD on the chunks. If any chunk has only one class, LAD will provide an empty set of patterns. We then evaluate the quality and the classification accuracy of the final set of patterns using the testing data sets. The quality will be evaluated in terms of degree, weight and homogeneity of the pattern.

4.2. Experimental Results

First, we compare the patterns set produced by running classic LAD on the whole data, and the final patterns set produced from merging operations. The comparison is illustrated in Table 4 by averaging the results from the 5 folds. Because this data set has six classes, the OvO multi-class type has thirty positive sets as shown in table 5. The merging operation of LAD produces higher number of patterns in most of the sets. The average degree of patterns produced by

merging operation is higher than those produced by one run of LAD on the whole data. It is obvious that the average weights of the patterns produced by the merging operation are lower than those produced by one run of LAD, especially when the number of patterns is high such as in the cases of the two pattern sets 1v6 and 6v1. This is because of distributing the coverage to many patterns. But we must consider the maximum weight which is not very low regarding to the average; 45% and 37% for 1v6 and 6v1 sets respectively. This observation reveals that there are some dominant patterns which have high weights. In addition, we relaxed the purity condition on the merging operations that transfers some coverage to opposite class patterns and reduces the weights of the patterns.

Table 3. Data Summary

Feature		Value
# of attributes		8
# of classes		6
# of observations	Class 1	2099
	Class 2	517
	Class 3	4828
	Class 4	744
	Class 5	459349
	Class 6	830
	Total	468367

Second, we compare the performance of the sets of patterns in term of the accuracy for each class. The average accuracy is determined by performing classifiers that are produced by the whole data run of classic LAD and the merging operation classifiers on the five testing data subsets. The comparison is illustrated in Fig. 7. This comparison reveals that the merging operation performs well in this data set. The worst accuracy is the class six accuracy which is reduced from 98.6% to 93.8%, That is, about 5% reduction from the whole data run of classic LAD.

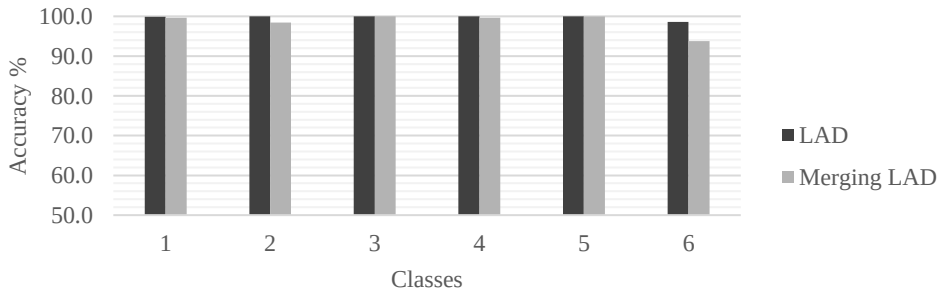

Figure 7. Accuracy comparison between LAD and merging operation

Table 4 shows the CPU runtime of performing the whole data run of classic LAD and the proposed merging operation on each fold of the data set. The computing platform was an Intel i5-3320M 2.60 GHz processor with 12 GB RAM. Although the merging operation was performed sequentially and not on parallel environment, the reduction of the computational time is 87%.

Table 4. Comparison of CPU Time (Minutes) between sole run of LAD and merging operation

Fold	LAD	Merging LAD	Reduction %
1	30.37	4.62	84.79
2	30.50	3.88	87.28
3	29.70	4.00	86.53
4	29.72	2.83	90.48
5	29.80	4.37	85.34
Average	30.02	3.78	87.41

Table 5. Comparison of patterns produce by LAD and Merging Operation of LAD

Patterns set	Patterns produced by LAD (sole run on the whole data set)								Patterns produced by Merging Operation of LAD									
	# of patterns	Degree			Weight (ω) %			Homogeneity %	# of patterns	Degree			Weight (ω) %			Homogeneity %		
		min	average	max	min	average	max			min	average	max	min	average	max	min	Average	max
1v2	1	1	1	1	100	100	100	100	2.8	2	2.1	2.2	22.5	35.3	48.0	99.4	99.6	100
1v3	1	1	1	1	100	100	100	100	1	1	1	1	100	100	100	100	100	100
1v4	1	1	1	1	100	100	100	100	1	1	1	1	100	100	100	100	100	100
1v5	1	1	1	1	100	100	100	100	1	1	1	1	100	100	100	100	100	100
1v6	8	1	2.65	3.8	1.0	17.0	56.0	100	73.4	1.2	4.32	6.6	0.08	3.5	45.0	33.4	97.6	100
2v1	1	1	1	1	100	100	100	100	3.2	1	1.8	2.6	53.4	67.0	99.4	48.6	61.4	84.3
2v3	2	1	2.4	3.4	65.0	81.0	98.0	100	8.4	1	4.26	6.4	20.9	50.7	97.9	50.2	70.4	96.9
2v4	1	1	1	1	100	100	100	100	1	1	1	1	100	100	100	87.5	87.5	87.5
2v5	1	1	1	1	100	100	100	100	2.2	1.6	2	2.4	7.7	27.3	59.1	25.2	55.1	88.5
2v6	1.6	1	2	3	55.0	77.0	99.0	100	3.8	1.8	2.76	3.8	44.7	56.7	98.3	58.7	73.6	94.6
3v1	1	1	1	1	100	100	100	100	1	1	1	1	100	100	100	100	100	100
3v2	1	2	2	2	100	100	100	100	5.4	2	4.46	6.2	1.0	8.8	54.0	100	100	100
3v4	1	1	1	1	100	100	100	100	2	1	1.5	2	17.3	44.3	98.3	99.0	99.3	99.8
3v5	1	1	1	1	100	100	100	100	4.2	1	1.99	2.8	0.6	13.7	79.8	60.2	78.6	97.2
3v6	1	1	1	1	100	100	100	100	3.2	1.6	1.82	2.4	60.8	71.8	90.3	99.1	99.5	99.8
4v1	1	1	1	1	100	100	100	100	1	1	1	1	100	100	100	100	100	100
4v2	1	1	1	1	100	100	100	100	1.8	1	1.4	1.8	60.8	73.2	79.4	92.0	93.9	97.8
4v3	1	1	1	1	100	100	100	100	1	1	1	1	99.8	99.8	99.8	91.3	91.3	91.3
4v5	1	1	1	1	100	100	100	100	1.8	1.6	1.7	1.8	13.5	35.2	69.1	45.7	70.8	99.8
4v6	1	1	1	1	100	100	100	100	1.4	1	1.2	1.4	89.9	89.9	90.0	95.0	96.6	99.7
5v1	1	1	1	1	100	100	100	100	1	1	1	1	100	100	100	100	100	100
5v2	1	1	1	1	100	100	100	100	2.2	1.8	1.9	2	19.2	28.5	37.5	44.0	69.8	95.8
5v3	1	1	1	1	100	100	100	100	2.4	1.8	1.88	2	50.7	58.1	65.5	76.7	83.3	90.0
5v4	1	1	1	1	100	100	100	100	1.8	1.6	1.7	1.8	43.0	54.4	65.7	48.8	72.3	95.8
5v6	1	1	1	1	100	100	100	100	2.4	1	1.55	2	44.9	50.9	61.8	86.4	94.4	99.1
6v1	9.2	1	2.91	4.4	0.02	0.15	0.36	100	58.6	1	3.39	6	0.2	2.8	37.0	6.9	87.5	100
6v2	2	1.8	1.8	1.8	0.49	0.76	0.97	100	4.8	1.8	2.60	3.6	17.3	33.0	55.7	96.2	98.6	100
6v3	1	1	1	1	1	1	1	100	2.6	1.4	2.05	2.8	59.2	64.7	73.4	54.9	78.2	94.0
6v4	1	1	1	1	1	1	1	100	1.3	1	1.1	1.2	81.7	86.1	95	76.9	80.0	86.3
6v5	1	1	1	1	1	1	1	100	2.6	1	1.53	2	35.9	43.1	54.2	49.0	56.7	71.8

5. Conclusion

In this paper, we developed an efficient merging operation for different sets of patterns which are produced by performing LAD on different data subsets. The motivation behind this operation is developing incremental learning framework using LAD and allowing LAD to process big data sets in parallel computing environments. We evaluated the proposed technique in terms of pattern's quality, accuracy and computational time. Overall, the quality of the patterns is reduced in terms of the degrees and weights comparing to the patterns produced by a single run of LAD, but the accuracy and the computational time are promising to apply the proposed technique in an incremental learning platform and parallel computing environments. In the future research, we plan to modify the merging operation for the incremental learning platform by adding some tuning parameters to reduce the noise of new chunks of data until building a robust model. In addition, LAD will be implemented on different parallel computing systems using this merging operation and will be evaluated using various data sets.

References

Al-Jarrah, O.,Siddiqui, A.,Elsalamouny, M.,Yoo, P. D.,Muhaidat, S. and Kim, K., Machine-learning-based feature selection techniques for large-scale network intrusion detection, *2014 IEEE 34th International Conference on Distributed Computing Systems Workshops (ICDCSW)*, 2014.

Andrzejak, A.,Langner, F. and Zabala, S., Interpretable Models from Distributed Data via Merging of Decision Trees, *2013 Ieee Symposium on Computational Intelligence and Data Mining (Cidm)*, 2013.

Bennane, A. and Yacout, S., LAD-CBM; new data processing tool for diagnosis and prognosis in condition-based maintenance, *Journal of Intelligent Manufacturing* vol. 23, pp: 265-275, 2012.

Boros, E.,Hammer, P. L.,Ibaraki, T. and Kogan, A., Logical analysis of numerical data, *Mathematical Programming* vol. 79, pp: 163-190, 1997.

Boros, E.,Hammer, P. L.,Ibaraki, T.,Kogan, A.,Mayoraz, E. and Muchnik, I., An implementation of logical analysis of data, *Ieee Transactions on Knowledge and Data Engineering* vol. 12, pp: 292-306, 2000.

Crama, Y.,Hammer, P. L. and Ibaraki, T., Cause-effect relationships and partially defined Boolean functions, *Annals of Operations Research* vol. 16, pp: 299-325, 1988.

Dexin, cbmLAD software, 2017.

Hagan, M. T.,Demuth, H. B.,Beale, M. H. and De Jesús, O. *Neural network design*. Volume 20: Pws Pub. Boston, 1996.

Hammer, P. L. and Bonates, T. O., Logical analysis of data—An overview: From combinatorial optimization to medical applications, *Annals of Operations Research* vol. 148, pp: 203-225, 2006.

Hammer, P. L.,Kogan, A.,Simeone, B. and Szedmák, S., Pareto-optimal patterns in logical analysis of data, *Discrete Applied Mathematics* vol. 144, pp: 79-102, 2004.

Jocelyn, S.,Chinniah, Y.,Ouali, M.-S. and Yacout, S., Application of logical analysis of data to machinery-related accident prevention based on scarce data, *Reliability Engineering & System Safety* vol. 159, pp: 223-236, 2017.

Moreira, L. M. 'The use of Boolean concepts in general classification contexts,' in *Book The use of Boolean concepts in general classification contexts*, edited by Editor. City: EPFL, 2000.

Mortada, M.-A.,Yacout, S. and Lakis, A., Diagnosis of rotor bearings using logical analysis of data, *Journal of Quality in Maintenance Engineering* vol. 17, pp: 371-397, 2011.

Mortada, M.-A.,Yacout, S. and Lakis, A., Fault diagnosis in power transformers using multi-class logical analysis of data, *Journal of Intelligent Manufacturing* vol. 25, pp: 1429-1439, 2013.

Owen, S.,Anil, R.,Dunning, T. and Friedman, E. *Mahout in action*. Manning Puplications Co, 2011.

Polikar, R.,Upda, L.,Upda, S. S. and Honavar, V., Learn++: An incremental learning algorithm for supervised neural networks, *IEEE transactions on systems, man, and cybernetics, part C (applications and reviews)* vol. 31, pp: 497-508, 2001.

Ragab, A.,El-Koujok, M.,Poulin, B.,Amazouz, M. and Yacout, S., Fault diagnosis in industrial chemical processes using interpretable patterns based on Logical Analysis of Data, *Expert Systems with Applications* vol. 95, pp: 368-383, 2018.

Ragab, A.,Ouali, M.-S.,Yacout, S. and Osman, H., Remaining useful life prediction using prognostic methodology based on logical analysis of data and Kaplan–Meier estimation, *Journal of Intelligent Manufacturing* vol. 27, pp: 943-958, 2016.

Ragab, A.,Yacout, S.,Ouali, M.-S. and Osman, H., Multiple failure modes prognostics using logical analysis of data, *2015 Annual Reliability and Maintainability Symposium (RAMS)*, 2015.

Safavian, S. R. and Landgrebe, D., A survey of decision tree classifier methodology, *IEEE transactions on systems, man, and cybernetics* vol. 21, pp: 660-674, 1991.

Sankar, K. and Karau, H. *Fast data processing with Spark*. Packt Publishing Ltd, 2015.

Scholkopf, B. and Smola, A. J. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.

Shaban, Y.,Yacout, S.,Balazinski, M. and Jemielniak, K., Cutting tool wear detection using multiclass logical analysis of data, *Machining Science and Technology* vol. 21, pp: 526-541, 2017.

Tax, D. M. and Duin, R. P., Using two-class classifiers for multiclass classification, *Object recognition supported by user interaction for service robots*, 2002.

Zhang, Y.,Ren, S.,Liu, Y. and Si, S., A big data analytics architecture for cleaner manufacturing and maintenance processes of complex products, *Journal of Cleaner Production* vol. 142, pp: 626-641, 2017.

Zhou, L.,Pan, S.,Wang, J. and Vasilakos, A. V., Machine learning on big data: Opportunities and challenges, *Neurocomputing* vol. 237, pp: 350-361, 2017.

Biographies

Osama Elfar is a Ph.D. student at Mathematical and Industrial Engineering Department, Ecole Polytechnique de Montreal, Canada. He earned bachelor's and master's degrees in Mechanical Design and Manufacturing Engineering from Cairo University, Egypt. His research interests include machine learning, data mining, big data processing, intelligent maintenance systems, and diagnosis and prognosis systems.

Soumaya Yacout is full Professor in the Department of Mathematics and Industrial Engineering at Polytechnique Montreal in Canada since 1999. She is also the founder, President and CEO of DEXIN Inc. , an enterprise dedicated in offering state of the art technologies for data-driven solutions to help companies in achieving the highest level of value added performance by keeping their physical assets in good health. She earned her doctoral degree in Operations Research at The Georges Washington University in 1985, her bachelor degree in Mechanical Engineering in 1975, and her masters in Industrial Engineering in 1979, at Cairo University. She designed and taught courses on quality engineering, reliability and maintenance, and discrete event simulation for undergraduate, graduate, and professional engineers in Canada and internationally. Her research interests include preventive, predictive and prescriptive maintenance and optimization of decision-making. She has publications in peer-reviewed journals including Quality Engineering, International Journal of Production Research, Computers and Industrial Engineering, IEEE Transactions, Journal of Intelligent Manufacturing, Expert Systems with Applications, and papers in international conferences, some of which received the best paper award. She is the co-editor and the co-writer of a book 'Current Themes in Engineering Technologies' on minimal repair, and the book 'Ontology Modeling in Physical Asset Integrity Management' on interoperability and exchangeability of data. She is a senior member of the American Society for Quality ASQ, and the Canadian Operations Research Society CORS. She is a Registered Professional Engineer in Quebec. http://www.polymtl.ca/expertises/en/yacout-soumaya

Hany Osman received the Ph.D. degree in industrial engineering from Concordia University, Montréal, QC, Canada, in 2011. He is an Assistant Professor with the College of Computer Sciences and Engineering, King Fahd University of Petroleum and Minerals, KSA. His research interests include data mining, operations research, and operations management. Specifically, his research focuses on investigating problems related to knowledge extraction from datasets, scheduling, inventory management, and transfer line balancing. He has developed efficient algorithms by using decomposition techniques and nature-inspired metaheuristics.

Said Berriah is leading and supervising for more than a decade the research and development at R2 inc (Montréal Canada). Co-inventor of more than 20 patents in the field of monitoring industrial electrolysis cells using data science and machine learning. His research interests cover the monitoring and control of chlor alkali electrolysis cells and PEM cells, machine learning, time series analysis, decision making, artificial intelligence and deep learning. He received his M.Sc.A in Electrical engineering and Intelligent Systems from Université de Sherbrooke (Québec, Canada) in 2000 and since then achieved several proficiency tracks in Machine Learning and Data Science such as the Certificate on Complex Systems Modeling from the Massachusetts Institute of Technology (Boston USA, Summer 2012). He is a professional engineer licenced from the order of engineers of Quebec.