Computing (2019) 101:1687–1710
https://doi.org/10.1007/s00607-018-0683-9

A distributed, proactive intelligent scheme for securing quality in large scale data processing

Kostas Kolomvatsos[1]

Received: 15 February 2018 / Accepted: 22 November 2018 / Published online: 1 December 2018

Abstract
The involvement of numerous devices and data sources in the current form of Web leads to the collection of vast volumes of data. The advent of the Internet of Things (IoT) enhances the devices to act autonomously and transforms them into information and knowledge producers. The vast infrastructure of the Web/IoT becomes the basis for producing data either in a structured or in an unstructured way. In this paper, we focus on a distributed scheme for securing the quality of data as collected and stored in multiple partitions. A high quality is achieved through the adoption of a model that identifies any change in the accuracy of the collected data. The proposed scheme determines if the incoming data negatively affect the accuracy of the already present datasets and when this is the case, it excludes them from further processing. We are based on a scheme that also identifies the appropriate partition where the incoming data should be allocated. We describe the proposed scheme and present simulation and comparison results that give insights on the pros and cons of our solution.

Keywords Data quality · Data accuracy · Data partitions

Mathematics Subject Classification 68U99 · 94D05

1 Introduction

1.1 Motivation

With the advent of the new form of the Web and the Internet of Things (IoT), one can observe increased volumes of data produced by current applications in various domains. Numerous devices generate large scale data that demand intelligent mech-

✉ Kostas Kolomvatsos
kostas.kolomvatsos@glasgow.ac.uk

[1] Department of Computing Science, University of Glasgow, Lilybank Gardens, Glasgow G12 8RZ, UK

anisms for their processing. Usually, data are not structured making their processing more difficult. Unstructured data cannot be easily managed while their quality is limited. They are heterogeneous and variable in nature coming in many formats e.g., text, document, image, video and more [8]. Such kind of data do not follow a pre-defined data model or schema.

According to Eurostat [10], data quality refers in the following aspects: (a) the characteristics of the statistical measurements on top of data; (b) the perception of the statistical measurements by the user; and (c) some characteristics of the statistical process. A metric/dimension, among others, that depicts data quality is accuracy [20]. As stated in [7], consistency and accuracy are the two central criteria for data quality. Accuracy refers to the closeness of estimates to the (unknown) exact or true values [22]. In other words, accuracy depicts the error between the observation/estimation and the real data. As accuracy refers in the closeness of data, it may also depict their 'solidity'. In this paper, we consider that a 'solid' dataset will exhibit a high accuracy that is realized when the error/difference between the involved data is low. In a 'solid' dataset, the standard deviation of the data will be limited. Researchers have identified that accuracy is significant for the responses delivered to queries defined by users or applications. Efficient response plans, for each type of queries, may be defined when the accuracy of the underlying data is secured.

As noted, numerous devices produce huge volumes of data (e.g., terabytes, petabytes), thus, the usual approach is their separation into a number of partitions to facilitate their management. The separation of data could be imposed by the application domain (e.g., the application requires fast responses, thus, we may split the data to process them) or data could be reported by different streams in different locations. The number of partitions depends on the adopted separation technique (e.g., [13,26,28,34,35]) or the locations where the data are collected. The optimal partitioning of a dataset has already been investigated by the research community to deliver the optimal number of partitions when a dataset should be separated [15]. Partitions may be present in various locations, even around the World and they are stored in a set of servers. Data separation facilitates the parallel processing, however, a mechanism for data coordination is imperative. This distributed nature of the described approach imposes new challenges in data processing. Multiple data partitions are available and knowledge should be derived on top of this ecosystem. Our work is motivated by a scenario where multiple streams report data to a set of partitions requiring real time processing. Each data partition should be characterized by high accuracy (i.e., it should be a 'solid' dataset). Our mechanism tries to keep similar data to the same partitions to reduce the error/distance between them, thus, increase the accuracy. Actually, 'solid' datasets is the target of data separation algorithms as adopted by the research community. Such algorithms aim to provide a number of small non-overlapping datasets and distributed on the nodes of the network [35]. The decision of allocating data to the available partitions is based on the statistical similarity of the incoming data with every partition. Keeping similar data into the same partitions is a kind of proactive 'clustering' to create the basis for the application of Machine Learning (ML) algorithms and the production of knowledge. Our motivation is to, finally, have a view on the statistical dispersion of data that will facilitate the generation of efficient response plans for the incoming queries.

A set of research efforts focus on the data quality management and they have identified its necessity in any application domain. However, they seldom discuss how to effectively validate data to ensure data quality [12]. The poor quality of data could increase costs and reduce the efficiency of decision making [29]. A major research question is how to integrate heterogeneous data that are stored, dispersed and isolated from one another [27]. The integration of such data demands for intelligent management that will extract and integrate data into a high level system. Social media are also defining more challenges. Efficient models should be incorporated to manage the quality of social media data in each processing phase of the big data pipeline [17]. Such approaches should efficiently work in real time when data are received through a set of streams. If data quality is not managed, it could often result in poor decisions, with individuals bearing the greatest risk [43]. However, risks may pose negative impact in the products delivered by companies. The practical orientation of a data quality assurance mechanism should be at the combination of off-line and on-line methods to detect and replace doubtful data [2].

1.2 Contribution

We propose a pre-processing mechanism that, in real time, secures the quality of the available data. We depart from legacy solutions and instead of collecting huge volumes of data and post-process them trying to derive knowledge, we propose their real time management and allocation keeping similar data to the same partitions. We offer a pre-processing distributed scheme that decides where data should be allocated. Other solutions involve the centralized collection of vast volumes data and, accordingly, for producing knowledge, the preparation of data before the post-processing. Outliers, missing values or any other 'harmful' data should be efficiently managed through ML or data mining techniques. However, applying ML and data mining in large scale data is challenging and may require increased computational resources and time. In addition, outliers affect the performance of ML algorithms. An experimental study performed in [1] shows that the error in e.g., a classification process is reduced more than 25% when outliers are removed from the dataset. Our scheme acts *proactively* and tries to keep similar data in the same dataset preparing them beforehand to become the basis for knowledge production devoting limited resources and time in the post-processing phase.

Our data quality assurance mechanism decides if the incoming data could jeopardize the accuracy of the available datasets. We focus on the accuracy and not on the consistency as we want to identify and manage the error between the incoming data and the available partitions. Moreover, consistency relates to the usability of data, a subject that is beyond the scope of this paper. Accuracy may be jeopardized when the incoming data significantly differ with the stored datasets. Initially, we want to identify the discussed difference that will be the basis for deciding the rejection or the acceptance of data. If we identify a significant difference, we reject the data. If the incoming data are similar to the available datasets, we aim to select the partition where the data should be allocated to maintain the accuracy at high levels (secure the 'solidity' of each partition). This is because the incoming data may not be similar to the

partition where they are initially reported. A solution that stores the data where they are initially reported may affect data dispersion with negative effects in the accuracy. In addition, our approach secures that datasets will have the minimum overlapping that is also the target of legacy data separation techniques [35]. It should be noted that this paper does not aim to propose an integration model of the available data partitions. It aims at the correct allocation of the incoming data to one of the available partitions towards increasing the accuracy and, data quality respectively. The incoming data are organized in the form of vector of values where each value corresponds to a specific dimension/variable (multivariate scenario). The main research questions are: *Q1. How to maintain the accuracy of each data partition? Q2. Taking into consideration the data and their statistics present in each partition, which partition is the appropriate to store an incoming vector?*

We consider that in each partition, a processor is devoted to perform simple processing tasks (i.e., the management of the present and incoming data). Processors adopt our distributed model that incorporates the strategy for the identification of any accuracy violation event. We propose a distributed *Accuracy Maintenance Scheme* (AMS) responsible to identify violation events in any partition of the ecosystem. In cases of violation, the AMS can reject the incoming data keeping the accuracy of the corresponding partition at high levels. It is important that the AMS identifies violations in the entire ecosystem. A probabilistic approach is adopted on top of statistical measurements derived for each partition. These measurements are exchanged between processors. If a violation is not present, the AMS decides the partition that closely 'matches' to the incoming data, thus, data will be allocated there. The selection of the partition is performed by an uncertainty management scheme in terms of Fuzzy Logic (FL). We provide an FL controller responsible to derive the appropriate partition for each vector. The controller is responsible to manage/command the selection mechanism based on a FL knowledge base.

The proposed mechanism can be adopted in the domain of Edge Computing (EC) and the IoT where multiple edge nodes can be adopted to collect and process data. Such a setting can be very useful in Smart Cities (SCs) applications where the EC nodes are spread in the city and provide intelligent services to end users. These services are offered on top of the collected data, thus, their quality should be high. EC is an emerging technology that facilitates the local data processing for delivering intelligent analytics and reduce the latency when trying to respond in end users or applications requests. In addition, EC can assist in reducing the required bandwidth as limited amounts of data are transferred into the Cloud. Research community has already identified the advantages of data quality management models for EC. Experiments have been realized to deliver the performance and the data quality for complex EC applications [39]. An example scenario is the management of communications between moving vehicles and EC nodes or the analytics offered on top of this infrastructure. From the EC system point of view, it is very important to maintain the topology of the whole network being able to support failure detection, thing replacement and data quality detection [36]. Our scheme assists in securing the basis for creating knowledge, i.e., the quality of the collected data. Hence, data will be 'solid' enough excluding the outliers that may harm their statistics and lead to false decisions. The following list reports on the advantages of the proposed model:

- The proposed scheme proactively 'prepares' the data before algorithms for knowledge extraction are applied. Hence, we save time as no data preparation is necessary after their collection. For instance, it is not necessary to apply an outlier detection algorithm on top of huge volumes of data but to identify if outliers are present when data are collected from the environment and exclude them immediately.
- The proposed model proactively secures the quality of data as it excludes data that may lead to an increased error during knowledge production. For instance, as shown through experiments [1], outliers may lead to an increased error when participating in the desired processing.
- Our scheme leads to the minimum overlapping of the available datasets that is the target of the legacy data separation algorithms. However, this is realized in a pre-processing step placing the incoming data at the appropriate partition.

The paper is organized as follows. Section 2 reports on the related work while Sect. 3 presents the envisioned setting. In Sect. 4, we present the proposed mechanism and describe its parts. Section 5 discusses the evaluation of the proposed mechanism while in Sect. 6, we conclude our paper by giving future research directions.

2 Related work

A survey of data quality dimensions is presented in [37] while, at the same time, the authors propose a framework that combines data mining and statistical techniques to extract the correlation of these dimensions. The aim is to measure the dependencies and figure the methods for creating new knowledge. In [29], the authors propose the use of a model that consists of nine determinants of quality. From them, four are related to information quality and five describe system quality. The determinants of quality are: (a) Information quality: accuracy, completeness, currency, format; (b) System quality: accessibility, reliability, response time, flexibility, integration. It should be noted that the assessment of dimensions could be task-independent or task-dependent [31]. Task-independent dimensions represent scenarios where data are evaluated without the knowledge of the application domain. Hence, such metrics may be applied in any dataset. Task-dependent dimensions incorporate requirements of the organization and the application domain.

Recently, the advent of the large-scale datasets defined more requirements on the data quality assessment. Given the range of big data applications, potential consequences of bad data quality can be more disastrous and widespread [32]. In [25], the authors propose the '3As Data Quality-in-Use model' composed of three data quality characteristics i.e., contextual, operational and temporal adequacy. The proposed model could be incorporated in any large scale data framework as it is not dependent on any technology. A view on the data quality issues in big data is presented in [32]. A survey on data quality assessment methods is discussed in [6]. Apart from that, the authors present an analysis of the data characteristics in large scale data environments and describe the quality challenges. A framework dealing with hierarchical data quality assessment is also proposed. The provided framework consists of big data quality dimensions, quality characteristics, and quality indexes. The evolution of the data

quality issues in large scale systems is the subject of [5]. The authors discuss various relations between data quality and multiple research requirements. Some examples are: the variety of data types, data sources and application domains, sensor networks and official statistics.

Other application-specific large scale data quality assessment methods are as follows. In [42], a high number of data quality issues are identified based on the 'testbed' of the Vrije Universiteit, Brussels. The use case aims to reveal data quality dimensions and prioritize cleaning tasks in different dimensions. Apart from that, another goal is to facilitate the use of dimensions from users that do not have domain knowledge. The result is to setup the basis for providing automated mechanisms for data quality evaluation on top of multiple correlated dimensions. In [40], the authors present results and practical insights from a large scale study conducted in the telecommunications industry. The case study shows how the requirements for data quality can be collected to define an architecture adopted for the quality assessment. The authors also propose a data quality approach that combines data quality and data architectures into a framework where a set of steps, processes and tools are adopted. In [3], the authors discuss data quality in health information systems. In the beginning, a review of the relevant literature is presented and, accordingly, data quality dimensions and assessment methodologies in the health domain are discussed. The results of the research show that, among a total of fifty dimensions, eleven are identified as the main dimensions. Widely adopted dimensions are: completeness, timeliness, and accuracy. The authors in [33] describe the outcomes of a Workshop titled 'Towards a common protocol for measuring and monitoring data quality in European primary care research databases'. The aim is to point out the most significant issues that affect data quality in databases from the perspective of researchers, database experts and clinicians. Multiple ideas were exchanged on what data quality metrics should be available to researchers.

The above discussion reveals that, among the identified data quality dimensions, accuracy plays an important role. For defining accuracy in the data, multiple methods are adopted (e.g., clustering). In [18], the main scenario is a setting where multiple sensor nodes report their values. A distributed clustering algorithm is proposed that adopts the spatial data correlation among sensors. Data accuracy is performed for each distributed cluster. The identified clusters are not overlapped and incorporate different sizes to collect most accurate or precise data at each cluster head. The provided accuracy model is compared to results from an information accuracy model. Fuzzy logic is applied in [38]. The authors propose a distributed fuzzy clustering methodology for identifying data accuracy. The fuzzy clustering model is accompanied by a facilitator model to define a novel distributed fuzzy clustering method.

Research efforts that are close to our work are presented in [2,17,27]. These efforts discuss models for the management of the data either off- or on-line to secure their quality when large scale data are taken into consideration. Outliers and fault detection accompanied by autoregressive models, on top of streams, are adopted to evaluate the data quality [2]. In a high level, business decision making techniques undertake the responsibility of validating the data as they arrive [17]. The aforementioned efforts do not take into consideration the presence of multiple data partitions and their management. Multiple partitions are the subject of the research presented in [27]. However, the

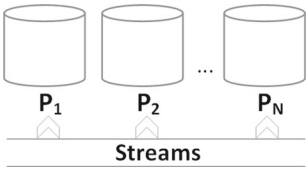

Fig. 1 The proposed streaming environment

authors propose an integration scheme opposite to our work where we aim to provide means for assigning data to the underlying partitions.

3 Preliminaries

Let N partitions be available for storing the incoming data fed by the corresponding streams. In each partition, an amount of data (tuples/vectors) is stored. Every tuple/vector conveys data for a set of variables (multivariate scenario). The ith partition contains the dataset D_i. We focus on a setting where multivariate data are the envisioned scenario. Without loss of generality, we consider the same number of variables in every partition. Data are characterized by a set of $|\mathbf{x_i}|$ vectors, i.e., $\mathbf{x_i} = \{x_{ij}\}, j = 1, 2, \ldots, M$, M is the number of variables in each vector. For instance, x_{12} represents the realization of the 2nd variable in the 1st partition. Streams report vectors of values to the corresponding partitions (i.e., every stream to a specific partition) represented by $\mathbf{x}$. Example vectors are as follows: $\langle 12.0, Low, Area1, 55.1\rangle$, $\langle 32.0, High, Area2, 45.2\rangle$, $\langle 22.0, Medium, Area5, 75.9\rangle$. Variables could be realized by nominal, binary, ordinal or numeric values. In Fig. 1, we present the envisioned setting. It should be noted that we do not perform any assumption for the location of the partitions as well as their characteristics.

Our aim is to identify if $\mathbf{x}$ deviates from the ecosystem of partitions and, if not, to 'allocate' $\mathbf{x}$ to the 'appropriate' partition. With the term 'appropriate', we denote the partition where statistics of $\mathbf{x_i}$ 'match' to $\mathbf{x}$. Let us give a specific example. Suppose we have available two (2) partitions and our data vectors consist of two (2) variables (without loss of generality, we consider numeric values for both variables). In the first partition, the mean vector is $\langle 0.2, -1.2\rangle$ while in the second partition the mean vector is $\langle 1.8, 1.7\rangle$. Initially, we receive the vector $\langle -4.3, 4.5\rangle$. This vector based on the Mahalanobis distance may be an outlier [21]. The inclusion of this vector in each of the two datasets will 'destroy' the means and may lead to false knowledge extraction as already explained. Hence, the vector is rejected by the proposed scheme as it does not fit to the ecosystem of the available datasets. Suppose now we receive the vectors $\langle 0.1, -0.6\rangle$ and $\langle 1.7, 2.0\rangle$. Both vectors are not outliers, thus, the first can be placed at the first partition while the second can be placed at the second partition updating their statistics. It should be noted that the proposed mechanism considers a number of vectors present in every dataset. A warm up period may be adopted to initially store the first incoming vectors into the discussed datasets. At this point, we could adopt a specific strategy concerning the location where each vector will be stored, e.g., a centralized approach will conclude the partition for each vector (such a centralized approach will not add a significant overhead as the warm up period 'delivers' a limited

number of vectors). However, in the case where the first incoming vectors are outliers compared to the vectors that will arrive in the future, we may conclude wrong statistics in our calculations. Hence, the proposed mechanism can be adopted as the second step after a pre-processing phase that refers in the collection of vectors in the warm up period and the initial exclusion of the outliers present in the available datasets.

In the current research, we focus on a 'strict' mechanism that when identifies an increased error between the incoming data and the entire set of the available partitions decides that the data quality is jeopardized (i.e., accuracy violation event). A model that takes into consideration the natural evolution of data in the error identification is left as a future work. For simplicity, we consider that data stored in the envisioned partitions are generated by a Normal distribution. The proposed mechanism checks the statistical similarity of $\mathbf{x}$ with every partition and decides if $\mathbf{x}$ will be rejected or kept for storage in a partition. The rejection is made when $\mathbf{x}$ does not match to the statistics of any partition in the ecosystem.

As noted, processors placed in front of each partition have the responsibility of managing the incoming vectors and performing simple statistical calculations (e.g., extraction of mean and variance). We consider that in each partition there is knowledge on the mean and variance vectors in the remaining partitions. The mean vector depicts the mean of each variable as realized in the corresponding dataset. A similar rationale holds true for the variance vectors. The calculation is performed at pre-defined intervals and, accordingly, the mean and variance vectors are sent to the peer partitions. Hence, the entire set of partitions has a view on the statistics of data stored in the entire ecosystem. Some problems may arise when specific parts of the network are unreachable. In such cases, the delay for transferring the vectors in the network will be huge probably making obsolete the current view on the statistics of the 'invisible' partitions. For solving this problem, we could combine the proposed mechanism with a monitoring or an aging mechanism that will be responsible to provide information for the unreachable parts of the network. The unreachable parts can be excluded from the envisioned processing supporting an adaptive model that will exclude/include partitions based on the network's performance. However, this aspect is beyond the scope of the current effort. In any case, the proposed model can be efficiently adopted in EC applications considering nodes in close proximity and a good connectivity to be able to exchange the discussed information. Through this approach, we aim to provide a 'global' outlier detection scheme that secures the accuracy in the entire ecosystem no matter the location where the incoming vectors arrive. Instead of recording the entire set of data and, afterwards, removing the outliers and separate the data into a set of non overlapping partitions, we provide a proactive mechanism that, in real time, it decides if the incoming data should stored and where they should be placed. This way, we require the minimum time and lower amount of resources for pre-processing compared to the case where we should post-process vast volumes of data and split them into the discussed partitions.

Our AMS consists of two parts, i.e., the *Accuracy Violation Detection Scheme* (AVDS) and the *Partition Identification Scheme* (PIS). The AVDS aims to identify accuracy violation events on top of the collected mean and variance vectors. AVDS adopts a probabilistic model that derives the probability that $\mathbf{x}$ is generated by each partition. The assignment of $\mathbf{x}$ in a partition is PIS's responsibility. PIS adopts an

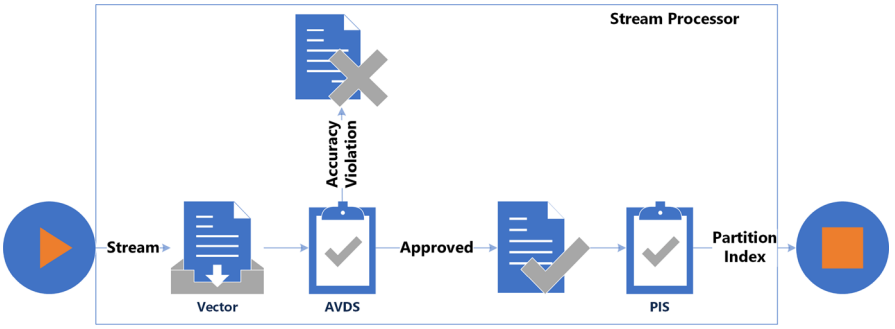

Fig. 2 The process model of the proposed framework

uncertainty driven decision making in terms of FL. An FL controller receives the result of the AVDS (a set of probabilities) and the distance of $\mathbf{x}$ with each partition. The controller depicts the strategy of selecting partitions through the conversion of a linguistic control method on top of a set of fuzzy control rules. The 'allocation' decision is based on the result of the FL controller that calibrates the probability of having $\mathbf{x}$ generated by each partition with the maximum distance realized for every variable in the datasets.

Figure 2 presents the processing of the proposed framework. The proposed scheme consists of a sequential processing on the incoming vectors that finally concludes to their rejection or acceptance and, accordingly, in case of an acceptance, it selects the appropriate partition where vectors will be placed. Vectors that do not 'match' with the available partitions are seen as outliers to the ecosystem. Most real-world databases include a certain amount of exceptional values, generally termed as outliers [19]. The identification and the isolation of any outlier data, in real systems, is significant to improve the quality of the original data and reduce the impact of the outliers in their statistics and, thus, the knowledge extraction. Our mechanism offers this outlier elimination process before the data are stored and, accordingly, be processed for delivering analytics.

4 The distributed accuracy management scheme

4.1 The accuracy violation detection model

Without loss of generality and for simplicity in our calculations, we consider that data, in every partition, follow the same distribution with N realizations i.e., different values of the distribution parameters. The bottom line is to check if $\mathbf{x}$ is generated by any of them. Actually, AVDS tries to see if $\mathbf{x}$ is generated by a mixture of N distributions before it decides the acceptance or the rejection of the incoming vector. We focus on a finite mixture of $N \cdot M$ distributions $\Theta_{ij}, i = 1, 2 \ldots, N, j = 1, 2 \ldots, M$ represented by the *Probability Density Functions* (PDF) $f_{\Theta_{ij}}(x)$. Given the distributions Θ_{ij} and constant weights $w_i > 0$, the PDF of the mixture is

$$f_{\Theta_i}(x) = \sum_{\forall i} w_i \prod_{\forall j} f_{\Theta_{ij}} \tag{1}$$

In this work, we consider that $w_i = \frac{1}{N} \forall i$. When $\mathbf{x}$ arrives in a partition, the corresponding AVDS calculates $P(\mathbf{x})$ which is the probability that the vector is generated by one of the distributions characterizing the available partitions and decides if an accuracy violation is present. As we focus on multivariate vectors, the discussed probability is calculated as follows:

$$P(\mathbf{x}) = \sum_{\forall i} \prod_{\forall j} f_{\Theta_{ij}}(\mathbf{x}) \tag{2}$$

Actually, $P(\mathbf{x})$ is calculated on top of the single probabilities that $\mathbf{x}$ is generated by each partition, i.e., $P(\mathbf{x}, D_i)$. If $P(\mathbf{x})$ is below a pre-defined threshold P_T (it is defined by experts), $\mathbf{x}$ is rejected, otherwise PIS is triggered to deliver the partition where $\mathbf{x}$ will be allocated.

Additionally, our mechanism is also based on the distance between $\mathbf{x}$ and each vector in a partition, i.e., $\mathbf{y_i}, i = 1, 2, \ldots, N$. Initially, we calculate the probability that $\mathbf{x}$ is generated by the available partitions and, accordingly, we adopt an additional technique to depict the 'closeness' of $\mathbf{x}$ with each partition. We rely on a distance metric to build a model that is based on the trade-off between the probability that $\mathbf{x}$ is generated by a partition and the distance with that partition. The reason is that we may identify multiple partitions exhibiting a high probability of generating $\mathbf{x}$. We combine this probability with another, different statistical measure to provide an 'ensemble' mechanism that handles the uncertainty present in this scenario. The uncertainty is related to the decision on the selection of the appropriate partition where $\mathbf{x}$ should be allocated. For instance, if we focus on two partitions and an incoming vector and get a probability equal to 0.7 depicting that the vector could be 'produced' by both partitions, we should finally rely on the distance between the vector and the two partitions. Let the distance with the first partition be 0.7 and the distance with the second partition be 1.0. This means that, finally, the first partition takes an 'advantage' to host the incoming vector. However, the final decision will be delivered on top of the result of the FL controller described below. The FL controller tries to manage the uncertainty present in such scenarios (in our example, the distance with both partitions is very close). We propose the adoption of the L_∞-norm or Chebyshev distance [14]. The L_∞-norm is calculated over the entire set of the envisioned variables in the available vectors. The following equation holds true:

$$L_\infty(\mathbf{x}, D_i) = \frac{1}{|D_i|} \sum_{j=1}^{|D_i|} \lim_{h \to \infty} \left(\sum_{k=1}^{M} |x_k - y_{jk}| \right) = \frac{1}{|D_i|} \sum_{j=1}^{|D_i|} \max_{k=1}^{M} |x_k - y_{jk}| \tag{3}$$

where $y_{jk} \in D_i$ is the kth variable in the jth vector that belongs to D_i. Both, $P(\mathbf{x}, D_i)$ and $L_\infty(\mathbf{x}, D_i)$ are fed into the proposed uncertainty management mechanism realized as a FL controller. The FL controller receives N pairs of probability-distance and

derives the *value of similarity* (ω) between $\mathbf{x}$ and each partition. The partition with the highest ω is selected to host the incoming vector.

As noted, the available partitions exchange the mean and variance vectors (we consider that data are numeric) to maintain a view on the hidden statistics of each dataset. The calculation of the mean and the variance for each variable in the datasets could be easily performed through the adoption of the Expectation Maximization (EM) algorithm. The EM algorithm [9] could be adopted for learning the mean and the standard deviation/variance. The EM algorithm generalizes the maximum likelihood model assuming an incomplete dataset. It tries to find the parameters that maximize the log probability $\log P(\mathbf{x}; \theta)$ on top of a set of data. A number of sub-problems are selected in a way that their solutions $\hat{\theta}_1, \hat{\theta}_2, \ldots$ guarantee the convergence to a local optimum of the aforementioned function $\log P(\mathbf{x}; \theta)$. The process involves two steps: (a) the expectation step (E-step); (b) the maximization step (M-step). In the E-step, a function g is selected that bounds $\log P(\mathbf{x}; \theta)$ everywhere with

$$g\left(\hat{\theta}_k\right) = \log P\left(\mathbf{x}; \hat{\theta}_k\right) \tag{4}$$

In the M-step, the algorithm moves to a different parameter/solution, say $\hat{\theta}_{k+1}$, that maximizes the function g. As the new parameter maximizes g and g bounds $\log P(\mathbf{x}; \theta)$, the following equation holds true:

$$\log P(\mathbf{x}; \theta_k) = g\left(\hat{\theta}_k\right) \leq g\left(\hat{\theta}_{k+1}\right) = \log P(\mathbf{x}; \theta_{k+1}) \tag{5}$$

The EM algorithm is applied in each partition and estimates the mean $\boldsymbol{\mu_i}, i = 1, 2, \ldots, N$ and can be easily extended for multivariate data.

4.2 The selection process

For selecting the appropriate partition where $\mathbf{x}$ will be allocated, we propose the use of a Type-2 FL controller [44]. It should be noted that the adopted controller accompanied by its knowledge base (in terms of rules) is triggered just after the decision for the acceptance of $\mathbf{x}$. It does not require any initialization process but just to feed it with the envisioned inputs. A Type-2 FL controller is an FL controller that adopts Type-2 fuzzy sets in the definition of membership functions for each input and output variable. Membership functions in a Type-2 FL controller are intervals defining the upper and the lower bounds for each fuzzy set [23] (explained later). The area between the two bounds is referred to as Footprint of Uncertainty (FoU). The controller defines the knowledge base of the proposed scheme in the form of a set of *Fuzzy Rules* (FRs). Such FRs try to manage the uncertainty related to if $\mathbf{x}$ closely matches to each partition D_i. FRs refer to a non-linear mapping between two inputs: (1) $P(\mathbf{x}, D_i)$ and (2) $L_\infty(\mathbf{x}, D_i)$ and a single output, i.e., the ω_i. The antecedent part of FRs is a (fuzzy) conjunction of inputs and the consequent part of the FRs is the β parameter indicating the belief that an event *actually* occurs, i.e., the belief that a specific partition should host $\mathbf{x}$. The proposed FRs have the following structure: **IF** $P(\mathbf{x}, D_i)$ is A_{1k} **AND** $L_\infty(\mathbf{x}, D_i)$ is A_{2k} **THEN**

ω_i is B_k, where A_{1k}, A_{2k} and B_k are membership functions for the k-th FR mapping $P(\mathbf{x}, D_i)$, $L_\infty(\mathbf{x}, D_i)$ and ω_i (*values into unity intervals*), by *characterizing* their values through the terms: *low*, *medium*, and *high*. A_{1k}, A_{2k} and B_k are represented by two membership functions corresponding to *lower* and *upper* bounds [23]. For instance, the term '*high*', whose membership for $P(\mathbf{x}, D_i)$ is a number $z(P(\mathbf{x}, D_i))$, is represented by two membership functions. Hence, $P(\mathbf{x}, D_i)$ is assigned to an interval $[z_L(P(\mathbf{x}, D_i)), z_U(P(\mathbf{x}, D_i))]$ corresponding to a lower and an upper membership function z_L and z_U, respectively. The interval areas $[z_L(P(\mathbf{x}, D_i)_j), z_U(P(\mathbf{x}, D_i)_j)]$ for each $P(\mathbf{x}, D_i)_j$ reflect the uncertainty in defining the term, e.g., '*high*', useful to determine the exact membership function for each term.

Without loss of generality, we assume that $P(\mathbf{x}, D_i), L_\infty(\mathbf{x}, D_i) \in [0, 1]$. We also define $\omega_i \in [0, 1]$. A ω_i close to unity denotes the case where the corresponding partition is similar with the incoming vector $\mathbf{x}$. The opposite stands when ω_i tends to zero. The aim of the proposed mechanism is to secure that the distribution of the data will not be considerably changed. The envisioned PIS is responsible to deliver the most appropriate partition where a vector will be stored. PIS's inputs are related to the statistical similarity between the incoming vector and the available partitions. Hence, the proposed module decides having the goal of keeping the changes in the distribution of the data limited as it allocates the incoming vectors to the most similar datasets. For inputs and the output, we consider three linguistic terms: *Low*, *Medium*, and *High*. *Low* represents that a variable (input or output) is close to zero, while *High* depicts the case where a variable is close to one. *Medium* depicts the case where the variable is around 0.5. For each term, human experts define the upper and the lower membership functions. Here, we consider triangular membership functions as they are widely adopted in the literature.

In Table 1, we present the adopted knowledge base (i.e., the set of FRs). The discussed Table presents the FL rules that define the basis for extracting ω_i based on two inputs, i.e., $P(\mathbf{x}, D_i)$, $L_\infty(\mathbf{x}, D_i)$. For instance, when the probability $P(\mathbf{x}, D_i)$ is low and the distance between the $\mathbf{x}$ and the ith dataset $L_\infty(\mathbf{x}, D_i)$ is also low, then ω_i will be low as well. The reason is that in this case, there is limited probability to have $\mathbf{x}$ produced by the ith dataset, thus, the ith dataset should not a candidate for hosting $\mathbf{x}$. FL rules can be defined either manually by experts in the domain or can be extracted through automated generation algorithms like those presented in [4]. However, the automated generation of rules has the prerequisite of the appropriate dataset containing numerical values for the 'combinations' of the adopted input variables. Every FR is a combination of the FL sets depicted by the aforementioned linguistic values. The proposed system calculates the firing strength of each FR to, finally, calculate the linguistic value of ω_i (which is, then, defuzzified to a real number). The reception of $\mathbf{x}$ triggers the AVDS to calculate $P(\mathbf{x}, D_i)$ and $L_\infty(\mathbf{x}, D_i)$ and feed them to the controller (that is the core part of PIS). Initially, the calculation of the interval (based on the membership functions) for each input takes place followed by the calculation of the active interval of each FR. Afterwards, the controller combines the active interval of each FR and the corresponding consequent. Finally, the defuzzification phase[1]

[1] Defuzzification is the process of producing a quantifiable result in FL, given fuzzy sets and the corresponding membership degrees.

Table 1 The knowledge base of the proposed controller

Rule	$P(\mathbf{x}, D_i)$	$L_\infty(\mathbf{x}, D_i)$	ω_i
1	Low	Low	Low
2	Low	Medium	Medium
3	Low	High	Medium
4	Medium	Low	Medium
5	Medium	Medium	Low
6	Medium	High	Medium
7	High	Low	High
8	High	Medium	Medium
9	High	High	Medium

produces a crisp value (i.e., a real value) for ω_i. The most common method for 'type reduction' is the *center of sets type reducer* [24]. Our mechanism collects the realization of the matches $\omega_i, i = 1, 2, \ldots, N$ and derives the highest value $\alpha = \max_{\forall i} \omega_i$ accompanied by the index of the corresponding partition. This partition will finally host the incoming vector $\mathbf{x}$.

5 Experimental evaluation

5.1 Performance metrics and simulation set-up

We report on the performance of the proposed mechanism through the adoption of a set of performance metrics. Our main target is to reveal the 'solidity' of the datasets as realized after the management of multiple vectors. A high 'solidity' depicts a low error in the data present in each partition. According to [41], erroneous data can heavily affect analysis, leading to biased inference. Hence, in this paper, we consider that the quality of the data is represented by the data accuracy, i.e., the lowest possible error among the data and, thus, the lowest possible error between the incoming vectors and the available partitions. Actually, in our paper, accuracy is defined by the closeness to the already present values. In our evaluation plan, for identifying the quality of the data stored in each partition, we rely on statistical metrics like the standard deviation. Standard deviation is a measure of the dispersion of a dataset from its mean (i.e., error between the data as depicted by the mean). A low value of the standard deviation shows that data are around their mean, thus, a high 'solidity' is identified.

Initially, we focus on the standard deviation of each dataset, i.e.,

$$\mathbf{s} = \sqrt{\frac{\sum_{i=1}^{|D|} (\mathbf{x_i} - \boldsymbol{\mu})}{|D| - 1}} \tag{6}$$

$\mathbf{s}$ is calculated after the management of the incoming vectors and separately for each variable. At pre-defined intervals, we calculate the standard deviation for each variable incorporated in the datasets. We also provide results for the *probability density*

estimation (pde) of **s** to reveal the 'solidity' of each dataset. The pde is delivered on top of the standard deviation values as calculated in our experiments. **s** quantifies the amount of dispersion of the resulted datasets. A low value for **s** depicts a solid dataset where the included vectors exhibit low difference between each other. **s** will be low when the incoming vectors are allocated in datasets with which they exhibit a low distance.

The metric δ depicts the percentage of the correct allocations as realized through the distance between the incoming vectors and the statistics of the selected partition. δ is defined as follows:

$$\delta = \frac{|\psi < \epsilon|}{E} \tag{7}$$

where ψ is the distance between the accepted vector and the selected partition and ϵ is a pre-defined threshold adopted to consider a low distance. When ϵ is met, the accepted vector and the selected partition are considered as close (low distance). In the aforementioned equation, E is the the total number of the accepted vectors. The best performance is achieved when $\delta \rightarrow 1.0$.

In addition, we adopt the metric γ that represents the percentage of the partition checks that are necessary to place the incoming vectors. The following equation holds true:

$$\gamma = \frac{\xi}{E} \tag{8}$$

where ξ is the number of the necessary partition checks (recall that E is the the total number of the accepted vectors). γ depicts the number of vectors that should be allocated in another partition from that they are initially reported.

We adopt two types of datasets, i.e., a synthetic and a real. The synthetic trace is generated through the adoption of the Gaussian distribution. We simulate the production of 10,000 vectors and randomly select in the interval [1,N] the initial partition where each vector is reported. For each variable in the incoming vector, we generate a random value in the interval [0, 100]. The real dataset comes from National Agency for New Technologies, Energy and Sustainable Economic Development.[2] The dataset contains 9358 instances of hourly averaged responses from an array of 5 metal oxide chemical sensors embedded in an Air Quality Chemical Multisensor Device. Data were recorded from March 2004 to February 2005 (1 year) representing the longest freely available recordings of on field deployed air quality chemical sensor responses. From this dataset, we adopt the measurements of four parameters, i.e., hourly averaged concentration CO, temperature in Celsius , relative humidity and absolute humidity. We consider that each parameter corresponds to a partition where the collected values should be finally allocated. As the initial partitions are only four, we replicate them to deliver a dataset that contains more partitions. In each round of the performed simulations, we randomly select (1) a random trace from the available; (2) a random row in this trace and we reason about the acceptance and the allocation of the selected row.

[2] http://archive.ics.uci.edu/ml/datasets/Air+Quality.

The initialization of the proposed scheme refers in the separation of the adopted traces in a number of datasets/partitions. As far as the FL controller concerns, we define the membership functions and the FL rules adopting the Juzzy Fuzzy Logic toolkit [45] as depicted by the Table 1. We perform an extensive set of simulations for $N \in \{5, 10, 50, 100\}$ and report on our results. We simulate the exchange of the mean and variance vectors through the adoption of the ϕ parameter. We consider that $\phi \in \{0.2, 0.8\}$. Mean and variance vectors are exchanged in the network with probability ϕ. When $\phi = 0.2$, we simulate a low exchange rate, i.e., the mean and variance vectors are not frequently exchanged between the partitions. It becomes obvious that a low exchange rate limits the number of messages in the network, however, a low ϕ may have consequences in the statistical measurements performed by the proposed mechanism. When $\phi = 0.8$, the exchange rate is high and it leads to an increased number of messages in the network.

5.2 Performance assessment

The first set of our experiments involve the synthetic dataset. In Fig. 3, we present our results concerning the pde of $\mathbf{s}$ for $N = 10$. We observe that the higher the ϕ is the lower the $\mathbf{s}$ becomes. This means that an increment in ϕ leads to low dispersion of the vectors allocated in each partition. In Fig. 4, we increase the number of partitions and get $N = 50$. Now, a high ϕ makes the results to be dispersed in the entire interval while a low ϕ 'concentrates' $\mathbf{s}$ around 5.0. When we get $N = 100$ (see Fig. 5), the increment in ϕ makes $\mathbf{s}$ to be concentrated around 10.0. In general, a low number of partitions leads to a low dispersion in the vectors allocated in each partition. The reason is that the 'choices' for storing the incoming vectors are limited, thus, the proposed mechanism copes to allocate the data into the appropriate partitions. The adopted FL controller manages to derive the right partition minimizing the variation in the allocated vectors. A low number of partitions leads to a high number of vectors included in each partition with 'positive effects' in the statistical measurements. More accurate statistical results are delivered and, consequently, the FL controller may derive partitions with high similarity with the incoming vectors. When the number of the partitions is high, there is an increased possibility of having multiple datasets that may be 'similar' to the incoming vectors, thus, the final allocation leads to partitions containing a low number of vectors which may increase the dispersion of the involved data. The performance of the proposed PIS module is affected by the FL controller and its fuzzy rules. Fuzzy rules are responsible to deliver the final decision related to the partition where a vector should be placed. Hence, the design of the rules is crucial. In our future research plans, we aim to provide a model that is based on the automatic generation of fuzzy rules based on numerical values as well as in the involvement of more variables in the FL controller. Hence, the final decision will not be based only on the distance between the incoming vectors and the available partitions but also on other aspects of the problem (e.g., the cardinality of each partition).

We report on the comparison between the proposed model and an outlier detection scheme. As we focus on the performance of the 'outlier detection aspect' of the proposed model (the first part of our scheme), we rely on a single variable for the

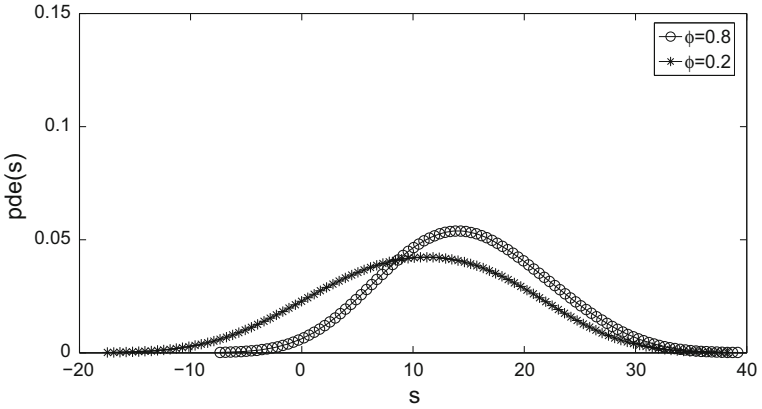

Fig. 3 The pde of **s** ($N = 10$)

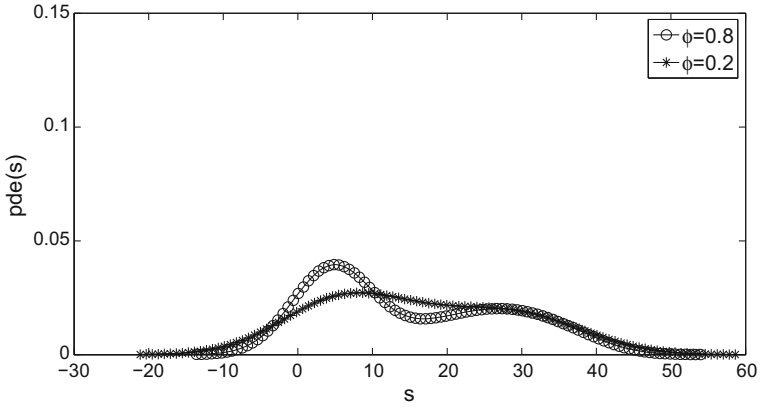

Fig. 4 The pde of **s** ($N = 50$)

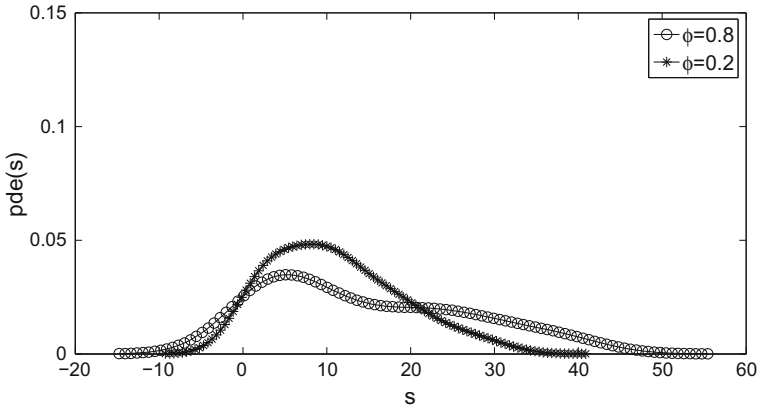

Fig. 5 The pde of **s** ($N = 100$)

evaluation. We compare our model with the CumSum algorithm [30] that is widely adopted for outliers detection. We focus on the synthetic trace and perform experiments for $N \in \{10, 50, 100\}$. Recall that the synthetic trace involves 10,000 vectors with random values for each variable. Our scheme's results incorporate 9405, 1526 and 430 outliers for $N \in \{10, 50, 100\}$, respectively. The results of the CumSum are 9932, 9924 and 9908 outliers for the same N realizations. The CumSum algorithm exhibits worse performance (as it characterizes the majority of data as outliers) than the proposed model as it relies on the difference of the incoming vectors with the distribution of the mean values as calculated in each partition. The adoption of a dataset that corresponds to a very dynamic environment (as depicted by the synthetic trace) where values change continuously, it negatively affects the performance of the CumSum. The proposed model is positively affected by the increased N. The number of vectors characterized as outliers is limited (compared to the total number of vectors) when $N \rightarrow 100$.

We report on the number of correct allocations for two scenarios: (a) when a single variable is the case; (b) when multiple variables form each incoming vector. Figure 6 presents our results for the single variable experimental scenario while Fig. 7 depicts the results for the multiple variables scenario. In the single variable scenario, we observe that the increased exchange rate of mean and variance vectors positively affects the performance. In such cases, δ increases together with N. When $\phi = 0.2$, there are fluctuations in the δ realizations when we focus on increments in the number of partitions. The low exchange rate may ruin the statistics of each partition as partitions are not updated for the mean and variance measures on their peers. Hence, the final decisions may be taken on top of obsolete statistics leading to sub-optimal solutions. In addition, we observe the opposite behavior when we focus on multiple variables. δ decreases as $N \rightarrow 100$ when $\phi = 0.8$ while it increases for a high N and $\phi = 0.2$. This behavior shows us that the proposed mechanism is heavily affected by the number of variables and the probability that a vector 'belongs' to a partition. As described, the final probability, for each partition, is the product of the single probabilities for each variable. This is a 'strict' model that may be affected by the probability of just a single variable. Assume a vector with three variables. Let the probability for the two variables be equal to 1.0 and the probability of the third variable be equal to 0.5. The final result is 0.5. An intelligent scheme for excluding variables from the calculations is in the first place of our research agenda. For instance, we may adopt principal component analysis techniques or other models for feature selection to identify the most significant variables in the incoming vectors. Such 'important' variables may be the basis for calculating the discussed probabilities, thus, limiting the search space.

In Figs. 8 and 9, we present our results for the γ metric. Recall that γ represents the number of the incoming vectors that match to another partition compared to the partition where they are initially reported. In our simulations, we produce the initial partition based on a very dynamic environment where any vector could be reported in any partition. Our results show that, in such cases, multiple checks/movements may be needed. This is more intense when we consider multiple variables in each vector. γ is equal to 100% which means that every reported vector is allocated in another partition compared to the partition of the initial presence. In the single variable scenario, γ increases as N increases no matter the value of ϕ. This is considered as natural due to

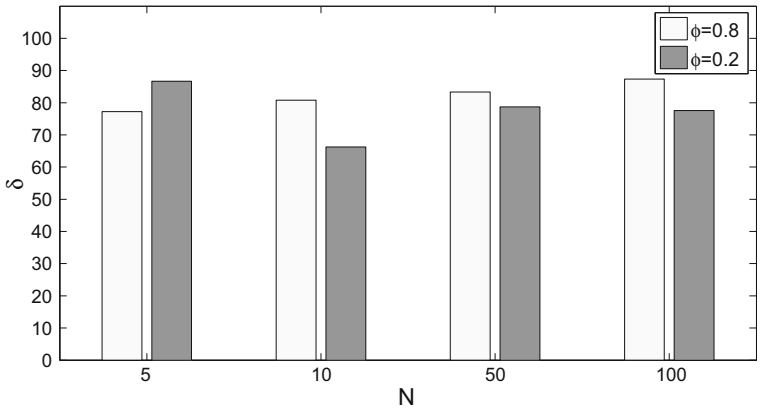

Fig. 6 δ Results for the synthetic trace (a single variable)

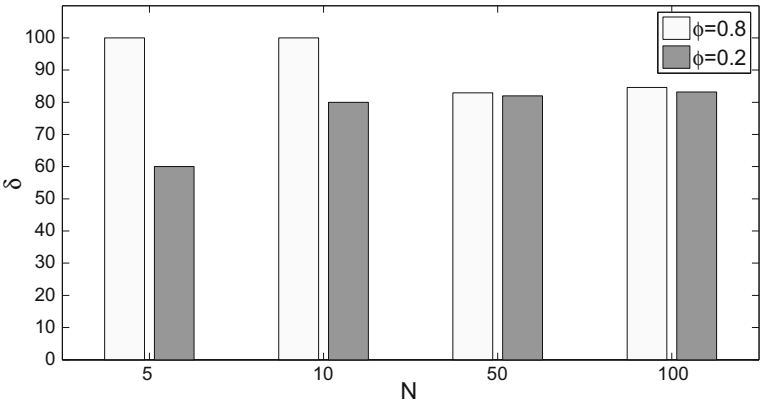

Fig. 7 δ Results for the synthetic trace (multiple variables)

the Uniform distribution of the initial presence of each vector. It should be noted that an increased number of checks/movements will increase the messages circulated in the network. A mechanism that deals with the trade off between the storage of a vector locally and the cost for the movement in the final decision is left for future work.

The second part of our experiments focuses on the involvement of a real dataset. We replicate the initial dataset to produce multiple partitions and execute a set of simulations for $N \in \{4, 15\}$. It should be noted that the initial dataset consists of four (4) partitions. Figures 10 and 11 depict the performance of the proposed mechanism as realized through the dispersion of data in each partition. We observe similar results for the two experimental scenarios ($N = 4$, $N = 15$). A low ϕ leads to a higher $\mathbf{s}$ than in the scenario where $\phi = 0.8$. Recall that a low ϕ negatively affects the final decision as it may be made on top of obsolete statistics.

Figure 12 reports on the performance of our mechanism concerning the δ metric. No matter the ϕ value, we observe a slight increment in δ. The increased number of partitions positively affects the performance of the proposed mechanism as more

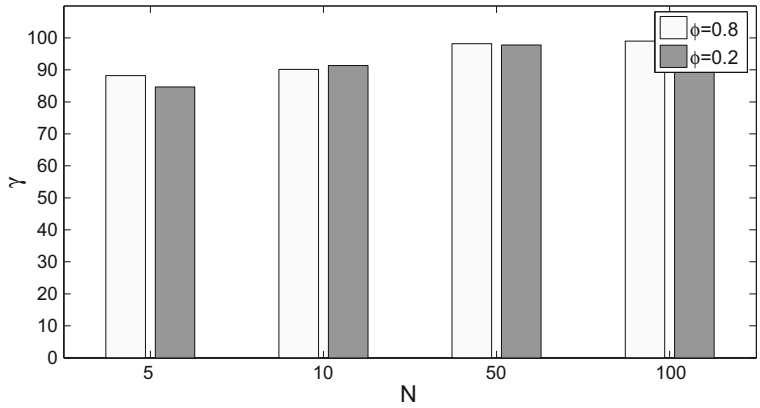

Fig. 8 γ Results for the synthetic trace (a single variable)

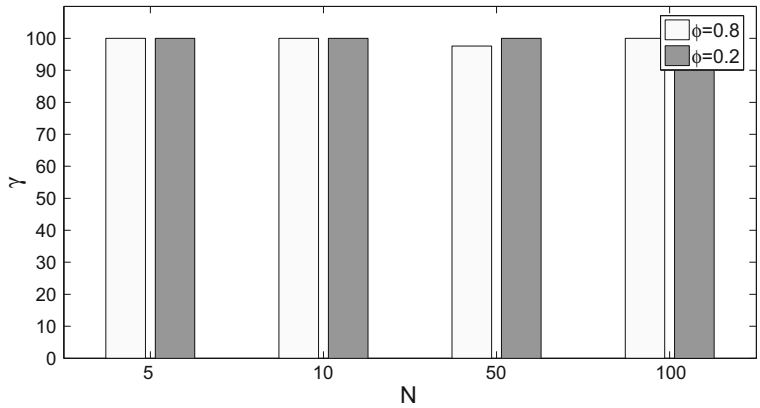

Fig. 9 γ Results for the synthetic trace (multiple variables)

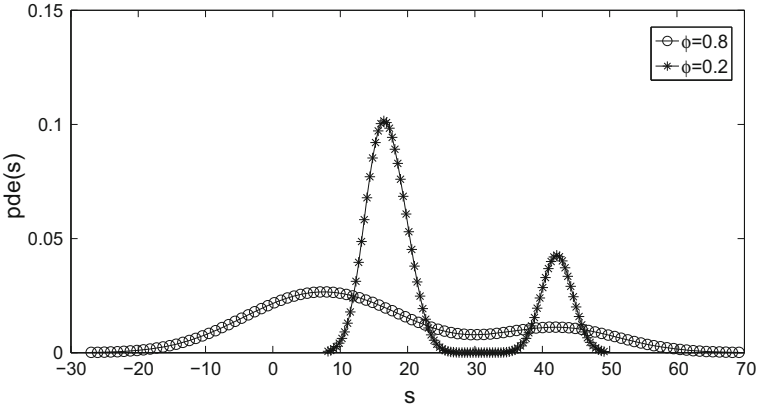

Fig. 10 The pde of **s** for the real dataset ($N = 4$)

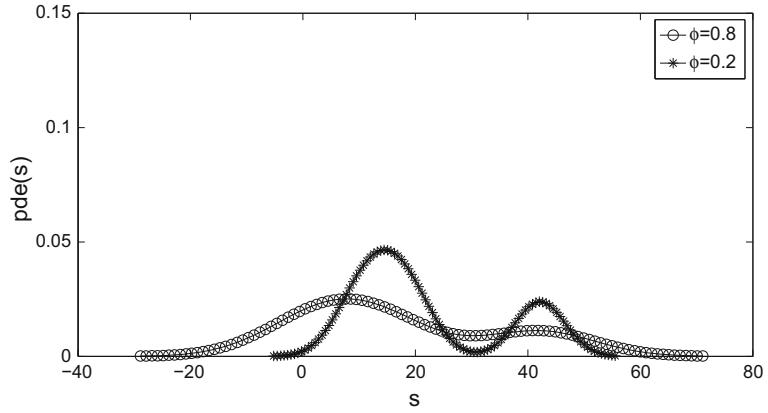

Fig. 11 The pde of **s** for the real dataset ($N = 15$)

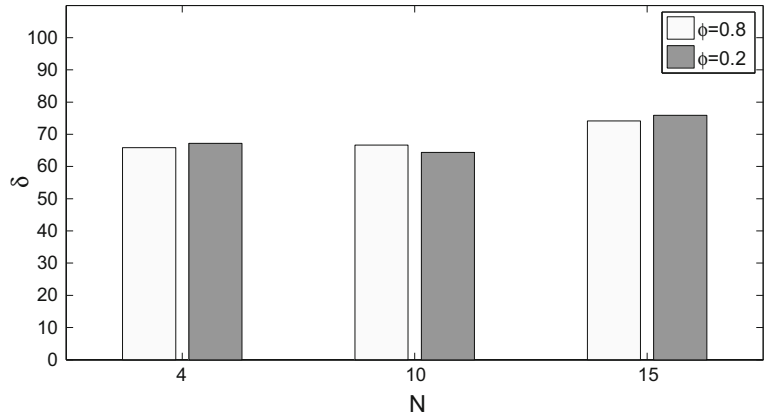

Fig. 12 δ Results for the real trace

vectors are allocated in the correct partition. Finally, in Fig. 13, we present our results for the γ metric. γ increases as N increases as well. In this experimental scenario, we observe a similar performance like in the evaluation with the synthetic dataset. However, in the real dataset scenario, γ is not close to 100% meaning that there are vectors allocated in the partition where they are initially reported.

We evaluate our model concerning its robustness when communication failures are present. Communication problems affect the delivery of the statistical vectors that depict the data present in every node in the network. We perform simulations where $\lambda \in \{10\%, 50\%\}$ where λ represents the percentage of lost statistical vectors due to communication problems. In these cases, the proposed mechanism does not update the statistical vectors of its peers and the envisioned decisions are taken on top of their most recent version. Our results are related to the synthetic trace. In Table 2, we present our outcomes for γ and δ metrics. The δ results are lower than the results presented in the above discussed figures (e.g., Fig. 6) depicting how the model is affected by

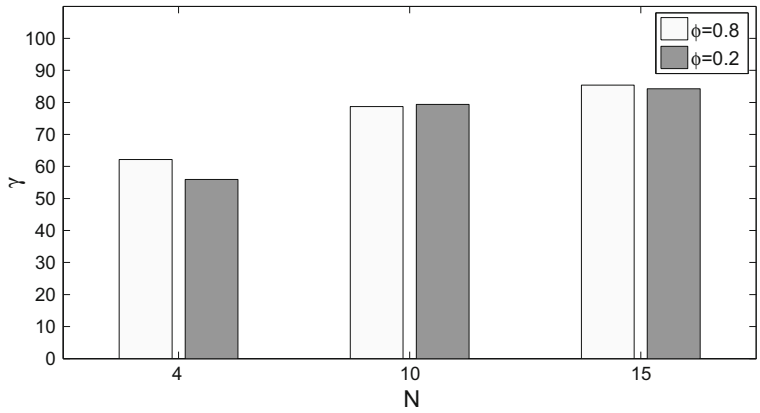

Fig. 13 γ Results for the real trace

Table 2 Results when communication problems are present

N	$\lambda = 10\%$		$\lambda = 50\%$	
	γ	δ	γ	δ
10	88.14	71.70	90.27	75.29
50	97.94	61.67	97.95	68.97
100	99.00	70.41	99.21	73.07

the lost statistical vectors. However, the difference with the previous results is low. The proposed model can still allocate correctly the incoming vectors. Concerning the γ results, we observe a similar behaviour as concluded in the above discussed figures. Comparing the results for different λ realizations, we observe that when 50% of the statistical vectors are lost, the majority of the incoming vectors will be placed in different partitions from the partition where they are initially reported. This result stands even for a low number of partitions (e.g., $N = 10$).

In order to compare our model with a centralized approach, we perform a set of experiments to deliver the time required for concluding the decision process for a single vector. Recall that the proposed model manages the incoming vectors at the time they are reported in a node. A centralized approach should wait to collect all the reported vectors, identify and eliminate the outliers and, finally, to separate data into a number of partitions. For centralized approaches, we focus on the summary of the time required for the last two steps (outliers detection and data separation). It becomes obvious that our model is not affected by the time required for the first step as the incoming vectors are processed immediately. Starting from the data separation techniques, in [13], we can see that the authors compare three data separation techniques. The first scheme is proposed by the authors and it firstly partitions the data along their feature space, and apply the parallel block coordinate descent algorithm for distributed computation; then, it continues to partition the data along the sample space, and adopt a novel matrix decomposition and combination approach for distributed processing. The authors evaluate the model for four (4) datasets; two of them are adopted for

classification purposes (D1, D2) and two are adopted for regression (D3, D4). The average time requirements (we provide the results depicting the time required per vector) are: (1) for D1: 1.61 s; (2) for D2: 0.0059 s; (3) for D3: 0.00059 s; (4) for D4: 0.00047 s. In addition, the authors provide results for two more models, i.e., the Parallelizing Support Vector Machines on Distributed Computers (P-SVM) model [46] and the Consensus-Based Distributed Support Vector Machines (CB-DSVM) model [11]. The results for the P-SVM are: (1) for D1: 0.0046 s; (2) for D2: 0.0139 s; (3) for D3: 0.0051 s; (4) for D4: 0.0044 s. The CB-SVM results are as follows: (1) for D1: 0.00064 s; (2) for D2: 0.0162 s; (3) for D3: 0.0027 s; (4) for D4: 0.0025 s. In the time required for data separation, we should add the time devoted to the pre-processing and the outliers detection steps. In [16], the authors provide a comparison for various outlier detection techniques. In these results, the average required time per vector fluctuates from 0.000023 to 0.00109 s. In our model, the total processing time per vector is 0.0036 s for the synthetic trace and 0.000075 s for the real trace. We observe that when we adopt the real trace, the proposed model outperforms all the aforementioned techniques.

6 Conclusions and future work

Current ICT applications involve huge volumes of data produced by numerous devices. Data are reported through streams and stored in multiple locations. For facilitating the parallel management, a number of data partitions are available where processing tasks are realized. Our effort aims to secure the quality of data stored in each partition through the management of data accuracy. We propose the use of a probabilistic and an uncertainty management mechanism that decides if the incoming data 'fit' to the available partitions. The uncertainty is managed through a fuzzy logic controller to derive the final decision related to the allocation of data. The proposed mechanism checks the similarity of the incoming data with the available partitions and if the accuracy is secured, it decides the appropriate partition where data will be allocated. Our mechanism is distributed, thus, the incoming data may be transferred to the correct partition. The proposed approach is characterized by simplicity while being capable of identifying the correct partition for placing the data. The aim is to reduce the dispersion of data, thus, to increase the accuracy and data quality. Our future research plans involve the definition of an intelligent scheme for selecting the most significant variables in the incoming data to reduce the dimensions and enhance the efficiency of the approach. In addition, the cost of the allocations will be studied especially when data should be transferred in another partition compared to the partition where they are initially reported. Our plans also involve the study of the trade off between the quality of data and the cost of the allocation in a remote partition.

Acknowledgements This work is funded by the EU/H2020 Marie Sklodowska-Curie Action Individual Fellowship (MSCA-IF-2016) under the Project INNOVATE (Grant No. 745829).

References

1. Acuna E, Rodriguez C (2005) An empirical study of the effect of outliers on the misclassification error rate. Trans Knowl Data Eng 17:1–21
2. Alferes J, Poirier P, Lamaire-Chad C, Sharma AK, Mikkelsen PS, Vanrolleghem PA (2013) Data quality assurance inmonitoring of wastewater quality: univariate on-line and off-linemethods. In: Proceedings of the 11th IWA conference on instrumentation control and automation. pp 18–20
3. Alipour J, Ahmadi M (2017) Dimensions and assessment methods of data quality in health information systems. Acta Med Mediterr 33:313–320
4. Arapoglou R, Kolomvatsos K, Hadjiefthymiades S (2010) Buyer agent decision process based on automatic fuzzy rules generation methods. In: Proceedings of the 2010 IEEE world congress on computational intelligence (WCCI 2010), FUZ-IEEE, July 18th–23rd, Barcelona. pp 856–863
5. Batini C, Rula A, Scannapieco M, Viscusi G (2015) From data quality to big data quality. J Database Manag 26(1):60–82
6. Cai L, Zhu Y (2015) The challenges of data quality and data quality assessment in the big data era. Data Sci J 14(2):1–10
7. Cong G, Fan W, Geerts F, Jia X, Ma S (2007) Improving data quality: consistency and accuracy. In: Proceedings of the VLDB, Vienna, Austria. pp 1–12
8. Das TK, Kumar PM (2013) Big data analytics: a framework for unstructured data analysis. Int J Eng Technol 5(1):153–156
9. Do CB, Batzoglou S (2008) What is the expectation maximization algorithm? Nat Biotechnol 26(8):897–899
10. Eurostat (2007) Handbook on data quality assessment methods and tools. European Commission, Luxembourg
11. Forero P, Cano A, Giannakis G (2010) Consensus-based distributed support vector machines. JMLR 11:1663–1707
12. Gao J, Xie C, Tao C (2016) Big data validation and quality assurance-issues, challenges and needs. In: Proceedings of the IEEE symposium on service-oriented system engineering (SOSE). https://doi.org/10.1109/SOSE.2016.63
13. Guo H, Zhang J (2016) A distributed and scalable machine learning approach for big data. In: Proceedings of the 25th international joint conference of artificial intelligence, New York
14. Han J, Kamber M, Pei J (2012) Data mining, concepts and techniques, 3rd edn. Morgan Kaufmann Publishers, Burlington
15. Halkidi M, Varzigiannis M (2001) Clustering validity assessment: finding the optimal partitioning of a dataset. In: Proceedings of the IEEE international conference on data mining, San Jose, USA,
16. Hasani Z (2017) Robust anomaly detection algorithms for real-time bigdata: comparison of algorithms. In: Proceedings of the 6th Mediterranean conference on embedded computing (MECO)
17. Immonen A, Paakkonen P, Ovaska E (2015) Evaluating the quality of social media data in big data architecture. IEEE Access 3:2028–2043
18. Karjee J, Jamadagni HS (2011) Data accuracy model for distributed clustering algorithm based on spatial data correlation in wireless sensor networks. Networking and internet architecture. arXiv:1108.2644
19. Last M, Kandel A (2001) Automated detection of outliers in real-world data. In: Proceedings of the 2nd international conference on intelligent technologies
20. Loshin D (2011) Monitoring data quality performance using data quality metrics. Informatica, The Data Integration Company, white paper
21. Majewska J (2015) Identification of multivariate outliers-problems and challenges of visualization methods, Studia Ekonomiczne. Zeszyty Naukowe, Uniwersytetu Ekonomicznego w Katowicach, No 247
22. Management Group on Statistical Cooperation (2014) Report of the sixteenth meeting. European Commission, Eurostat, vol Doc., p MGSC/2014/14
23. Mendel JM (2007) Type-2 fuzzy sets and systems: an overview. IEEE Comput Intell Mag 2(2):20–29

24. Mendel JM (2001) Uncertain rule-based fuzzy logic systems: introduction and new directions. Prentice-Hall, Upper Saddle River
25. Merino J, Caballero I, Rivas B, Serrano M, Piattini M (2016) A Data quality in use model for big data. Future Gener Comput Syst 63:123–130
26. Mishra S, Suman AC (2016) An efficient method of partitioning high volumes of multidimensional data for parallel clustering algorithms. Int J Eng Res Appl 6(8):67–71
27. Mohammed AO, Talab SA (2015) Enhanced extraction clinical data technique to improve data quality in clinical data warehouse. Int J Database Theory Appl 8(3):333–342
28. Navathe S, Ceri S, Wiederhold G, Dou J (1984) Vertical partitioning of algorithms for database design. ACM Trans Database Syst 9:680–710
29. Nelson RR, Todd PA, Wixom BH (2005) Antecedents of information and system quality: an empirical examination within the context of data warehousing. J Manag Inf Syst 21(4):199–235
30. Page ES (1954) Continuous inspection scheme. Biometrika 41(1/2):100–115
31. Pipino LL, Lee YW, Wang RY (2002) Data quality assessment. Commun ACM 45(4):211–218
32. Rao D, Gudivada VN, Raghavan VV (2015) Data quality issues in bigdata. In: Proceedings of the IEEE international conference on bigdata, Santa Clara, CA, USA
33. Rosemary Tate A, Kalra D, Boggon R, Beloff N, Puri S, Williams T (2014) Data quality in European primary care research databases. Report of a workshop held in London September 2013. In: Proceedings of the IEEE-EMBS international conference on biomedical and health informatics (BHI), Valencia, Spain
34. Sacca D, Wiederhold G (1985) Database partitioning in a cluster of processors. ACM Trans Database Syst 10:29–56
35. Salloum S, He Y, Huang JZ, Zhang X, Emara T (2018) A random sample partition data model for big data analysis. arXiv:1712.04146
36. Shi W, Cao J, Zhang Q, Li Y, Xu L (2016) Edge computing: vision and challenges. IEEE Internet Things 3(5):637–646
37. Sidi F, Panahy PHS, Affendey LS, Jabar MA, Ibrahim H, Mustapha A (2012) Data quality: a survey of data quality dimensions. In: Proceedings of the international conference on information retrieval and knowledge management (CAMP), Kuala Lumpur, Malaysia, pp 300–304
38. Son LH (2015) DPFCM: a novel distributed picture fuzzy clustering method on picture fuzzy sets. Expert Syst Appl 42(1):51–66
39. Truong H, Karan M (2018) Analytics of performance and data quality for mobile edge cloud applications. In: Proceedings of the IEEE international conference on cloud computing, workshop: cloud and the edge San Francisco, USA
40. Umar A, Karabatis G, Ness L, Horowitz B, Elmagardmid A (1999) Enterprise data quality. Inf Syst Front 1(3):279–301
41. Urbano F, Basille M, Cagnacci F (2014) Data quality: detection and management of outliers, chapter. In: Spatial database for GPS wildlife tracking. Data: a practical guide to creating a data management system with Postgre SQL/Post GIS and R. Springer
42. Van den Berghe S, Van Gaeveren K (2017) Data quality assessment and improvement: a Vrije Universiteit Brussel case study. Procedia Comput Sci 106:32–38
43. Wigan MR, Clarke R (2013) Big data's big unintended consequences. IEEE Comput 46(6):46–53
44. Wu D (2012) On the fundamental differences between interval type-2 and type-1 fuzzy logic controllers. IEEE Trans Fuzzy Syst 20(5):832–848
45. Wanger C (2013) Juzzy—a java based toolkit for type-2 fuzzy logic. In: Proceedings of the IEEE symposium on advances in type-2 fuzzy logic systems, Singapore
46. Zhu K, Wang H, Bai H, Li J, Qiu Z, Cui H, Chang E (2008) Parallelizing support vector machines on distributed computers. In: Proceedings of NIPS 20