An energy-aware data cleaning workflow for real-time stream processing in the internet of things

Egberto A. R. de Oliveira[1], **Flavia C. Delicato**[2], **Marta Mattoso**[1]

[1]COPPE / Universidade Federal do Rio de Janeiro (UFRJ) – Rio de Janeiro – RJ – Brazil

[2]Instituto de Computação / Universidade Federal Fluminense (UFF) – Niterói – RJ – Brazil

egberto@cos.ufrj.br, fdelicato@ic.uff.br, marta@cos.ufrj.br

Abstract. *The Internet of things (IoT) has recently transformed the internet, enabling the communication between every kind of objects (things). The growing number of sensors and smart devices enhanced data creation and collection capabilities and led to an explosion of generated data in the form of Data Streams. Processing these data streams is complex and presents challenges and opportunities in the stream processing field. Due to the inherent lacking of accuracy and completeness of sensor generated data, the quality of raw data is often poor. Data cleaning tasks are required to help increasing the quality of the data being processed in an IoT application. This work proposes a data stream processing workflow for IoT to be deployed at the edge of the network. It performs a fast data cleaning with low power consumption from edge and sensor nodes. The edge computing paradigm is used to bring the data cleaning task closer to the data sources and allow actions to be triggered immediately. In addition, an energy-aware data collection component is designed to reduce the network traffic and, as a consequence, decrease the power consumption of the network devices. The proposed workflow enables the deployment of long running real-time processing systems on remote outdoor environments.*

1. Introduction

The Internet of things (IoT) is transforming the internet in recent years, enabling the communication between every kind of objects (things) and creating a vision of "anytime, anywhere, any media, anything" communications [Atzori et al. 2010]. The growing number of sensors and smart devices led to an explosion of volume, variety and velocity of generated data, empowering a new way of value creation to people and corporations [Dias de Assunção et al. 2018]. In addition to this increased number of devices, technological advances have also enhanced their data collection capabilities, resulting in an even larger amount of data generated in the form of continuous streams also known as Data Streams [Karkouch et al. 2016]. The processing of these "firehoses" of data from existing and newly emerging applications presents a challenge and an opportunity in the stream processing field.

Processing data streams is complex because of several factors. Often, there is no control over the order or frequency of streamed data, which is transient or non-persisted. They are also potentially unbounded in size, have a high volume of information and can demand stringent processing requirements such as real time responses. The demand for computational resources capable of processing large volumes of data has historically been an obstacle for creating high volume and/or high speed data processing solutions. Cloud computing based approaches are widely adopted in IoT systems. The data is pushed to the cloud to be processed and the outcome is delivered back to the local system. However, the internet backbone is unable to meet the requirements of low latency to transport a huge amount of data coming at a high speed. This creates a communication bottleneck and leads to the search for other approaches, non-cloud based, to manipulate IoT-generated data [Janjua et al. 2019].

In this work, we aim to meet real-time processing requirements in data streams produced by IoT devices. To deal with the network bandwidth vs. data production bottleneck, we decided to bring the processing physically closer to the data source making use of a new paradigm called Edge Computing [Dautov et al. 2018]. It is an effort to involve decentralized agents to perform necessary processing, which can reduce the burden on centralized processing units [Shi and Dustdar 2016]. Edge Computing is potentially useful and has been adopted in several domains such as smart home, smart city, smart health, and smart transportation. In these applications, data is processed by an edge device such as a gateway to extract meaningful information from it and take necessary actions immediately [Janjua et al. 2019].

Addressing data quality in the IoT context is challenging due to the inherent lacking of accuracy and completeness of sensor generated data. Accuracy is affected by incorrect readings or even fails from sensors. Completeness is compromised whenever an actual event is lost for various reasons and variables under which the system might be exposed such as: limited energy sources, weather conditions and more. Information and decisions derived from such data will also be subject to failure [Klein and Lehner 2010]. Identifying errors/inconsistencies on a data stream is crucial to improve the accuracy of the data being processed. These errors/inconsistencies are often called outliers, which are readings considered outside the regular state of the data being collected. For example, data points that differ significantly from others in a data set and can represent either errors or events of importance to the application [Karkouch et al. 2016]. Clustering and classification algorithms, which are well known machine learning techniques, are frequently used for outlier detection in several use cases [Aggarwal 2013].

Among the diverse outlier detection techniques available, those based on statistical models and artificial intelligence (AI) stand out [Karkouch et al. 2016]. Both groups of techniques are often dependent of compute intensive algorithms. This might represent an issue to IoT applications, in which it is quite common the presence of constrained devices, in terms of memory, CPU, and more [Atzori et al. 2010]. This compute intensive behavior, as a consequence, imposes another challenge regarding sensing infrastructure which consists in keeping the sensors working as long as possible. This is critical specially when deploying real-time processing systems on remote outdoor environments such as forests, open fields and watercrafts. In such environments there is no access to continuous sources of electricity thus requiring the use of batteries, solar panels or other types limited power sources.

Frameworks for data stream processing usually involve multi-layer compositions with loosely coupled components to facilitate maintenance and provide scalability and availability. In general, data collectors are responsible for the acquisition and preprocessing of raw data from the most diverse sources, transferring it to processing engines with a specific (business logic) purpose to deliver value-added information to a final consumer. Messaging systems, built on specific communication protocols, and different types of technologies for data storage are also part of the range of key components found in these frameworks [Dias de Assunção et al. 2018].

Due to the inherent lacking of accuracy and completeness of sensor generated data, the quality of raw data is often poor. Data cleaning tasks are required to help increasing the quality of the data being processed in an IoT application. This work proposes a data stream processing workflow to be deployed at the edge of the network. It performs a fast data cleaning with low power consumption from edge and sensor nodes. It serves as an input for other processing components or systems which provide real-time responses and decisions for applications. To achieve such a major goal, the secondary objectives of the proposed workflow are:

- To provide an energy-aware data collection component to reduce the network traffic and, as a consequence, the power consumption of the sensor and edge nodes;
- To implement a density-based clustering component to efficiently perform the data cleaning task by quickly identifying and removing outliers from the data stream;
- To deliver a curated secondary data stream output which can be consumed by business applications, services or additional workflow tasks with real-time processing requirements.

The major benefit expected by adopting the proposed workflow is the capability of deploying long running real-time processing systems on remote outdoor environments.

The rest of this article is organized as follows. Section 2 presents relevant works which individually tackle the issues we aim to solve together. Section 3 describes the proposed workflow. A proof of concept to evaluate the proposal is presented and discussed in Section 4. Finally, Section 5 concludes the paper and provides additional information about the ongoing and future work on this research.

2. Related work

Relevant proposals addressing separately the issues of real-time responses, data accuracy and completeness or power consumption can be found in the data stream processing fields. However, to the best of our knowledge, no solution tackling these three concerns together has been found so far. This makes it difficult to deploy solutions that can efficiently respond to real-time events in power-constrained environments, such as a forest fire suppression system, a malfunction detection system on small ships, etc.

IRESE [Janjua et al. 2019] presents an outlier (denoted as ”rare-event” in the paper) detection system that applies unsupervised machine learning techniques at the gateway to identify events on audio data streams. A data framing module takes buffered data and breaks it into smaller pieces (frames). It is a tumbling window, which moves over the buffered data stream in a way that two consecutive windows do not overlap with each other. Then a two-stage strategy is applied. In the first stage, the high speed data is processed with BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) in real-time to quickly extract statistical information from it in the form of micro-clusters. It can indicate the presence of outliers in the data steam. In the second stage, the data is further processed and merged together with Agglomerative Clustering. The final output is always in the form of two clusters: cluster A is dense and containing data points reflecting normal behavior. Cluster B contains rare-events (if they exist) which is an outlier and different from other events occurring in that specific interval of buffered data. Despite the significant results achieved in terms of data quality and real-time response, no concern regarding energy consumption is mentioned. Therefore, this solution might not be feasible on environments with limited power sources, which is a major concern for our work.

Virtual Sensing Framework [Sarkar et al. 2016], or simply VSF, is a data collection framework for Wireless Sensor Networks (WSN). WSN are groups of tiny multi-functional sensor nodes that communicate in short distances to monitor an environment over a collaborative effort [Akyildiz et al. 2002] and can be considered as part of IoT. The proposal aims at reducing activities of the sensor nodes to decrease the overall energy consumption of the network while keeping the desired redundancy for the sensing requirements. Virtual sensors (VS) are logical representations of physical sensors (PS) at the gateway node. The proposed activity reduction schema assumes that a predicted value for a VS must consider the past readings of its correspondent PS and also consider cross correlations among the nodes to handle long dormant periods of PS. These correlations are obtained with an adaptive node correlation schema which does not assume a prior knowledge such as physical placement of the nodes to determine it. The goal is to select a minimal number of active nodes (maximal sleeping node policy) so that the union of the correlated companions of all the active nodes contains all the nodes in a WSN. Despite an efficient data collection scheme is introduced to reduce the overall energy consumption on a WSN, no data stream processing capability is mentioned. Time intervals in the order of hours are considered to "virtualize" the data using several statistical methods to perform calculations. Bringing this approach "as is" to a data stream processing scenario, where the data comes on a high rate, is not possible. Our proposal takes the central idea behind the "Virtual Sensing Framework" and makes it simple as described in the next section.

A high-dimensional data cleaning method for mobile edge nodes on WSNs is presented in [Wang et al. 2019]. It is an adaptive mechanism which combines machine learning techniques with the edge computing paradigm to optimize the cleaning model in real-time. The authors highlight the importance of the data cleaning task on the Industrial IoT scenario and how an edge computing approach can significantly contribute to reduce energy consumption of sensor nodes and to accelerate the cleaning speed of abnormal data. The proposed solution is presented as a three-tier architecture, which would be an evolution from a traditional two-tier approach. It brings the task of cleaning the data to the middle layer (edge) and relieves the workload of the sensor nodes that would be responsible for this task in the two-layer architecture. The authors also propose a new algorithm to clean data using the mobile edge nodes called Angle-based Oultlier Detection (ABOD). It uses a combination of the variance of angles and distance between data points to calculate factors and identify outliers. An online machine learning algorithm is also present to optimize the parameters of the cleaning model, thus adding an adaptive behaviour to the proposed solution. It is important to underline that in Mobile Cleaning [Wang et al. 2019] the edge computing paradigm is applied to handle power consumption, unlike what happens in IRESE [Janjua et al. 2019] where it is used to address real-time processing. This might be an indicative of how representative edge computing can be on the data stream processing for the IoT field.

A novel image superpixel segmentation using Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is introduced in [Shi and Dustdar 2016]. Traditional segmentation algorithms tend to be costly in terms of computation and speed. This work introduces DBSCAN due to its reduced computational cost and its ability to find arbitrarily shaped clusters. To achieve the stated goals, an adaptation is made on the conventional DBSCAN by reducing its search range. While the original approach would search an entire image, the proposed approach limits the search range in the rhombus neighbor region around the seed. This creates a local search strategy which makes each superpixel with a uniform shape as possible. The results show that the proposed model is able to achieve the same results as traditional models in less time. This is a good indication that DBSCAN can be a good fit for other scenarios which demand smaller response times such as real-time data stream processing.

The works presented in this section propose effective solutions to address requirements of real-time responses, data accuracy and completeness or power consumption, but none of them tackle all these three concerns together. Combining these three requirements in the same solution is complex because the approach used to solve one problem can negatively impact the solution of another. For example, statistical methods based on intensive computing can efficiently solve the problem of lack of accuracy or completeness of the data but demand a high energy consumption of the devices. The contribution of our work consists of combining approaches such as those described, promoting the necessary adaptations so that the three requirements are jointly met.

3. Proposed workflow

Considering the heterogeneity of IoT environments, there is an increasing number of proposed architectures which have not yet converged to a reference model [Atzori et al. 2010]. From the pool of the proposed models, in this work we consider a three-tier architecture as proposed in [Li et al. 2017]. In such architecture, the first tier is the Things tier and it comprises physical sensors and embedded devices responsible for collecting information from the monitored environment and generating data to the IoT system. The second is the edge/fog tier composed of devices located physically close to the things and responsible for less compute intensive tasks such as preprocessing the incoming data. The upper tier is the cloud, encompassing robust devices (data centers) capable of handling more compute intensive processing tasks and/or permanently store relevant data (archiving).

To provide a fast and energy-aware data cleaning method for IoT streamed data, we propose a workflow to be deployed at the thing and edge tiers. The design of this workflow starts by following the guidelines of the online data stream processing workflow introduced in [Dias de Assunção et al. 2018]. Since the reference is a general model, the particularities defined for the context of this work are presented in Fig. 1. Sensors embedded in IoT devices are the data sources and the starting point of the workflow. The grey rounded-bordered boxes represent the workflow activities with their inner tasks shown as white rectangles. The full arrows represent data flow while the dashed ones represent triggered actions. A generic consumer component stands as the end of the workflow. Each component is further described in the following subsections.

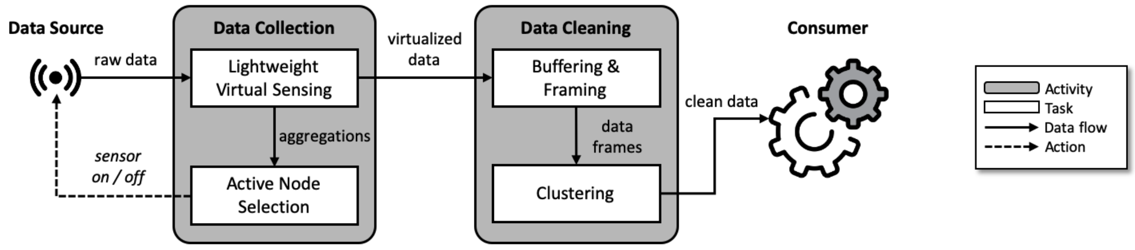

Figure 1. Overview of the proposed data cleaning workflow for real-time data stream processing.

3.1. Data source

A group of several sensors placed on the bottom tier (things) which generates data about a monitored or observed environment (or object, entity, etc.) serves as input stream of raw data for the workflow. It is expected that more than one sensor is used to collect information about a single entity to provide redundancy on sensor readings and thus improve the completeness of the generated data.

3.2. Data collection

Considering that idle-listening and packet overhearing are significant sources of energy drain and still taking into account that data transmission and reception require higher energy compared to sensing [Sarkar et al. 2016], reducing both the node activities and the communication between nodes is a reasonable way to reduce the overall power consumption on a network. In an IoT context this number can easily scale to tens, hundreds or even thousands of devices. Therefore, this activity in the proposed workflow is implemented by two components which act together over time to disable a subset physical sensors and predict their values directly on the edge nodes. This activity aims to reduce the communication burden for both sensor and edge nodes. Further details for these tasks are described below. The output of this data collection activity generates a secondary data stream (that we call virtualized data) that will serve as an input for the next activity.

3.2.1. Lightweight Virtual Sensing (LVS)

The Virtual Sensing Framework [Sarkar et al. 2016] provides an efficient way to implement virtual sensors to reduce the communication between sensor nodes and their consumers without compromising data accuracy as described in section 2. The idea is to create virtual sensors corresponding to each existing physical sensor where the virtual sensor readings can, at a given moment, be obtained through the reading made by the corresponding physical sensor and, in another moment, be forecasted based on a prediction model.

Although the latest published release of VSF features a prediction model that combines complex tasks such as adaptive node correlation, linear regression and adaptive parameter update, earlier publications of the same framework implemented simplified prediction models that also achieved good results. Taking into account the edge computing approach, which is often composed of resource-constrained devices, we propose a simplified prediction model in our workflow. This choice is to keep the implementation as lightweight as possible. The predicted value for the virtual sensor will always be computed as the last cached reading of its correlated physical sensor. The rules that define the lightweight virtual sensing task are described below:

1. An active physical sensor only sends the reading to the sink if its value is different (or higher than a predefined threshold) from its last reading;
2. A virtual sensor defines its values as the last reading from its physical correspondent.

3.2.2. Active node selection

An active node selection is an important component responsible for defining from time to time which sensor nodes will be active and which will be on a dormant state. Only the active nodes will send their readings to processing nodes. Similarly to what happens for LVS, this task has been simplified from VSF [Sarkar et al. 2016] to make the implementation lighter and more adapted to resource-constrained devices. Assuming homogeneous power consumption between sensor nodes, a simple round robin algorithm is responsible for ensuring a uniform distribution of the workload based on a predefined time interval (turn):

1. A parameter will indicate a percentage (%) of the sensors which will be disabled each turn;
2. The node selector component:
   (a) Organizes sensors on a queue (if it does not exist – 1st turn);
   (b) Puts the actual “dormant” nodes on the end of the queue;
   (c) Calculates the absolute number (X) of sensors to disable;
   (d) Pops the first X sensors from the queue and puts on a “dormant” state;
3. Active sensors send readings to the sink;
4. Dormant sensors do not send readings to the sink.

3.3. Data cleaning

Sensor generated data, regardless of physical or virtual origin, need to be accurate and complete. Physical sensors fail. Incorrect readings are a reality in the IoT context and they need to be identified and discarded as best as possible to improve the quality of the information and decisions based on the acquired data. A common practice in sensed environments is to use a group of sensors to monitor the same entity, providing the necessary redundancy that makes it easier to identify incorrect or noisy readings. The proposed workflow implements a buffering and framing technique on the subsets of sensor readings at a given time and applies a clustering algorithm to identify outliers as errors and discard them. The output of this activity is a new stream of more accurate data that will serve as an input for consumers (application, services, etc.).

3.3.1. Buffering and framing

Data streams are continuous flows of isolated data points. In an IoT use case with sensor generated data, these data points are represented by sensor readings. No data point can be considered an outlier on an individual basis analysis. Thus, defining a way to group and analyze these data points is a major concern when designing an outlier detection task. Buffering incoming data on a predefined length or time interval to create frames (windows) is a common approach to perform operations on data streams [Tsai et al. 2014]. This is also adopted in this task so that these data frames can be transmitted to the next task which will perform the outlier detection itself.

3.3.2. Clustering

Clustering is a problem widely studied in the data mining and AI literature. However, it is more difficult to adapt arbitrary clustering algorithms to the context of data stream processing. Its potentially unbounded in size feature makes this adaptation especially complex [Aggarwal 2013]. K-means is certainly one of the best-known clustering and also the starting point for a number of variations tailored for stream processing [Tsai et al. 2014]. However, the number of clusters is an input parameter, making such an algorithm unsuitable for some situations. The micro-clustering technique is more effective and versatile than K-means for the context of data streams [Aggarwal 2013]. Density-based techniques are possible solutions to the problem of clustering in an IoT scenario [Aggarwal 2013]. Techniques like this, in turn, are able to determine the number of clusters as an output. Therefore, in our work we apply density-based spatial clustering of applications with noise (DBSCAN) on the incoming data frames to detect and discard outliers and aggregate the remaining readings into a new accurate data stream output in a real-time fashion.

3.4. Consumer

Assuming the proposed workflow is executed as a pre-processing step of a data stream processing, it is not expected that its output is directly consumed by end users or visualization tools since it is still a data stream. The expected consumer for this output would be an application, service or any software component capable of ingest and process streaming data.

4. Evaluation

In this section we perform a proof of concept (PoC) for the proposed workflow in order to evaluate (i) how efficient the edge computing paradigm can be in terms of low latency when compared to the cloud computing paradigm; (ii) DBSCAN as a clustering approach for the outlier detection task on the data cleaning activity and (iii) the proposed lightweight virtual sensing approach.

All components were implemented using the Flask framework (Python 3), in the form of web APIs under HTTP protocol. A processor node is implemented to get data from a remote sensor node, perform data collection and data cleaning activities and trigger a new request to the sensor node with the time series data to be further processed as shown in Fig. 2. Data is processed in windows (or frames), which are subsets of the streamed data collected under a predefined time interval. The size of each window (S) and the number of windows processed (N) during the runtime are the two parameters considered to measure and compare the workflow's performance on each scenario.

In terms of infrastructure, the specifications for the edge and cloud devices mentioned on the subsections to follow are described as follows:

- Cloud device: an Amazon EC2 instance t3a.large (2*2,5GHz vCPU / 8GB RAM);
- Edge device: a Raspberry Pi 3 model B+ (Quad core 1,4GHz CPU / 1GB RAM);
- Sensor device: a group of seven DHT11 sensors (temperature + humidity digital output) attached to a Raspberry Pi 3 model B+.

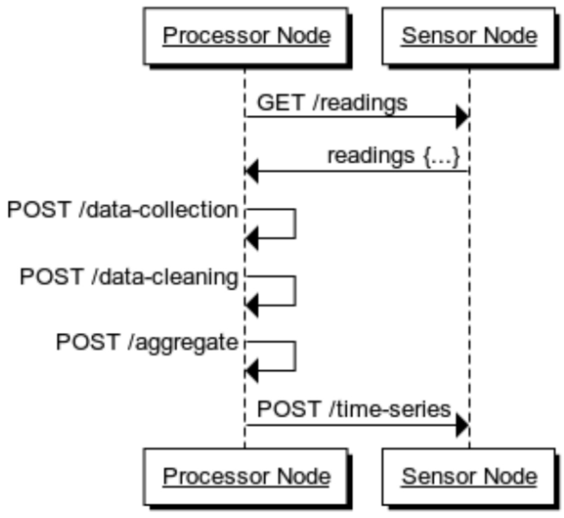

Figure 2. The interaction between the nodes implemented for this PoC.

4.1. Edge computing paradigm

We aim to evaluate how efficient the edge computing paradigm can be to achieve real time responses when compared to cloud computing. The proposed workflow has been run multiple times with the same parameter set and the processor node role runs both on at the edge and the cloud nodes specified in this section.

The goal is to measure and compare the total runtime of the workflow in two different parameter set:

1. Fixed number of windows, ramping up the window size;
2. Fixed window size, ramping up the number of windows.

Despite the remarkable difference on computing power capabilities (RAM and CPU) between the edge and cloud nodes, Fig. 3 shows an edge computing average processing time of approximately half of cloud computing average processing time for both cases. This is in line with what was said in section 1: the data transportation bottleneck has more impact on response times than the computational capacity of the devices involved in processing data coming at a high speed.

4.2. Outlier detection with DBSCAN

4.2.1. Speed

To assess the feasibility of using DBSCAN as an outlier detection technique for the data cleaning activity of the workflow, the algorithm has been executed several times for different sized data sets, still considering both edge and cloud computing scenarios. Fig. 4 shows how faster is the cloud node processing when compared to edge node processing, which is expected considering that the algorithm is processed centrally (on a single

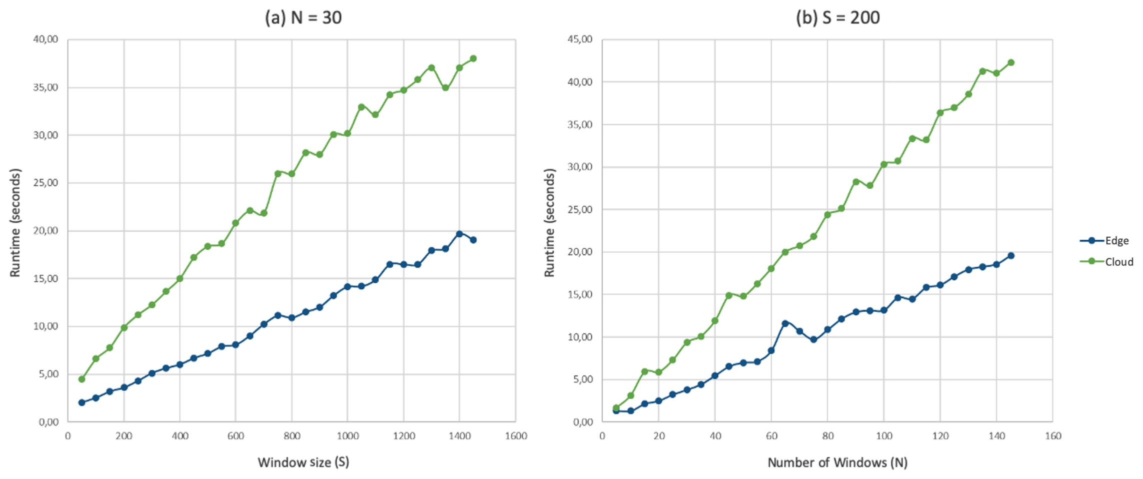

Figure 3. Runtime comparison with fixed number of windows (a) and window size (b).

node) rather than distributed. The node with the highest computational capacity runs the algorithm faster. However, proving this superiority is not the purpose of this test, but identifying the data set size thresholds for running DBSCAN on a millisecond basis for each scenario. These thresholds can later be used to define window sizes on future implementations of the proposed workflow. It is also important to mention that the edge node was not able to run DBSCAN for data sets larger than 13,500 data points.

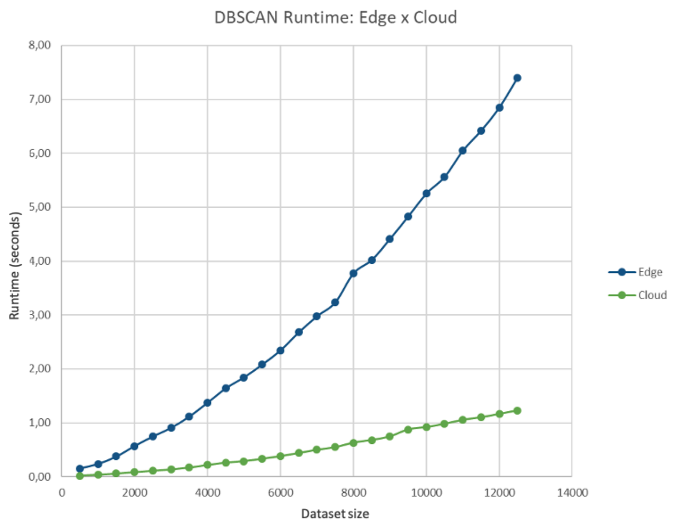

Figure 4. DBSCAN runtime on edge and cloud nodes for different dataset sizes.

DBSCAN runs in less than 1 second for data sets with up to 3,000 data points on the evaluated edge node and for data sets with up to 10,000 data points on the evaluated cloud node. For data sets with up to 2,000 data points, the algorithm runs extremely fast (up to 0,5s), which means that, on a real use case, these devices would be able to identify outliers on a frame of 2,000 sensor readings in half a second. These numbers indicate that DBSCAN is a fast outlier detection algorithm for data streams generated by IoT devices.

Considering the big difference of DBSCAN runtime in both devices, this DB-

SCAN standalone evaluation reinforces what was stated in section 4.1: the data transportation bottleneck has more impact on response times than the computational capacity of devices involved. Despite being capable of processing 10,000 data points in less than 1 second on a cloud node, the internet backbone would take too much time to transfer this data from its source and bring a high latency to the entire workflow, which would not achieve real time processing requirements.

4.2.2. Accuracy

Data collected from the sensors (temperature and humidity) on a time frame of 60 minutes were used to check the accuracy of DBSCAN as the engine of the outlier detection task. In Fig. 5 the red line gives an idea of how outliers can negatively impact data analysis while the green line shows how the same data set looks like after being cleaned using DBSCAN. Further tuning can still be applied by adjusting DBSCAN parameters, but the results achieved so far were enough for the PoC purposes. DBSCAN can be taken as a fast and accurate outlier detection technique for data stream processing.

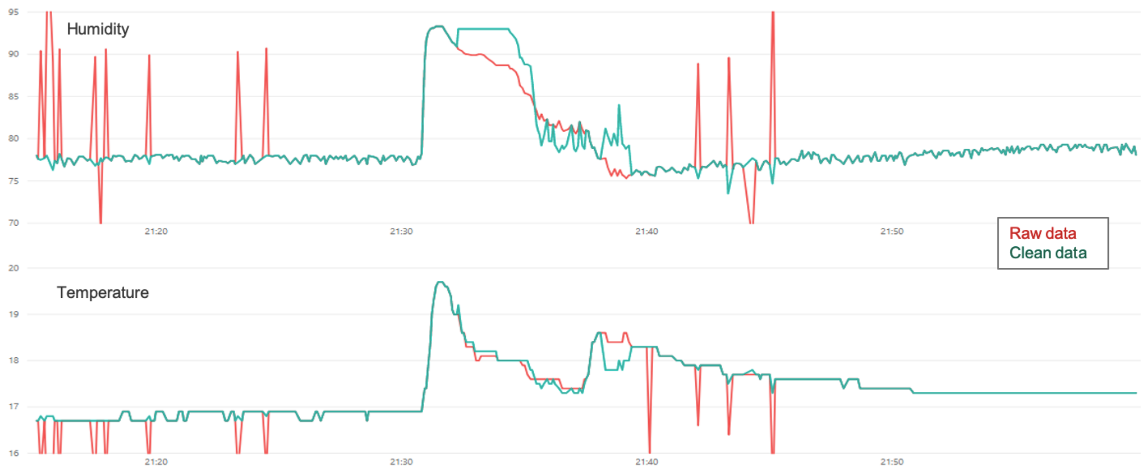

Figure 5. DBSCAN as an outlier detection technique for data cleaning.

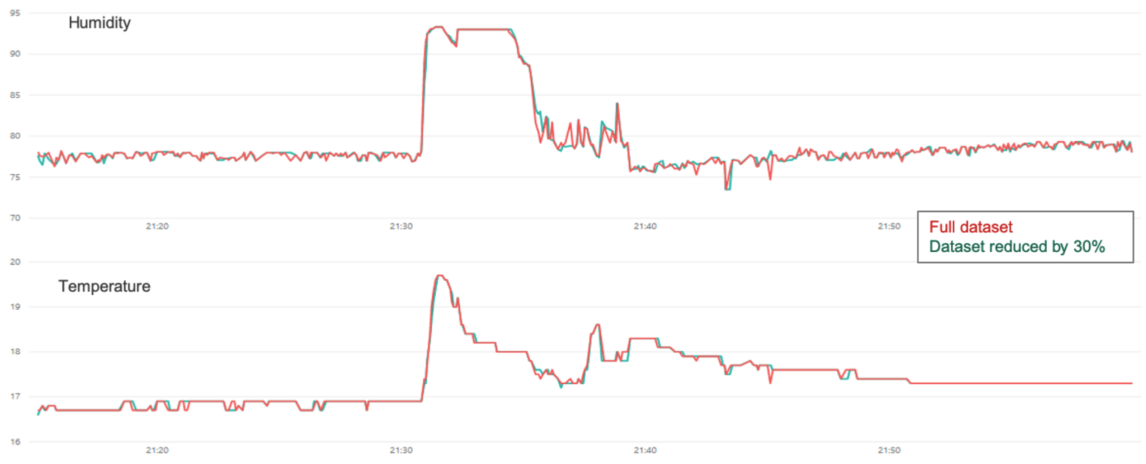

Figure 6. Simplified version of virtual sensing implemented for the PoC.

4.3. Lightweight Virtual Sensing

The same data collected from DHT11 sensors and used on section 4.2.2 were also used to evaluate a simplified version of VSF [Sarkar et al. 2016]. The main goal is to testify its efficiency on reducing the communication between sensor and processor nodes without significantly impacting data accuracy. Fig. 6 demonstrates an almost imperceptible difference between the projection of humidity and temperature readings using the full data set (red line) and using a data set with sampling reduced by 30% (green line). These data points are forecasted by LVS described on section 3.2.1. Taking as an assumption that data transmission contributes more to a device's energy drain than computing instructions [Sarkar et al. 2016], we can conclude that this lightweight implementation of VSF enables reduction of device power consumption without significantly affecting the accuracy of the output data of the proposed workflow.

5. Conclusion and Future Work

This work presented an energy-aware data stream processing for IoT. It successfully addresses real time issues by using the edge computing paradigm. Experiments show that accuracy and completeness are improved due to a fast and efficient data cleaning task based on DBSCAN algorithm. Finally, being energy aware, as shown on section 4.3 has reduced the energy consumption on constrained devices without a significant impact on the data accuracy. Based on the experiments described on the sections 4.1 and 4.2.1, it is possible to confirm what is stated on the section 1: the data transportation bottleneck has more impact on response times than the computational capacity of devices involved in processing data coming at a high speed. The tests performed on 4.2 does not guarantee that DBSCAN is the best solution to perform the outlier detection task for the data cleaning activity, but proves it is a fast and accurate alternative, thus a good choice based on the achieved results. Lightweight VSF, which is a simplified version of the Virtual Sensing Framework [Sarkar et al. 2016], described on section 3.2.1 and evaluated on section 4.3 stands-out as a promising technique to achieve a better use of edge devices, in terms of power consumption, without significantly affecting the accuracy of the output data. This is a proven successful approach for reducing network traffic on WSN which was adapted to a real time data stream processing context. Therefore, the proposed workflow might be used to enable the deployment of long running real time data stream processing IoT systems on remote outdoor environments such as forests, open fields and watercrafts, where there is no access to continuous sources of electricity thus requiring the use of batteries, solar panels or other types limited power sources.

This is an incipient and promising study which shall be extended to a more complete asset to promote the development of energy aware real time data stream processing IoT systems. The proposed processing workflow can be evolved into an IoT framework with appropriate abstraction and reuse levels to speed up the development of systems with the common purpose of real time data stream processing on resource constrained edge devices. Additional features such as alternative scheme's for active node selection or the possibility of using overlapping windows would increase the flexibility and usability of the solution. Extending the experiments on energy consumption to fine tune the results achieved with the PoC is an ongoing work.

Acknowledgment

This work has been partially funded by Fundação de Amparo à Pesquisa do Estado de São Paulo - FAPESP (grant 2015/24144-7), Fundação de Amparo à Pesquisa do Estado do Rio de Janeiro - FAPERJ (grant 2017/233868) and Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq (grant 434874/2018-3). Marta Mattoso and Flavia Delicato are CNPq Fellows.

References

Aggarwal, C. C. (2013). Mining Sensor Data Streams. In *Managing and Mining Sensor Data*, pages 143–171. Springer US, Boston, MA.

Akyildiz, I., Su, W., Sankarasubramaniam, Y., and Cayirci, E. (2002). Wireless sensor networks: a survey. *Computer Networks*, 38(4):393 – 422.

Atzori, L., Iera, A., and Morabito, G. (2010). The Internet of Things: A survey. *Computer Networks*, 54(15):2787–2805.

Dautov, R., Distefano, S., Bruneo, D., Longo, F., Merlino, G., and Puliafito, A. (2018). Pushing intelligence to the edge with a stream processing architecture. In *Proceedings - 2017 IEEE International Conference on Internet of Things, IEEE Green Computing and Communications, IEEE Cyber, Physical and Social Computing, IEEE Smart Data, iThings-GreenCom-CPSCom-SmartData 2017*.

Dias de Assunção, M., da Silva Veith, A., and Buyya, R. (2018). Distributed data stream processing and edge computing: A survey on resource elasticity and future directions. *Journal of Network and Computer Applications*.

Janjua, Z. H., Vecchio, M., Antonini, M., and Antonelli, F. (2019). IRESE: An intelligent rare-event detection system using unsupervised learning on the IoT edge. *Engineering Applications of Artificial Intelligence*, 84:41–50.

Karkouch, A., Mousannif, H., Al Moatassime, H., and Noel, T. (2016). Data quality in internet of things: A state-of-the-art survey.

Klein, A. and Lehner, W. (2010). Quality and Performance Optimization of Sensor Data Stream Processing. *International Journal on Advances in Networks and Services*.

Li, W., Santos, I., Delicato, F. C., Pires, P. F., Pirmez, L., Wei, W., Song, H., Zomaya, A., and Khan, S. (2017). System modelling and performance evaluation of a three-tier Cloud of Things. *Future Generation Computer Systems*.

Sarkar, C., Rao, V. S., Venkatesha Prasad, R., Das, S. N., Misra, S., and Vasilakos, A. (2016). VSF: An Energy-Efficient Sensing Framework Using Virtual Sensors. *IEEE Sensors Journal*, 16(12):5046–5059.

Shi, W. and Dustdar, S. (2016). The Promise of Edge Computing. *Computer*.

Tsai, C.-W., Lai, C.-F., Chiang, M.-C., and Yang, L. T. (2014). Data Mining for Internet of Things: A Survey. *IEEE Communications Surveys & Tutorials*, 16(1):77–97.

Wang, T., Ke, H., Zheng, X., Wang, K., Sangaiah, A. K., and Liu, A. (2019). Big Data Cleaning Based on Mobile Edge Computing in Industrial Sensor-Cloud. *IEEE Transactions on Industrial Informatics*, pages 1–1.