A Multi-tier Data Reduction Mechanism for IoT Sensors

Liang Feng
China North Vehicle Research Institute
Beijing, China
finalion@gmail.com

Pranvera Kortoçi
Aalto University
Espoo, Finland
pranvera.kortoci@aalto.fi

Yong Liu
China North Vehicle Research Institute
Beijing, China
yongliu@noveri.com.cn

ABSTRACT

The increasing number and variety of IoT (Internet of Things) devices produce a huge amount of diverse data upon which applications are built. Depending on the specific use case, the sampling rate of IoT sensors may be high, thus leading the devices to fast energy and storage depletion. One option to address these issues is to perform data reduction at the source nodes so as to decrease both energy consumption and used storage. Most of current available solutions perform data reduction only at a single tier of the IoT architecture (e.g., at gateways), or simply operate a-posteriori once the data transmission has already taken place (i.e., at the cloud data center). This paper proposes a multi-tier data reduction mechanism deployed at both gateways and the edge tier. At the gateways, we apply the PIP (Perceptually Important Point) method to represent the features of a time series by using a finite amount of data. We extend such an algorithm by introducing several techniques, namely interval restriction, dynamic caching and weighted sequence selection. At the edge tier, we propose a data fusion method based on an optimal set selection. Such a method employs a simple strategy to fuse the data in the same time domain for a specific location. Finally, we evaluate the performance of the proposed filtering and the fusion technique. The obtained results demonstrate the efficiency of the proposed mechanism in terms of time and accuracy.

Author Keywords

Data filtering; data fusion; Internet of Things; sensors; time series.

INTRODUCTION

IoT (Internet of Things) devices such as sensors, wearables, smart objects, and wireless cameras are being deployed in a wide variety of areas, ranging from building and home automation, to smart cities, automotive and health care [11]. These devices enable the creation of commercial and industrial applications that demand high-speed processing of a substantial amount of data. Indeed, the burden put in the network bandwidth, energy consumption, local storage and data processing is considerable. Such an issue has given rise to different architectural and computing models including edge, fog and cloud [5] to process heterogeneous and bandwidth-intensive input data ranging from simple scalar readings (e.g., temperature and humidity) to multimedia content [8].

In particular, high energy consumption and limited storage space at the source devices call for data reduction before transmission. Such issues become crucial as the amount of data at the IoT devices increases. In fact, up to 80% of the total energy expenditure in an IoT sensor network is due to the wireless data transmission [2]. Therefore, reducing the amount of data prior to transmission is of essential importance. Data reduction at the source not only leads in to energy saving. In fact, transmitting compressed and (or) filtered data also reduces the stress on the network bandwidth, which results in a more efficient usage of the available resources [1].

Simple yet fundamental data aggregation techniques leverage the correlation among the data generated by different sensor nodes. Such a correlation is *a*) spatial: sensor nodes observe the same phenomena in a limited geographical area, causing the sensory data from various sensors to have high similarity *b*) temporal: the readings of the event or phenomenon slowly change over time, therefore adjacent time series of the data present slight variations from each other, and *c*) semantic, where the data generated by different sources belongs to the same connotation or domain. Existing data reduction techniques reduce the amount of data either by leveraging the spatial-temporal redundancy [3, 18], hence selecting only a subset of sensor nodes to be active at once; automatically adjusting the sampling rate of the sensor to the network conditions [15]; or by (equally or randomly) compressing the sensory data [17, 16, 19]. Previous research focused on aggregation, routing and efficient data representation; however, such solutions are limited as they operate at a single level of the network architecture in most cases.

In this paper, we propose a *multi-tier data reduction mechanism that operates at two levels*, namely, at the gateway and at the edge tier of the network. At the gateway tier we apply the PIP (Perceptually Important Point) method [7] to represent the features of a time series by using a finite amount of data. We extend such an algorithm by introducing several techniques, namely interval restriction, dynamic caching and weighted sequence selection. At the edge tier, we propose a data fusion method based on an optimal set selection. Such a

IoT 2017, October 22–25, 2017, Linz, Austria

DOI: https://doi.org/10.1145/3131542.3131557

method deploys a simple strategy to fuse the data in the same time domain for a specific location. Finally, we evaluate the performance of the proposed filtering and fusion techniques by simulations with real-world data sets. The obtained results show the efficiency of the proposed mechanism.

The rest of the paper is organized as follows. We first overview the state-of-the-art of existing data reduction mechanisms. We then describe our proposed filtering technique that extends the original PIP method. We continue by explaining the data fusion method employed at the edge tier. Next, we present an evaluation of the proposed multi-tier solution. Finally, we conclude the paper with a summary, along with remarks on the efficiency and applicability of our data reduction mechanism.

RELATED WORK

Below we review three main categories of methods to conduct data reduction during the data collection and transmission process.

Data Sampling

Data sampling either reduces the amount of sensory data to be further processed, paying special attention to extract the most representative points of the data set, or choosing a subset of sensors that contribute with sensory data among all the IoT sensors. The work in [23] proposes an adaptive sampling approach, where only few sensor nodes are first activated to sample. Such mechanism leverages spatial correlation. The higher the correlation (i.e., the close the IoT sensors to each other), the fewer the number of active sensors that sample data. Such data is then sent to a data fusion center which derives valuable information regarding the environment and the phenomenon under observation, activating more nodes if needed. The data fusion center allocates network resources accordingly. By contrast, [15] adapts the sampling rate of the sensors to the communication resources while minimizing the active sensors that stream data to a remote entity (i.e., server). Moreover, in [16] compression is applied to the already sensed data, whereas [17] proposes a compressed sensing framework, where compressed sensing is applied prior to data acquisition. In [19] instead, compressed sensing is applied to the data on-the-fly, under the assumption of opportunistic routing. Finally, [4] designs an approximate aggregation algorithm that adapts sampling to target precision requirements.

Data Aggregation

Data aggregation consists of considering diverse sources of data and gather them together to build an accurate representation of the phenomena under observation. These sources differentiate in that each of them is responsible of a specific sensing task. In IoT and WSN, data aggregation techniques include operations such as solving the maximum, minimum, sum, average, median, and the count. Aggregation methods can be divided into centralized and distributed methods. Centralized methods require continuous communication among network entities, hence, the communication overhead incurs in additional costs; consequently, such methods are not suitable for sensor networks. Therefore, distributed methods such as clustering, multipath and aggregate trees are commonly employed [10]. The work in [3] performs data aggregation with the purpose of minimizing messaging costs among the sensor nodes. The nodes from which the data is aggregated is proportional to the given size of the network. Similarly, [18] divides the sensor network into non-overlapping regions, then samples and aggregates data from each of such regions. Moreover, the work in [16] summarizes several methods of data compression, which consists of transmitting data as little as possible by aggregating. Such methods are often valid for specific lookup applications, hence not applicable to those that rely on fast- and ever-changing sensory data.

Most data aggregation methods are based on trees or other fixed data structures. In fact, the research in [14] proposes a spanning tree-based data aggregation mechanism in large scale networks. Each leaf in the tree is responsible of sensing data from a given location. Data aggregation starts from the leaves and propagates until the root (i.e., data collection entity). However, other data aggregation methods that rely on no specific data structures are proposed [9].

Multi-task Data Sharing

Multi-task sharing of the network leads to efficient use of the available bandwidth, however it introduces more computation and communication costs. Such a technique aims at collecting as little data as possible satisfying all tasks' needs.

IoT network architectures usually have a sensor-gateway-router-cloud structure [20]. Indeed, we refer to such an architecture to tailor the proposed mechanism for data reduction, aiming at reducing the amount of transmitted data (hence transmission power) at a lower time cost. The authors in [21] propose a technique for fast compression of time series data and indexing of the compressed series. The compression mechanism identifies important points of a series, discarding the remaining ones. This technique has a good performance and can also be applied to resource-concerned applications. In this paper, we improve the solution in [21] by proposing a framework for real-time data filtering over two tiers, namely, the gateway and the edge tier.

THE GATEWAY TIER: DATA FILTERING

Data filtering is based upon data pattern extraction from a huge amount of data, thus limiting the amount of data transferred in the network to the relevant applications. For instance, the work in [7] proposes a flexible algorithm to represent a time series by leveraging data similarities. Time series are made up of infinite number of data points. Patterns on a time series are represented by a subset of points that constitute the entire time series [12]. However, not all the points contain valuable information, hence, a small set of points can shape the basic pattern of the time series. Such points are necessary for pattern recognition and could be extracted as PIPs (perceptually important points). The work in [22] details the mechanism of human recognition of patterns as a process requiring continuous perceptual analysis. When we observe something, we concentrate on increasingly detailed characteristics of the objects. The algorithm of finding PIPs based on the human method of pattern recognition can be deployed to identify key points in time series data. In fact, given a time series data, the corresponding PIPs are constructed as below:

1. The first and the last item of the time series are decided as the first two PIPs.
2. The third PIP is the item with the highest distance to the line that connects the first two points.
3. The item with the highest distance to the line that connects its two adjacent PIPs will be the fourth PIP, and so forth.
4. If the order of the current item in the PIPs sequence is beyond a certain threshold, then it is reserved and forwarded.

Indeed, the original algorithm in [22] focuses on static time series. The series of the real-time data delivery in IoT networks is not fixed as new and continuous sensory data from the sensor nodes flow into the gateway nodes. When the new data comes, it should be decided at once whether to keep or discard it. To address such an issue, [20] proposes a streamification solution by introducing a cache and a cache projection. The cache is projected in to the cache projection with one of the three strategies: *a*) appending a copy of the current item (clone), *b*) appending a duplicate of the entire cache (twin), and *c*) appending an item with an average value (avg).

However, in our experiments with actual temperature and humidity data from *Intel Lab Data*[1], we find that in the filtering process there exist randomness which leads to some key data to be discarded because of ignoring some important data features. To overcome such a drawback, we improve the original algorithm in many aspects by introducing interval restriction, dynamic cache and weighted sequence selection.

Interval Restriction and Dynamic Cache

The importance T of a point falls in the range $[0,1]$ and it reflects the order of a point in the PIPs sequence as described above. Generally, the incoming data in a time series is received in approximate time intervals. When the time interval exceeds a certain threshold, the data is considered as a point of importance $T = 1$. The importance of a point not only relates to the distance between such a point and its adjacent ones, but also strongly relates to the time interval with the previously considered point. In fact, such a strategy benefits from the fact that it can cancel the time-consuming iteration process to calculate the point importance T.

In order to streamify the algorithm to decide the incoming data in real time, a data cache is introduced to save the present data which is already received. Therefore, the new data is stored in to the cache, waiting for the importance to be calculated. Hereby, we define a cache $C: \{c_1, c_2, \cdots, c_W\}$ of fixed-size W. Assuming a filtering ratio r and a time window of size W, only rW data can be sent to the next level of the network while the remaining data is discarded. We express the importance T as:

$$T = \begin{cases} 1 & \Delta t > t_{\max} \\ 1 - \frac{i}{rW} & \text{otherwise} \end{cases} \tag{1}$$

where i is the data index in the ordered cache.

Once the importance of the data reaches a given value, the cache will be cleared out and continues to buffer new incoming data. As a rule of thumb, cache clearing is triggered when the time interval reaches the threshold $t_{\max}$. Such an operation is necessary because the real-time capability would be seriously affected if the cache always keeps increasing. In fact, after a long enough time interval, the sensor nodes would experience a network service pause or halt. Such nodes in the network can be considered to have transitioned into a new state, where the new generated data is classified in a new cluster. We express the cache size C as:

$$C = \begin{cases} 0 & \Delta t = t_{\max} \\ C \cup \{P_k\} & \text{otherwise} \end{cases} \tag{2}$$

where P_k is the $k-th$ data received.

The size of the cache is fixed, but the data in the cache changes over time. Once the cache reaches its predefined maximum size, the cache will first remove the earliest data and add the newest one, then deploy the *twin* projection strategy to update the cache.

Weighted Sequence Selection

The ordering process allows to get the top three points including the two border ones and the one calculated by the two border ones. For instance, if the top 5 data points are ready, the next 4 data point $\{P_i | i = 6,7,8,9\}$ can be obtained by the sequences:

$$S_{1,5} = P_1, P_5 \Rightarrow \{P_k\} \quad S_{2,5} = P_2, P_5 \Rightarrow \{P_k\}$$
$$S_{3,5} = P_3, P_5 \Rightarrow \{P_k\} \quad S_{4,5} = P_4, P_5 \Rightarrow \{P_k\}$$

Therefore, given top n data points, the next $n-1$ ones are obtained:

$$\{S_{1,n}, S_{2,n}, \ldots, S_{k,n} | 1 < k < n\} \tag{3}$$

where $S_{k,n}$ represents the data sequence starting from index k to n, d is a certain distance between the $n-th$ point to the start and end point of this sequence, which can be Euclidean distance (ED), vertical distance (VD, $p_v(x_v, y_v)$) or perpendicular distance (PD, $p_d(x_d, y_d)$) (see Figure 1).

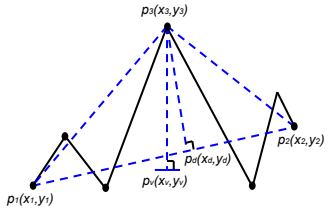


Figure 1: Three distance evaluation methods to find PIPs

However, the calculation order of the sequences is strongly related to the data importance. For instance, different order of calculation of sequences such as $S_{2,5} \to S_{3,5} \to S_{1,5} \to S_{4,5}$ and $S_{3,5} \to S_{1,5} \to S_{4,5} \to S_{2,5}$ can result in indifferent importance value of the data. Hereby, we deploy weighted selecting sequence to normalize the entire process. By adding the weight w_k to each sequence, Eq. 3 transforms into:

$$w_k \{S_{1,n}, S_{2,n}, \ldots, S_{k,n} | 1 < k < n\} \quad \text{s.t. } w_k = \sqrt{T_k^2 + T_n^2} \tag{4}$$

[1] http://db.csail.mit.edu/labdata/labdata.html

where T_k is the importance of the k-th data point. The sequence with a higher weight will be checked much earlier.

THE EDGE TIER: DATA FUSION

The incoming data from the gateways in the first tier is continuously sent to the routers in the edge tier. In fact, even the edge tier requires data reduction before such data is sent to the cloud data center. While the incoming data from the gateways is filtered, not all the sensory data is. Therefore, before transmitting the data from the routers to the cloud data center, we can further filter every dataset contributed by each gateway. Filtering is a lossy operation in terms of information, hence the filtered data in the gateway tier is already information-missing. If such operation is performed once more in the edge tier, such data will lose more valuable information, making it difficult to recover the original data. Hereby, we deploy the data fusion method to fuse all the data from gateways, which improves the data reliability for single data sources, and extends the observation range in the time and space domain.

During the data fusion operation, data modeling, cooperation and explanation are all challenging tasks to perform, hence it is difficult to implement a robust and fast data fusion system. In the IoT context, we simplify the strategy to execute this process by joining the data in the same time domain for a specified space location. Applying the data fusion in the data level removes redundant information and fills up the complementary information.

A key condition is set for the edge sensors to monitor an identical physical phenomenon (e.g., temperature or humidity). Consequently, incoming data from multiple gateways from the same source location can be fused at the data level but not feature or decision level. Assume that at time t, the gateways filter the data coming from n different source gateways $G = G_1, G_2, \ldots, G_n$. The independent observation values for the same physical parameter are given by:

$$G_i(t) = X + v_i(t) \quad i = 1, 2, \ldots, n \tag{5}$$

where $G_i(t)$ is the gateway G_i's filtered data at time t, X is the true value, while $v_i(t)$ is the noise added during the transmission process from gateways to routers. We map all the observations $G_i(t)$ at time t to axes, and the absolute distance between $G_i(t)$ and $G_j(t)$ is $d_{ij}(t) = \|G_i(t) - G_j(t)\|$. If the average value of $G_i(t)$ and all the observations is $d_i(t)$, and the average value among all the observations is $\bar{d}(t)$, then

$$d_i(t) = \sum_{j=1}^{n} d_{ij}(t) \tag{6}$$

$$\bar{d}(t) = \sum_{i=1}^{n} d_i(t) \tag{7}$$

Therefore, the dataset containing all the efficient data points that fall into the proximity of the true value X is the optimal dataset [13], when conditions below are satisfied:

$$d_i(t) < \bar{d}(t) \quad \forall G_i(t) \in \Phi \tag{8}$$

$$d_i(t) \geq \bar{d}(t) \quad \forall G_i(t) \notin \Phi \tag{9}$$

According to the above analysis of the optimal fusion dataset Φ, if the absolute distance between observations $G_i(t)$ and $G_j(t)$ is short, then such values are close to each other, hence the observation data from two gateways can be fused to the corresponding extent. By definition of fuzzy membership, we map $G_i(t)$ and $G_j(t)$ to a function c_{ij} which describes the fusion degree:

$$c_{ij}(t) = \exp\left(-\frac{1}{2}\|G_i(t) - G_j(t)\|\right) \tag{10}$$

From Equation 10 we know that $c_{ij}(t)$ is continuous in the range $[0,1]$. If $c_{ij}(t)$ is close to 1, the fusion degree between $G_i(t)$ and $G_j(t)$ is high, and the vice-versa. Finally, the fusion degree matrix C of the optimal fusion dataset can be written as:

$$C = \begin{bmatrix} 1 & c_{12}(t) & \ldots & c_{1m}(t) \\ c_{21}(t) & 1 & \ldots & c_{2m}(t) \\ \vdots & \vdots & \ddots & \vdots \\ c_{m1}(t) & c_{m2}(t) & \ldots & 1 \end{bmatrix} \tag{11}$$

For each row in the fusion degree matrix C, if the sum of the elements $\sum_{k=1}^{n} c_{ik}(t)$ is high indicates that the observation $G_i(t)$ of the gateway G_i is close to most of the other gateways' observations and the vice-versa. We define a consistent fusion degree $\mu_i(t) = \sum_{j=1}^{m} c_{ij}(t)/m$ to describe the fusion degree G_i with most gateways, and define a distribution balance $\tau_i(t) = 1/\sum_{j=1}^{m}(\mu_i(t) - c_{ij}(t))^2/m$ to describe the stability of the gateway.

Therefore, the fusion weight of G_i at time t is:

$$w_i(t) = \mu_i(t) \times \tau_i(t) \tag{12}$$

We normalize $w_i(t)$ to get the final fusion estimation:

$$\hat{x} = \sum_{i=1}^{m} w_i(t) s_i(t) \tag{13}$$

EVALUATION

The performance evaluation of the data reduction in the gateway tier and the data fusion in the edge tier is conducted as below.

Simulation Setup

We wrote a custom simulator in Python to evaluate our proposed solution. We use datasets from the *Intel Lab Data* which provides time-stamped topology information of the weather such as humidity, temperature, light and voltage values. The

weather information is a typical example of time series data for IoT applications. We conduct the evaluation experiments with temperature and humidity data. We select data from multiple sensors (deployed to sense the same phenomenon) to ensure data integrity and accuracy. We consider 1000 data samples per sensor and set the cache size to 200. For each data set (i.e., temperature and humidity), we simulate the original data (sensor 1) and other two cases where Gaussian noise with SD (standard deviation) values of 1 and 1.2 accordingly are added (sensor 2 and 3). Table 1 summarizes such scenarios.

		Parameters		
Data Source	*Distribution*	*Mean*	*SD*	*Data samples*
Sensor 1	Original	-	-	1000
Sensor 2	Gaussian	0	1	1000
Sensor 3	Gaussian	0	1.2	1000

Table 1: Sensors data simulation

Obtained Results

Filtering Efficiency in the Gateway Tier

Considering the uncertainty of the CPU time, we conduct the same experiment 3 times and take the average value for comparison purposes between Papageorgiou's method [20] and our proposed mechanism, as illustrated in Figure 2. We can see that our proposed method shows fewer time fluctuations, encountering lower time expenses than the Papageorgiou's method. Our method keeps at ≈ 0.018 s for completing a new coming point decision. Moreover, the initial slope is due to the dynamic cache storing the first 200 data samples (i.e, the cache size). Considering our simulation environment, we believe that there is enough room to improve the efficiency on actual embedded IoT devices.

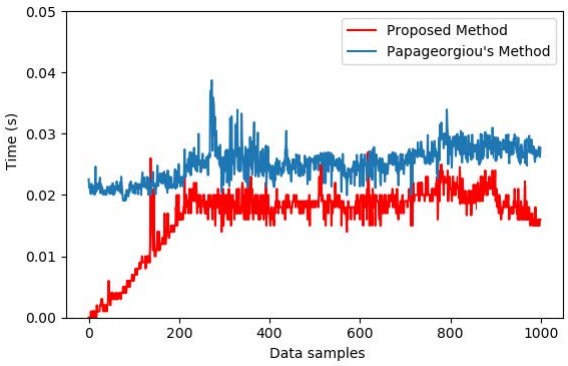
Given two time series TS_1 and TS_2,

$$TS_1 = [(t_1^{(1)}, v_1^{(1)}), (t_2^{(1)}, v_2^{(1)}), \cdots, (t_n^{(1)}, v_n^{(1)})]$$
$$TS_2 = [(t_1^{(2)}, v_1^{(2)}), (t_2^{(2)}, v_2^{(2)}), \cdots, (t_n^{(2)}, v_n^{(2)})]$$

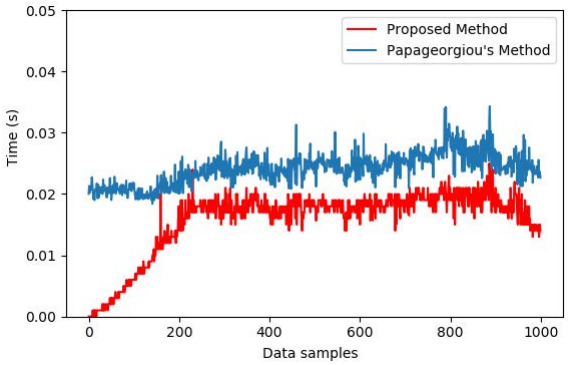
the similarity α between them is defined based on the Jaccard similarity coefficient [6] as,

$$\alpha = \frac{\sum_{i=1}^{n} \min(v_i^{(1)}, v_i^{(2)})}{\sum_{i=1}^{n} \max(v_i^{(1)}, v_i^{(2)})} \tag{14}$$

The importance threshold for the filtering process varies within the range $[0, 1]$ with a step of 0.02 and the similarity between the original series and the filtered series is obtained by Equation 14. As shown in Figure 3, for importance levels (i.e., reduction factor) smaller than 0.7, the similarity between filtered and the original series is above 80% and 90% for the temperature and humidity data sets, accordingly. Such a high similarity allows the filtered series to cover most of the features of the original one. For higher values of importance, the similarity decays almost linearly and only the data with the most significant features is kept.



(a) Temperature data



(b) Humidity data

Figure 2: Performance evaluation in terms of time.

Fusion Accuracy in the Edge Tier

In the edge tier, the proposed fusion algorithm based on the optimal set is tested and compared with fusion methods that rely on average operations. We use the mean squared error (MSE) and recovery accuracy as the evaluation index. MSE describes the data deviation, where a smaller MSE represents a higher accuracy. The results of the experiments are shown in Table 2.

Data	*Source*	*MSE (Proposed)*	*MSE (Average)*
Temperature	Sensor 1	0.598	1.027
	Sensor 2	0.908	1.312
	Sensor 3	0.912	1.409
Humidity	Sensor 1	3.578	3.901
	Sensor 2	2.868	3.215
	Sensor 3	3.026	3.298

Table 2: MSE comparison between proposed method and average method

As shown in Table 2, the data fusion method based on the optimal set has smaller deviation compared to the original data and performs better than the usual average fusion method.

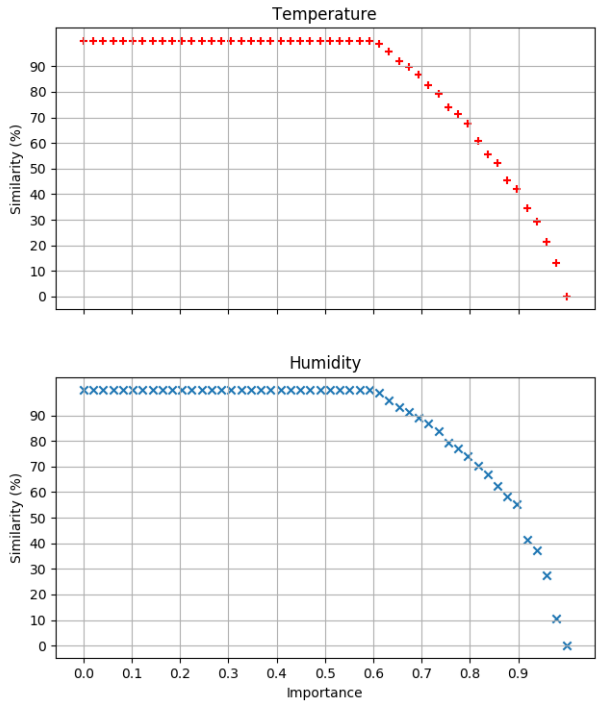


Figure 3: Similarity between the original series and the filtered series.

Then, we can calculate the recovery accuracy of the final fusion results relative to each sensor source. By Equation 14, we can regard the recovery accuracy as the similarity between the fusion data series $TS_o = [(t_{o_1}, v_{o_1}), (t_{o_2}, v_{o_2}), \cdots, (t_{o_n}, v_{o_n})]$ and the original data series $TS_r = [(t_{r_1}, v_{r_1}), (t_{r_2}, v_{r_2}), \cdots, (t_{r_n}, v_{r_n})]$.

We select samples every 20 data samples to compute the recovery accuracy, as shown in Figure 4. The accuracy of the data is higher than 90%, which shows that the fusion algorithm is very efficient in the two-tier network structure. However, we would like to mention that the data from all the sensors may be filtered at the same time. In such a case, the recovered data is taken as the former non-zero value, which gives some flat regions as shown in Figure 5. It indicates that the data have approximate features in these regions, but advanced interpolation algorithms could be considered to improve that.

Figure 5 shows the high fidelity of the data points that the proposed algorithms considers clear representatives of the original ones. In fact, such points closely follow the original data points, concluding that the overall data features are kept. Similarly in the edge tier, the incoming data from the gateways is fused effectively, leading to highly reliable important data representatives to be transmitted to the next level (e.g., cloud data center).

CONCLUSION

In this paper we have devised a multi-tier mechanism for reducing the data generated by IoT sensors. Our proposed solution operates at two levels in the network architecture, namely, the gateways and the edge tier. At the gateways, we leveraged a method based on perceptual characteristics of the data to select the key data points that are representative for the entire time series. We have improved the original algorithm by including additional features, namely, interval restriction, dynamic caching and weighted sequence selection. Our proposed data filtering algorithm can efficiently process streams of data, make real-time data filtering possible, and reduce the cost in terms of time. At the edge tier, we formulated a data fusion method based on optimal set selection to combine the data from all the gateways, which improves the data reliability of single data sources, thus extending the observation range in the time and space domain. This method relies on a simple strategy to fuse the data in the same time domain for a specific location, introducing different weights for each group of data from gateways. Our experiments have demonstrated that the proposed algorithm outperforms the state of the art.

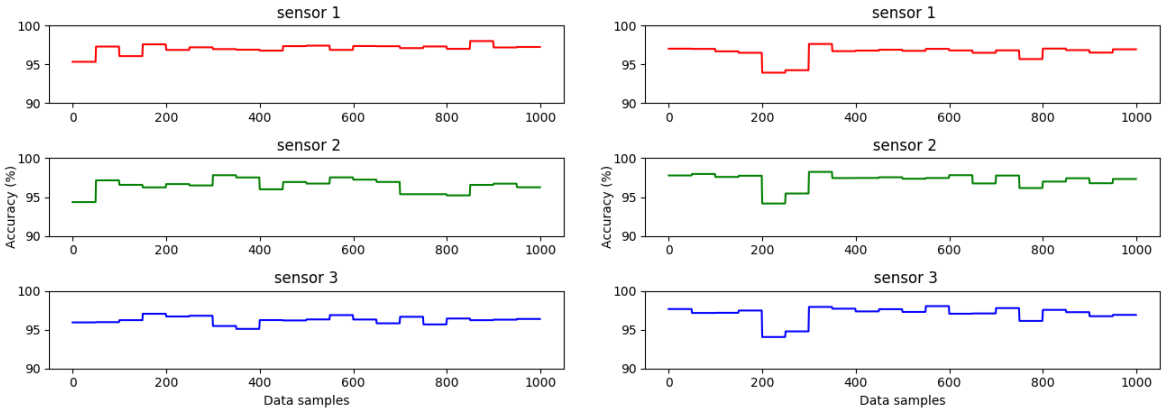
As a future work, we seek to apply information-theoretic approaches such as the information bottleneck theory to analytically characterize data reduction in selected use cases.

ACKNOWLEDGMENTS

This work was carried out while Liang Feng was visiting Aalto University under a scholarship granted by the China Scholarship Council. This work was partially supported by the Academy of Finland under grant number 299222.

REFERENCES

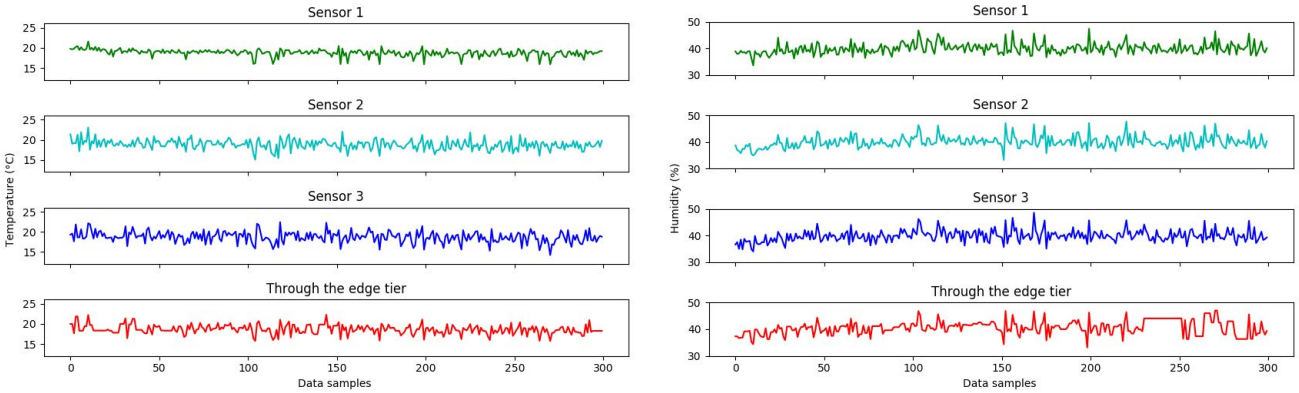
1. Mohammad Abu Alsheikh, Shaowei Lin, Dusit Niyato, and Hwee-Pink Tan. 2016. Rate-Distortion Balanced Data Compression for Wireless Sensor Networks. *IEEE Sensors Journal* 16, 12 (jun 2016), 5072–5083. DOI: http://dx.doi.org/10.1109/JSEN.2016.2550599
2. Mohammad Abu Alsheikh, Shaowei Lin, Hwee-Pink Tan, and Dusit Niyato. 2015. Toward a robust sparse data representation for wireless sensor networks. In *2015 IEEE 40th Conference on Local Computer Networks (LCN)*. IEEE, 117–124. DOI: http://dx.doi.org/10.1109/LCN.2015.7366290
3. Boulat A Bash, John W Byers, and Jeffrey Considine. 2004. Approximately uniform random sampling in sensor networks. In *Proceeedings of the 1st international workshop on Data management for sensor networks: in conjunction with VLDB 2004*. ACM, 32–39.
4. Siyao Cheng, Jianzhong Li, Qianqian Ren, and Lei Yu. 2010. Bernoulli sampling based (ε, δ)-approximate aggregation in large-scale sensor networks. In *Proceedings of the 29th conference on Information communications*. IEEE Press, 1181–1189.
5. M. Chiang, S. Ha, C. L. I, F. Risso, and T. Zhang. 2017. Clarifying Fog Computing and Networking: 10 Questions and Answers. *IEEE Communications Magazine* 55, 4 (April 2017), 18–20. DOI: http://dx.doi.org/10.1109/MCOM.2017.7901470



(a) Recovery accuracy of temperature data

(b) Recovery accuracy of humidity data

Figure 4: Data recovery accuracy relative to each source sensor.



(a) Temperature data

(b) Humidity data

Figure 5: The original data from three sensors and the output samples of such data at the edge tier.

6. Seung-Seok Choi, Sung-Hyuk Cha, and Charles C Tappert. 2010. A survey of binary similarity and distance measures. *Journal of Systemics, Cybernetics and Informatics* 8, 1 (2010), 43–48.
7. Fu-Lai Chung, Tak-Chung Fu, R Luk, and V Ng. 2001. Flexible time series pattern matching based on perceptually important points. (2001).
8. Mario Di Francesco, Mayank Raj, Na Li, and Sajal K. Das. 2012. A Storage Infrastructure for Heterogeneous and Multimedia Data in the Internet of Things. In *The 2012 IEEE International Conference on Internet of Things (iThings 2012)*. 26–33. DOI: **http://dx.doi.org/10.1109/GreenCom.2012.15**
9. Kai-Wei Fan, Sha Liu, and Prasun Sinha. 2007. Structure-free data aggregation in sensor networks. *IEEE Transactions on Mobile Computing* 6, 8 (2007), 929–942.
10. Elena Fasolo, Michele Rossi, Jorg Widmer, and Michele Zorzi. 2007. In-network aggregation techniques for wireless sensor networks: a survey. *IEEE Wireless Communications* 14, 2 (2007), 70–87.
11. Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. 2013. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems* 29, 7 (2013), 1645 – 1660. DOI: **http://dx.doi.org/10.1016/j.future.2013.01.010** Including Special sections: Cyber-enabled Distributed Computing for Ubiquitous Cloud and Network Services & Cloud Computing and Scientific Applications âĂŤ Big Data, Scalable Analytics, and Beyond.
12. Zaib Gul, Ahmed Uzair, and Ali Arshad. 2004. Pattern Recognition through Perceptually Important Points in Financial Time Series. *International Conference on Fuzzy Sets and Soft Computing in Economics and Finance* I (2004), 89–97.
13. YANG Guo-Ning, FENG Xiu-Fang, and FAN Liu-juan. 2012. Multi-Sensor Data Fusion Algorithm Based on

Optimal Fusion Set. *Journal of Software* 23, 1 (2012), 134–140.

14. Zengfeng Huang, Lu Wang, Ke Yi, and Yunhao Liu. 2011. Sampling based algorithms for quantile computation in sensor networks. In *Proceedings of the 2011 international conference on Management of data - SIGMOD '11*. ACM Press, New York, New York, USA, 745. DOI:http://dx.doi.org/10.1145/1989323.1989401
15. Ankur Jain and Edward Y Chang. 2004. Adaptive sampling for sensor networks. In *Proceeedings of the 1st international workshop on Data management for sensor networks: in conjunction with VLDB 2004*. ACM, 10–16.
16. Naoto Kimura and Shahram Latifi. 2005. A survey on data compression in wireless sensor networks. In *International Conference on Information Technology: Coding and Computing (ITCC'05)-Volume II*, Vol. 2. IEEE, 8–13.
17. Shancang Li, Li Da Xu, and Xinheng Wang. 2013. Compressed sensing signal and data acquisition in wireless sensor networks and internet of things. *IEEE Transactions on Industrial Informatics* 9, 4 (2013), 2177–2186.
18. Song Lin, Benjamin Arai, Dimitrios Gunopulos, and Gautam Das. 2008. Region sampling: Continuous adaptive sampling on sensor networks. In *2008 IEEE 24th International Conference on Data Engineering*. IEEE, 794–803.
19. Xiao-Yang Liu, Yanmin Zhu, Linghe Kong, Cong Liu, Yu Gu, Athanasios V Vasilakos, and Min-You Wu. 2015. CDC: Compressive data collection for wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems* 26, 8 (2015), 2188–2197.
20. Apostolos Papageorgiou, Bin Cheng, and Erno Kovacs. 2015. Real-time data reduction at the network edge of Internet-of-Things systems. In *2015 11th International Conference on Network and Service Management (CNSM)*. IEEE, 284–291. DOI: http://dx.doi.org/10.1109/CNSM.2015.7367373
21. Kevin B Pratt and Eugene Fink. 2002. Search for patterns in compressed time series. *International Journal of Image and Graphics* 2, 01 (2002), 89–106.
22. Michael J. Tarr. 2000. Visual Pattern Recognition. *Encyclopedia of psychology* (2000), 1–4.
23. Rebecca Willett, Aline Martin, and Robert Nowak. 2004. Backcasting: adaptive sampling for sensor networks. In *Proceedings of the 3rd international symposium on Information processing in sensor networks*. ACM, 124–133.