Received <day> <Month>, <year>; | Revised <day> <Month>, <year>; | Accepted <day> <Month>, <year>

DOI: xxx/xxxx

RESEARCH ARTICLE

Distributed Energy-efficient Data Reduction Approach based on Prediction and Compression to reduce data transmission in IoT networks

Ahmed Mohammed Hussein[1,2] | Ali Kadhum Idrees[†,2] | Raphael Couturier[3]

[1]Dept. of Information Networks, College of Information Technology, University of Babylon, Babylon, Iraq
[2]Dept. of Computer Science, University of Babylon, Babylon, Iraq
[3]FEMTO-ST Institute/CNRS, Universite Bourgogne Franche-Comte, Belfort, France

Correspondence
[†]Ali Kadhum Idrees, Email: ali.idrees@uobabylon.edu.iq

Abstract

In the modern world, it will be necessary to deploy a large number of sensor devices to sense everything around us in order to detect changes, risks, and hazards and to mitigate them. This increasing number of sensor devices represents an essential data provider in the Internet of Things (IoT). The devices generate and transmit a huge amounts of data which requires a large amount of storage and high processing power to come real-time processing and speed up the network. It also leads to an increase in high energy consumption. Thus, it is important to remove redundant data to reduce the data transmission before sending it to the Gateway while maintaining a good level of data quality. In this paper, a Distributed Energy-efficient Data Reduction (DEDaR) Approach based on Prediction and Compression to minimize the data transmission in IoT Networks is proposed. The DEDaR is used in periods to make decision. In each period, the AutoRegressive Prediction (ARP) is used to predict the data of the next period and make a decision on whether to send the data of the current period to the Gateway or not. In the case of data transmission, the redundant data is eliminated using an efficient compression approach based on adaptive piecewise constant approximation (APCA), Symbolic Aggregate Approximation (SAX), and finally Fixed Code Dictionary (FCD) based on Huffman Encoding. The simulation results based on real-sensed data show that the proposed DEDaR approach outperforms the other recent methods in terms of data reduction percentage, transmitted data size, energy consumption, and data accuracy.

KEYWORDS:
IoT, Sensor Networks, Prediction, Data Reduction, Data Compression, and Network Lifetime.

1 | INTRODUCTION

The fundamental aspect of the Internet of Things (IoT) is to allow the communication of virtual and physical things with each other[1]. IoT systems include embedded intelligence, wireless sensor networks, and cloud computing. Sensors, cameras, Radio Frequency Identifier (RFID), and other devices are used by IoT systems to gather environmental data[2]. These systems are capable of providing sophisticated services such as remote management, online analytics, and real-time remote monitoring. The IoT is used a range of remote monitoring applications, including healthcare, smart manufacturing, smart homes, smart cities, and smart agriculture, with the objective of improving productivity and decreasing costs[3,4]. In the future, Periodic Sensor Networks (PSNs) will be one of the most critical parts of the Internet of Things (IoT), and they will play a key part in people's lives due to their extensive use in a variety of applications.

These types of networks have received alot of attention from researchers in the past four years. The characteristics of these PSNs differentiate it from other ad-hoc wireless networks. Furthermore, several limitations due to these characteristics are imposed and lead to many challenges in the PSNs. Some of these challenges are routing, data aggregation, topology control, security, and coverage. One of the most fundamental research challenges in PSNs is to gather and consolidate huge amounts of data in an energy-efficient manner on a regular basis, then transport them to the sink to extend the network lifetime. Because sensor batteries have a limited lifetime, energy-efficient data collection and data reduction methods for frequent data gathering are required for energy optimization.

Transmitting/receiving the data by sensor devices is an expensive process, while in-network computations are much less expensive from the energy consumption point of view and are sometimes ignored as insignificant[5,6]. However, as shown in Figure 1 [6], the computation requires much less energy than the data transmission/receiving. The energy required to transmit a 1 KB data message over a distance of 100 meters, for instance, is nearly similar to the execution of nearly 3 million instructions on a normal microprocessor. As a result, any extra processing that lowers the data size even by one data bit would save energy. This

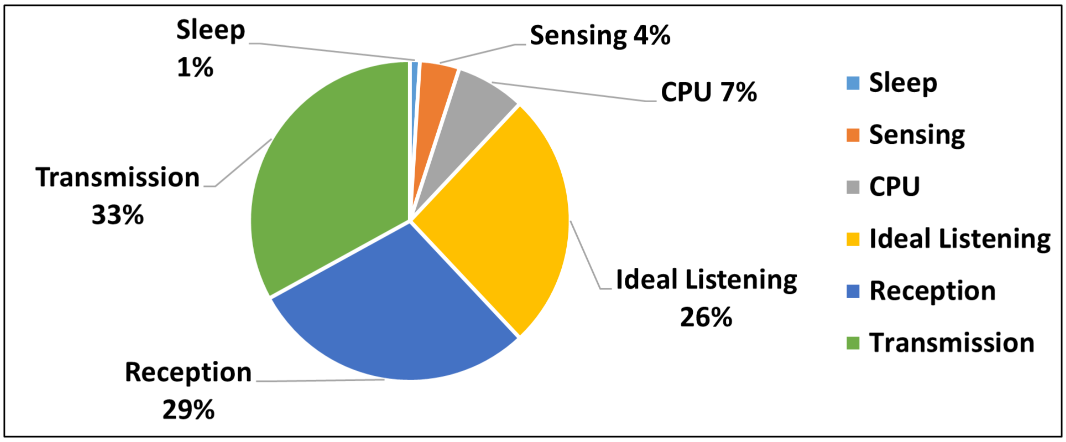


FIGURE 1 Energy consumption inside sensor device.

dramatic difference between transmitting/receiving and computation highlights the importance of in-network data processing in reducing consumed energy in the network[5]. This paper introduces the following contributions:

i i. A Distributed Energy-efficient Data Reduction (DEDaR) Approach based on prediction and compression to reduce data transmission and save energy in IoT networks is proposed.

ii ii. The proposed DEDaR approach executes the Auto Regressive Prediction model to predict the data for the next period and then make a decision about sending or not the current collected data to the Gateway.

iii iii. An efficient compression approach is proposed and implemented by the DEDaR approach to compress the data before sending them to the Gateway. This approach is based on APCA, SAX, and Fixed Code Dictionary based on Huffman Encoding to minimize the size of the data that have been sent to the Gateway and to preserve the power of sensor batteries thus extending the network's life span.

iv iv. A custom simulator based on the Python programming language and based on real observed data from sensor nodes put at the Intel Berkeley Research Lab[7] is used to run several simulation experiments. The suggested DEDaR technique is compared against DPDR[8], DP LSTM[9], DRR-IoT[10], and LMS[11], which are all recent existing methodologies. The DEDaR technique outperforms other methods in terms of data reduction, data accuracy, transferred data size, and consumption of energy, according to the comparison results.

The rest of the paper is laid out as follows: The related works are presented in Section 2, various data reduction methods are also discussed. The theoretical background is introduced in Section 3. In Section 4, the suggested DEDaR approach is presented. Section 5 introduces the simulation results and findings. Section 6 presents the conclusion as well as some recommendations for further research.

2 | RELATED WORKS

WSN-based IoT data reduction has received much interest in recent years. Some of the most often used fundamental data reduction technologies include clustering, scheduling, compressive sensing, multi-channel multi-paths, data compression, data aggregation, and prediction. In this section, certain a number of those alternatives will be presented details explains and one will explain why they work.

In[12], the authors suggest to combine a Gaussian process for robust prediction with a wavelet multiresolution transform. The wavelet domain is handles data and allowing the transform to capture geometric information and break them down it into smaller signals or subbands. For each subband of the wavelet, the deconstructed signal is estimated using a Gaussian process, allowing the processing of Gaussian to pick up a much simpler signal. Using difficult time series generated by several types of sensors,[13] built a multi-dimensional attribute selection model and a prediction model of sensor data reactive. This methodology enhances the accuracy and consistency of IoT sensor data long-term prediction results when compared with existing data prediction models. The prediction model was set to the test with sensor data from Intel Berkeley Research Lab, which have an accuracy of more than 98%, and weather and water data from the Chicago Park District, which have an accuracy of 92%.

As a solution for reducing IoT data, the use of in-networking data filtering and fusion is proposed in[10]. The suggested method is split into two levels, each of which may be adjusted on a single or two tiers. The data change detection and the divergence of true observations from their predicted values are two methods that refer to the initial layer of data filtering layer which was introduced by the proposed method. In the second layer, uses a least square error criteria to merge the data in the same certain region at the same time domain for individual sensors.

To decrease the data size, an adaptive method called AMDR was presented. This method works on two levels: the first level is represented by the sensor node and the second level is represented by the Gateway[14]. This method is used to reconstruct the data based on the threshold determined by the user in advance to maintain the accuracy of the data. In some cases, AMDR gets as good a result as 92% for data reduction while maintaining good forecast accuracy, according to real-world data. AMDR offers realization of all energy saving possibilities.

Some proposed prediction algorithms[15,16,17,18] examine the relationship between gathered data to create a model that compares historical and future values. The authors of[16] recommend modifying the dual prediction scheme (DPS) mechanism. Rather than updating the historical data table, the new version applies a series of models for data prediction during earlier DPS algorithm runs. In reality, the improved model of prediction is generated in (SN) and sent to the Gateway, or vice versa. The authors in[17] propose an unsupervised machine learning system based on self-organized maps to predicting data from sensor nodes. They demonstrate a new predictive method that puts the sensor into hibernation to reduce data transmission based on a self-organizing map. In[18], to track faulty data gathered at the sink, an online data tracking and estimation (ODTE) system is proposed. The Data Prediction System (DPS) and the Distortion Factor are the two basic systems used in ODTE (DF). DPS is used at the sensor level to decrease transmission by setting a limit, whereas DF estimates the best data are collected at the sink node. Compression and aggregation techniques are used in several of the suggested works to minimize data transmission. For periodic sensor applications based on clusters, the authors offer a structure fidelity data collection (SFDC) method in[19]. SFDC searches for spatial correlation between sensors using a distance function and temporal correlation using a similarity index. The authors then present a scheduling strategy to provide power to the sensor in wake/sleep nodes in clusters. Similarly, the authors in[20][14] use a similarity method to search the sensor nodes for spatio-temporal correlation to switch the associated nodes to sleep mode to reduce power in the network. The authors demonstrate that, when compared to alternative similarity metrics, PPMC provides the greatest results in terms of network energy conservation.

The authors in[21,22] proposed a modified k-nearest neighbour algorithm for data redundancy removal in the sensor node to save energy and extend the lifespan of the network. Then, they extend their work to include the data redundancy elimination on the second level of the network (Gateway). The received data vectors of sensor nodes are gathered into groups of similar data vectors and then one representative vector is sent for each group. In[23], the authors introduce a new data reduction method for saving energy in the IoT network. They implemented two algorithms to remove the redundant data at both sensor and aggregator level.

The divide and conquer method is implemented at the aggregator level while the clustering is used with the sensor nodes. The proposed work in[24] presents an energy-aware data transmission reduction in fog computing-based IoT networks. The method is activated at both sensor nodes and the fog Gateway. In the sensor devices, they implement combined grouping and easy encoding methods to remove the unnecessary data before forwarding them to the fog Gateway. In the fog Gateway, they proposed a clustering algorithm based on the "Dynamic Time Warping" (DTW) that combine with the simple encoding method to further remove the redundant data before transmitting them towards the cloud data center.

In[8], the authors suggest two phases of dual prediction to decrease the amount of data which are: (DPP) which refer to the data prediction phase and (DRP) which refer to the data reduction phase which decrease the volume of data sent to the Gateway, and save power in the network, while DPP processed on Gateway in synchronization with previous level (SN) to predicted non-transmitted data in previous level. In[9], producing two variant algorithms: Data Compression (DC) which use to minimize the traffic between Gateway and based station and Gateway and Dual prediction (DP) which utilize to decrease transform of data to the Gateway, they proposed NN network and Long Short-Term Memory networks (LSTMs) for runs the prediction. AM-DR proposed by[11] which depended on merge two decouple LMS of filtering of window that has variant size to find the measurement for two level (SN) and (GW), which make (SN) send just the immediate value that considered irregular value.

Several data reduction algorithms based on clustering techniques have been proposed in the past few years[25,26,27,28,29,30,31,32,33]. The PFF strategy is used in the sensor and aggregator devices[25]. The Jaccard similarity is used by researchers in the sensor node to lower the redundancy of data, and set similarity in the aggregator node to minimise duplicated sets of data. The authors in[26] describe an approach called ATP that was implemented in the sensor device. It lowers the amount of data before transferring them to the base station. They eliminate the redundancy of data at the sensor node, then use a variety of techniques to lessen spatially similar data in the gateway. The authors in[27] suggest a deredundancy algorithm that works at two levels. The first level produces a dual-metric distance, and to obtain clusters of similar nodes, the enhanced k-means method first identifies redundant nodes. At the second step, a Gaussian hybrid clustering classification technique is provided, which is used to build data similarity clusters for edge sensing data. The clustered data is randomly weighted in the third step to deduplicate the spatial correlation data. The EK-means strategy presented by the authors in[28] is a two-step approach. First, it uses a Euclidean distance-based data aggregation approach to reduce similar data collected at the sensor level. Then, it uses an improved k-means clustering algorithm to aggregate similar data sets generated by surrounding nodes into the same clusters, reducing the quantity of data transmitted to the sink even more. In[29], the authors proposed a data reduction method based on aggregation and re-scheduling using clustering techniques to save energy in sensor networks. The cluster formation and cluster head selection are implemented in an energy-efficient way. This model contributes to reducing the transmitted data and increasing the lifetime of the network. The authors in[30] proposed an in-network data lowering approach at the cluster head using an error-aware data clustering scheme. This approach allows the user to select the suitable model that satisfies their requirements and quality of data. The temporal data is clustered into groups, and the data redundancy is removed from each group. The random outliers are detected to introduce an error-aware data clustering that maintains the level of data error under a predefined threshold. For data gathering, a correlation clustering strategy is applied, in which sensor nodes with similar data are grouped as a cluster. Then, using independent component analysis on cluster head IoT devices, an algorithm is developed to gather the data. The data gathering phase is carried out on clusters with higher data correlation[31]. The proposed work in[33] suggests an Integrated Divide and Conquer with Improved K-means strategy for data gathering with power saving in WSNs. It gathers the data at two different levels: node and cluster head. At the sensor node, a divide and conquer technique is being used to eliminate data redundancy from the gathered data before transferring it to the head of the cluster. The head of the cluster uses an improved K-means strategy to group obtained data sets from IoT devices into groups of near-identical data sets and then sends the best representative set from each group to the sink.

The comprehensive resume of related literature mentioned above indicates many methods for data reduction using different techniques. Nevertheless, the proposed methods in existence cannot adequately eliminate redundant data while keeping a high level of data accuracy. Furthermore, some of these methods are more complicated and require high time and memory complexities, and they cannot be implemented inside the constrained resources sensor devices. Consequently, this paper proposed a new method that integrates the prediction and compression methods to highly reduce the data before sending them to the Gateway by eliminating the redundant data while preserving a suitable level of accuracy. A Distributed Energy-efficient Data Reduction (DEDaR) approach is proposed, based on prediction and compression to lower data transmission and preserve energy in IoT networks. First, the lightweight AutoRegressive Prediction algorithm is used to predict the data for the next period and perform a decision related to transmitting the data towards the Gateway. The DEDaR approach compresses these transmitted data before sending them to Gateway using APCA, SAX, and Huffman encoding to decrease the size of sent data to the Gateway and saves the energy of sensor batteries thus enhancing the lifetime of the network.

3 | THEORETICAL BACKGROUND

This study focus on the data reduction problem, in which saving energy is the essential condition. To address this problem, this paper suggests one level data reduction approach for reducing the data redundancy and reducing the power consumed while keeping a good data accuracy in the Gateway. In this section, some theoretical background about some employed technique will be presented.

3.1 | AutoRegressive Prediction (ARP)

The AutoRegressive Prediction (ARP) prediction technique is main aim is to predict the data in the next period in sensor nodes. Autoregressive statistical models predict following values by the history of previous values. An autoregressive model may attempt to stock market prices based on its previous performance for example[34]:

- Autoregressive models estimate future values based on historical values and are commonly employed in technical analysis to forecast future security prices.
- The underlying assumption in autoregressive models is that the future will be similar to the past. As a result, they may be wrong under specific market scenarios, such as financial crises or rapid technological growth.

Because they work on the principle that prior values have an influence on present values, autoregressive models are helpful for analyzing nature, economics, and other time-varying systems. Autoregressive models use an accumulation of the variable's prior values, whereas multiple regression procedures use a linear combination of predictors to predict a variable[34].

The AR(1) autoregressive process's current value is determined by the value immediately preceding it, whereas an AR(2) process's current value is determined by the previous two values. For electronic noise, an AR(0) process with no term dependency is used. There are many various techniques to determine the coefficients suitable calculations used, such as the least squares method, in addition to these variations. Technical analysts use these concepts and methodologies to forecast security prices. However, because autoregressive models rely solely on historical data to predict future values, they implicitly presume that the fundamental forces that drove past prices will remain constant over time. If the underlying dynamics at stake are fluctuate, for example if a particular industry is undergoes rapid and unprecedented technological development, this might lead to unexpected and incorrect forecasts. To create forecasts, an autoregressive model uses a linear combination of the target's past values. Naturally, the regression is performed against the target. An AR(p) model is mathematically stated as[35]:

$$x_t = c + \sum_{i=1}^{p} \oslash_i x_{t-i} + \epsilon_t \tag{1}$$

Where: p: is an order, c: is a constant, and epsilon: noise The AR(p) model is extremely adaptable, and it may be used to describe a wide range of time series patterns. Autoregressive models are often used only on stationary time series.

3.2 | Adaptive Piecewise Constant Approximation (APCA)

The APCA segments of time series can be obtaining from separate time series into sets of varying value segments (with a limited reconstruction error) by using APCA with different lengths depending on the data, with the purpose of reducing individual reconstruction mistakes. In more technical terms, $|R(S^{AP}) - S| < \epsilon$, $R(S^{AP})$ is the reinitialized function, and ϵ is an error threshold. One of the basic components of data analysis is data sorting. It enhances the search and combines the sequences more effectively. As a result, sorting the detected humidity values in descending order improves the efficiency of APCA by grouping the same (or nearly identical) signals together. Long segments represent low-activity data regions, whereas short segments represent high-activity data regions[36]. The (S^{AP}) is given as follows:

$$S^{AP} = \{(dm_1, dr_1), \ldots, (dm_m, dr_m)\}, \quad dr_0 = 0 \tag{2}$$

S_j^{AP} is given by a pair of numbers (dm_j, dr_j), where a mean value of humidity measurements is dm_j in the j^{th} segment, which is given[37]:

$$dm_j = \frac{\sum_{k=dr_{j-1}+1}^{dr_j} S^k}{dr_j - dr_{j-1}} \tag{3}$$

Where right endpoint of the j^{th} segment is dr_j.

3.3 | Symbolic Aggregate ApproXimation (SAX) representation

SAX refers to a method of representation of data of time series. By using symbols set to refer to a time series of length p into q, where the length of a time series is q_p. To begin, the time series is normalized, converting the standard deviation to unity and the mean to zero. This process ensures that input data are transformed to output series with a mean approximating 0, and a standard deviation approximating 1. This normalization is required so that the data reduction algorithm can focus on structural similarities and differences rather than amplitude. The following formula is used to calculate this normalization:

$$\mu = \frac{\sum_{i=1}^{p} S_i}{p} \tag{4}$$

$$\sigma = \sqrt{\frac{\sum_{i=1}^{p} S_i - \mu}{p-1}} \tag{5}$$

$$S_i' = \frac{S_i - \mu}{\sigma} \tag{6}$$

Where σ, μ, and S_i' are the standard deviation, mean, and normalized data readings.

The SAX is regarded as a pioneer in lowering time series dimensionality and numerosity. It is divided into two parts: the Adaptive Piecewise Constant Approximation (APCA) transformation and the numerical data transformation into a set of symbols. A finite alphabet determines the value of each symbol[38]. The following steps are used to convert the APCA considering S^P is the subset of the original time series S into the SAX (S^X) :

1. Divide the series of reading values into w parts.
2. For each component of the readings, calculate mean.
3. From N-letter in alphabet, draw mean value quantized into symbols.

The APCA representation is used in the first two steps. In step 3, employment of quantization (N-1) breakpoints to divide the Gaussian distribution's domain into proportionate regions equal α. A list of sorted numbers $\beta_1, \ldots, \beta_{\alpha-1}$ known as Breakpoints. The region from β_i to $\beta_{i+1} = 1/\alpha$ under a N (0, 1) Gaussian curve, where β_α and β_0 refer to ∞ and $-\infty$ respectively. By searching a table of statistical of them can find Breakpoints. For example, by searching in the table of breakpoints, it is possible to find α values ranging from 3 to 10[39], as shown in Table 1 .

TABLE 1 Breakpoint's values.

α	3	4	5	6	7	8	9	10
β_1	0.43	-0.67	-0.84	-0.97	-1.07	-1.15	-1.22	-1.28
β_2	0.43	0	-0.25	-0.43	-0.57	-0.67	-0.76	-0.84
β_3		0.67	0.25	0	-0.18	-0.32	-0.43	-0.52
β_4			0.84	0.43	0.18	0	-0.14	-0.25
β_5				0.97	0.57	0.32	0.14	0
β_6					1.07	0.67	0.43	0.25
β_7						1.15	0.76	0.52
β_8							1.22	0.84
β_9								1.28

The APCA coefficients can be quantified as follows once the breakpoints have been defined. Every APCA value is converted to "a" if it is larger than or equal to the smallest breakpoint. The other values are less than the second smallest breakpoint are changed to "b," and so on.

3.4 | Huffman Encoding

Huffman Encoding is a binary tree-based technique that leverages the frequency (or probability) property of symbols. It entails the following three steps[40]: probability calculation & symbol ordering, transformation of a binary tree, and assigning symbols with codes. In probability calculation & symbol ordering, the number of each symbol in the entire data is counted, then the count is divided by the total number of letters in the data to get the "probability" of each symbol. One of the advantages of Huffman Encoding relies on the fact that it is a probability-based method. The most common symbols — the once with the highest probabilities — are often represented with fewer bits than least common symbols[40]. For example, we have the probabilities as has been shown in Figure 2 for the following data with five different symbols: A B C D E:

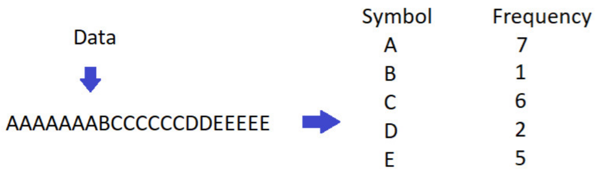


FIGURE 2 : Example of probabilities of five symbols A, B, C, D, E

Then, by modeling each symbol as a node and calling it our "collection," one can easily organize the symbols according to their probabilities. Now ready to go on to the next step (see Figure 3).

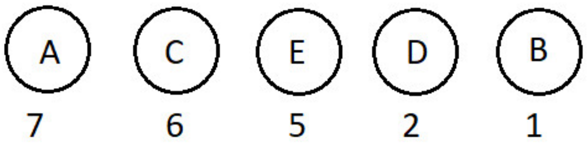


FIGURE 3 : The organized symbols according to their probability

The binary tree transformation can be achieved as follows (see Figure 4):

1. Select the two nodes with the shortest sum of probabilities from the collection. Then, merge the two nodes with a root includes the probability equal to the sum of probabilities of these two selected nodes.
2. Reintroduce the new tree to the collection.
3. Repeat this process until the tree that encompasses all of the input probabilities.

To assign the codes to symbols, the only thing that need to be done is to assign 1 for each time go right child and 0 for each time go left child (see Figure 5). Then, the binary tree is obtained that known as Huffman Tree[40]. Finally, using Huffman Encoding, the symbols and their codes can be retrieved.

Even if the difference between compressed and non-compressed data is only 21 characters, which this difference can be seen as significant. The number of bits calculation before and after compression can be shown in Figure 6 .

4 | PROPOSED DEDAR APPROACH

A Distributed Energy-efficient Data Reduction (DEDaR) Approach based on Prediction and Compression is proposed to decrease data transmission in IoT networks. The DEDaR approach is implemented on each sensor device. It is regarded as an effective

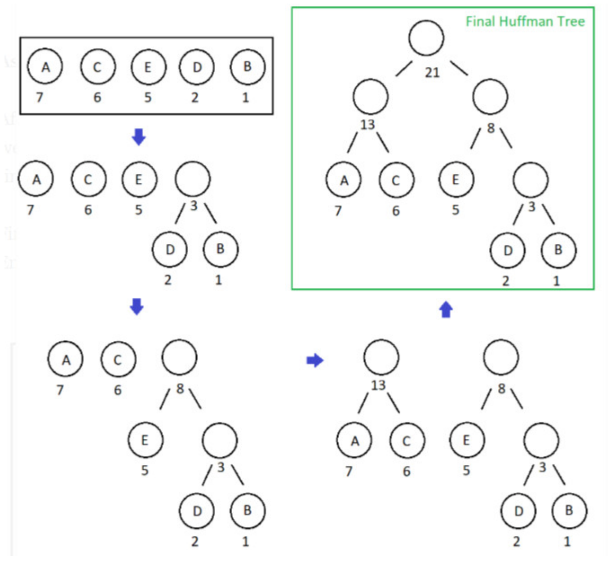


FIGURE 4 : Binary Tree Transformation

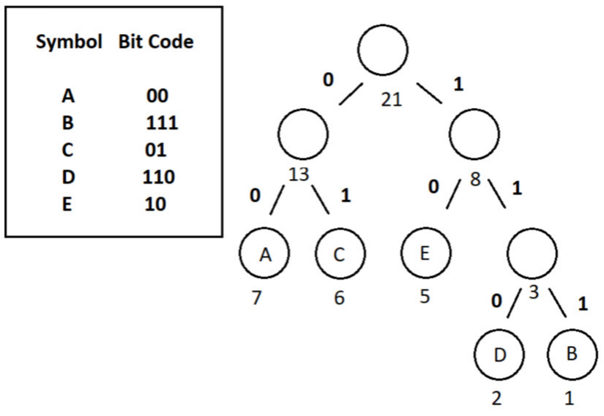


FIGURE 5 : Assigning Codes to Symbols

method to save power and reduce the amount of transferred data, and thus extends the lifetime of the network while maintaining the accuracy of received data at the Gateway. Figure 7 depicts the flowchart of the proposed DEDaR approach.

4.1 | Collection of IoT Sensor Data

The data collection is periodic, where the new reading is collected from sensor node i at slot time s. The node i then creates a new data vector (i.e. time series vector) from the captured readings $R = [r1, r2, \dots, yT]$ at each period, where T is the total amount of readings in each period. The IoT sensor node will capture the (same or very similar) measurements, especially when s is very short or when there is no change in the monitored area for a long time. This will allow to the IoT sensor node to send large volume of data to the Gateway at each period.

4.2 | APCA Technique

At the beginning, the first period will be sent to the Gateway. The data of this period and the other periods that need to be sent to the Gateway will be processed to remove the redundancy at the sensor node's level before sending them to the Gateway.

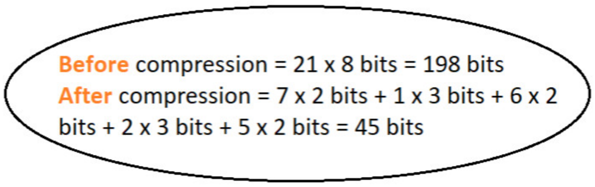

Before compression = 21 x 8 bits = 198 bits
After compression = 7 x 2 bits + 1 x 3 bits + 6 x 2 bits + 2 x 3 bits + 5 x 2 bits = 45 bits

FIGURE 6 : Number of bits calculation before and after compression

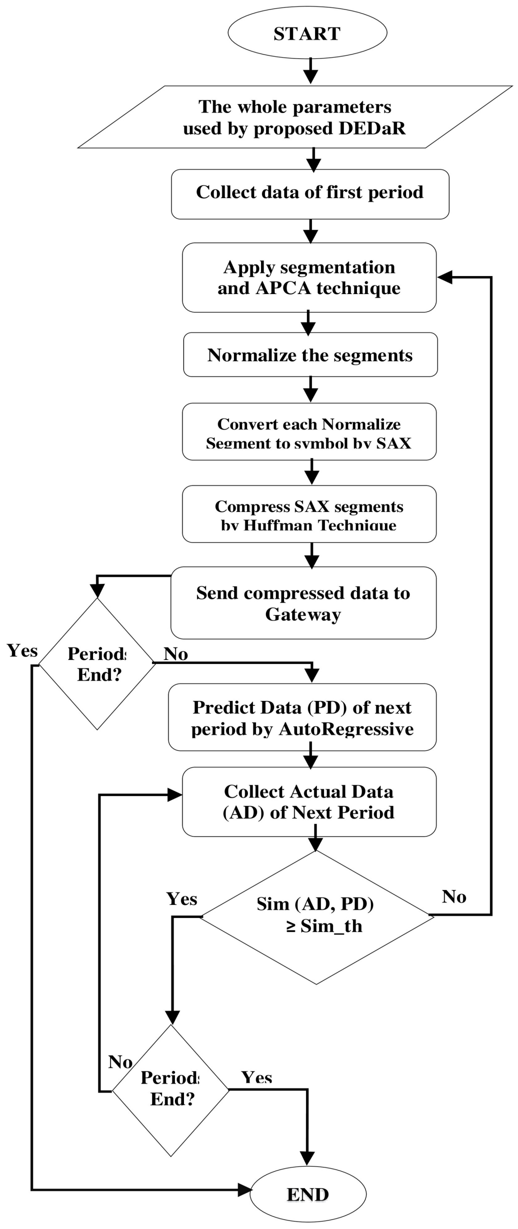

START
The whole parameters used by proposed DEDaR
Collect data of first period
Apply segmentation and APCA technique
Normalize the segments
Convert each Normalize Segment to symbol by SAX
Compress SAX segments by Huffman Technique
Send compressed data to Gateway
Yes
Period End?
No
Predict Data (PD) of next period by AutoRegressive
Collect Actual Data (AD) of Next Period
Yes
Sim (AD, PD) ≥ Sim_th
No
No
Period End?
Yes
END

FIGURE 7 : Flowchart of proposed DEDaR approach

Therefore, the dimensionality reduction based on segmentation and APCA are to segment the data into m segments of data readings with different lengths depending on their activities, where the long segments represent the low activities and the short segments represent the high activities. The algorithm 1 shows the Dimensionality Reduction based Segmentation and APCA.

Algorithm 1 :Dimensionality Reduction based Segmentation and APCA

```
Require: SoD (set of data reading of a period)
Ensure: $SoG = \{SoG_1, \dots, SoG_m\}$(set of m segments of readings), SegLen (set of lengths of segments)
 1: $l \longleftarrow 0$
 2: $\overrightarrow{set\_bfr} \longleftarrow \phi$
 3: $m \longleftarrow 0$
 4: $SoD \longleftarrow Sorting(SoD)$                                   ▷ Sorting of readings
 5: for $j \longleftarrow 1$ to $len(SoD)$ do
 6:     $set\_bfr_l \longleftarrow SoD_j$
 7:     $l \longleftarrow l + 1$
 8:     $Sm \longleftarrow 0$
 9:     $Count \longleftarrow 0$
10:     $flag \longleftarrow 0$
11:     for $i \longleftarrow 1$ to $len(\overrightarrow{set\_bfr})$ do
12:         $Sm \longleftarrow Sm + set\_bfr_i$
13:         $Count \longleftarrow Count + 1$
14:     end for
15:     $SEG\mu \longleftarrow \frac{Sm}{Count}$
16:     $i \longleftarrow 0$
17:     while $(flag = 0)$ and $(i < len(\overrightarrow{set\_bfr}))$ do
18:         if $(set\_bfr_i - SEG_\mu) \geq e\_max$ then
19:             $flag \longleftarrow 1$
20:             $i \longleftarrow i - 1$
21:         else
22:             $i \longleftarrow i + 1$
23:         end if
24:     end while
25:     if $flag = 1$ then
26:         $l \longleftarrow 0$
27:         $m \longleftarrow m + 1$
28:         $Count \longleftarrow 0$
29:         $Sm \longleftarrow 0$
30:         for $j \longleftarrow 1$ to $i$ do
31:             $Sm \longleftarrow Sm + set\_bfr_j$
32:             $Count \longleftarrow Count + 1$
33:         end for
34:         $SoG\mu \longleftarrow \frac{Sm}{Count}$
35:         $SegLen_\mu \longleftarrow len(\overrightarrow{set\_bfr})$
36:         $\overrightarrow{set\_bfr} \longleftarrow \phi$
37:     end if
38: end for
39: return $SoG, SegLen$
```

4.3 | SAX Representation

Many different ways to represent time series using symbolic representations have been introduced in recent decades. This approach converts from a numerical form of time series to a sequential series of discrete symbols using pre-defined mapping rules. Symbolic techniques are more noise resistant. SAX is the most extensively used symbolic representation approach for time series data mining. A large reduction in dimensions ensures this representation method as well as a reduced bounding property, improving the efficiency of algorithm. It is necessary to transform time series, which contain data readings from sensor nodes, into suitable forms for further analysis. To deal with time series, it is recommend to use two different representation techniques: a normalization and a symbolic representation.

After applying the APCA method, which divides the data into a set of segments, the normalization method was used on the data of each segment as shown in Algorithm 2. Then, the SAX method was used to convert the data into symbols by using a breakpoints value whose alphabet range is 3–10 symbols as shown in Table 1 . Note that the number of characters used determines the breakpoints values. In our method, the number of symbols that are used is 10. Every APCA value will be converted to the suitable representation using the SAX algorithm. Algorithm 3 demonstrates the SAX algorithm.

A symbolic representation of data readings for IoT sensor nodes from a normalized one by using the SAX approach. The SAX can be generated the quantification does this conversion by dividing the area under the Gaussian distribution into an equal proportional area using (N - 1) breakpoints. The breakpoints can be found by searching for them in a statistical table, as illustrated in Table 1.

Algorithm 2 :Normalization Data of Segments

```
Require: $SoG^x$: APCA Segment, $SegLen^x$: Length of APCA Segment $SoG^x$.
Ensure: $SoN^x$: Set of Normalize Segment.
 1: $Sum_\mu \leftarrow 0$
 2: for $j \leftarrow 1$ to $SegLen^x$ do
 3:     $Sum_\mu \leftarrow Sum_\mu + SoG^x_j$
 4: end for
 5: $Count \leftarrow Count + 1$
 6: $\mu \leftarrow \frac{Sum_\mu}{SegLen^x}$                          ▷ find mean values
 7: $Sum_\sigma \leftarrow 0$
 8: for $j \leftarrow 1$ to $SegLen^x$ do
 9:     $Sum_\mu \leftarrow Sum_\mu + (SoG^x_j - \mu)^2$
10: end for
11: $\sigma \leftarrow \sqrt{\frac{Sum_\mu}{SegLen^x}}$                 ▷ find standard deviation
12: for $j \leftarrow 1$ to $SegLen^x$ do
13:     $SoN^x_j \leftarrow \frac{SoG^x_j - \mu}{\sigma}$
14: end for
15: return $SoN^x$
```

4.4 | Huffman Encoding

The proposed DEDaR approach employs a static alphabet dictionary derived from the implementation of the Huffman algorithm on the data of the sensor 1 in the dataset and we named these fixed alphabet as Fixed Code Dictionary (FCD) that based on Huffman encoding. The FCD is constructed as a fixed dictionary to avoid frequent sending for dictionary of variable codes with transmitted encoded data to the Gateway to decode received data. The fixed alphabet must achieve two conditions: the absence of ambiguity in the code, and the code must be as minimal as possible.

The FCD will be static for all periods with a variable bit of code for each symbol. The Gateway will receive this FCD dictionary only once. Table 2 shows the Fixed Code Dictionary (FCD). In this table, the first column represents the symbols were generated by the SAX representation (Algorithm 3). The second column refers to the code of the symbols. The third column represents the number of required bits of each code to represent the symbols.

Algorithm 3 : SAX Representation

```
Require: $SoN^x$ : Set of Normalize values of Segment x, $SegLen^x$: length of Segment x, $\beta$ (breakpoints values of Table 1 )
Ensure: $SAX^x_{sm}$: Symbols of Segment x.
 1: $alphbet \longleftarrow \{'a','b','c','d','e','f','g','h','i','j'\}$
 2: $char \longleftarrow 10$                                   ▷ The no. of symbols that represent the values
 3: for $i \longleftarrow 0$ to $SegLen^x$ do
 4:     for $j \longleftarrow 0$ to $len(\beta)$ do
 5:         if $\beta_{[j+1][char-1]} > SoN^x_i \geq \beta_{[j][char-1]}$ then
 6:             $SAX^x sm_i \longleftarrow alphabet_j$
 7:         end if
 8:     end for
 9: end for
10: return $SAX^x_{sm}$
```

TABLE 2 Fixed Code Dictionary (FCD).

Symbol	Code	No. of Bit
a	00	2
b	0100	4
c	0101	4
d	0110	4
e	0111	4
f	1000	4
g	1001	4
h	10100	5
i	10101	5
j	10110	5

4.5 | AutoRegressive Prediction Model

Autoregression predicts the value at future time steps by using data from prior time steps as input to a regression equation. It is a simple method that can produce reliable forecasts for a variety of time series issues. The DEDaR approach employs a model for time series forecasting based on an Autoregressive prediction method. A linear regression model forecasts the result using a linear combination of input values and can be formulated as:

$$y_{hat} = b_0 + b_1 * X_1 \tag{7}$$

Where y_{hat} is the forecast, b_0 and b_1 are parameters discovered by model optimization on training data, and X is an input value. This approach can be applied to time series when the input parameters, referred to as lag parameters, are observations from prior time steps.

For instance, he value of the next time step (t+1) can be estimated based on the data from the previous two-time steps t-1 and t-2. This might be represented as a predictive model as follows:

$$X_{t+1} = b_0 + b_1 * X_{t-1} + b_2 * Xt - 2 \tag{8}$$

The regression model is known as an Autoregression because it uses the same input variable as data at earlier time steps. An Autoregression model is a linear regression model using lagged data as input variables. Manually design the linear regression model and the lag input parameters could use the Linear Regression. Alternatively, the statistical model package offers an Autoregression model that requires a lag value and trains a linear regression model. It is part of the AutoReg class. This model can be used by first generating the AutoReg() model and then training it on our dataset with fit(). An AutoRegResults object is returned. Once the model is fit, it can used it to generate predictions by using the predict() method for a set of future data.

The DEDaR approach collects the data inside the sensor node periodically. The collected data of the first period are used by the autoregressive prediction technique to predict the data of the next period. The sensed data of the first period will always be transmitted to the Gateway after compressing them using steps B, C, and D. The similarity algorithm (see Algorithm 5) will be applied between the predicted data and new collected data of the next period. If the results are greater than or equal to a predefined threshold Sim_th, they are considered as similar and they will not be transmitted to the Gateway. Otherwise, if the result of similarity function less than Sim_th, the sensed data are considered dissimilar and should be transmitted to the Gateway. The techniques in B, C, and D are applied on the data before sending them to the Gateway. Algorithm 4 shows the AutoRegressive Prediction algorithm is used by the proposed DEDaR approach. Every time the similarity between the actual collected data and the predicted data is less than Sim_th, the prediction model will be trained with actual collected data to produce the new predicted data set that will be used for the following periods.

Algorithm 4 AutoRegressive Prediction

```
Require: DF: Set of data readings of the period from the sensor
Ensure: PD: Set of predicted data
1: $count \leftarrow 0$
2: for $i \leftarrow 0$ to $\frac{len(DF)}{2}$ do
3:     $model \leftarrow AutoReg(DF, lags = i)$        ▷ call AutoReg function to create model with $lag_i$
4:     $model\_fit \leftarrow model.fit()$        ▷ Make fitting on model by call fit function
5:     $PD[count] \leftarrow model\_fit.predict(len(DF), len(DF)))$        ▷ make prediction
6:     $count \leftarrow count + 1$
7: end for
8: return $PD$
```

The AutoReg() method in Algorithm 4 gets data from the sensor as well as the number of lags to include in the model (if an integer) or a list of lag indices to be included. For example, [1, 4], includes only lags 1 and 4, whereas lags = 4 includes lags 1, 2, 3, and 4. None of the AR lags respond in the same way as 0. Then, using the fit() function, fit the model. Model fitting is a statistic for determining how well a machine learning model generalizes to data that is comparable to the data it was trained on. A model that is well-fitted produces more accurate results. After training the model, the following step is usually to make predictions on the testing set. To do so, we must use the predict() function, which will effectively use the learnt parameters from fit() to make predictions on fresh, unknown test data points. Predict(), in essence, will make a forecast for each test instance, and it normally only requires a single parameter (X). The predicted value for classifiers and regressors will be in the same space as the one observed in the training set.

Algorithm 5 illustrates the Similarity between Actual data readings sets and Prediction Data readings sets using Euclidean distance measurement.

Algorithm 5 : Similarity Algorithm (PD, AD)

```
Require: PD: predicted data set of the period, AD: actual data set of the period
Ensure: Sim: similarity value
1: $sum \leftarrow 0$
2: for $\imath \leftarrow 0$ to $len(PD)$ do
3:     $sum \leftarrow sum + (PD_i - AD_i)^2$
4: end for
5: $Sim \leftarrow \sqrt{sum}$
6: return $Sim$
```

4.6 | Illustrate Example of proposed system

This section illustrates an illustrative example for applying the steps in B, C, and D on the data set of size 20 data readings (19.98, 19.98, 19.98, 19.3, 19.3, 19.17, 19.14, 19.16, 19.11, 19.06, 19.03, 19.02, 19.01, 19, 18.98, 18.97, 18.96, 18.94, 18.92, 18.91). Table 3 presents an illustrative example.

TABLE 3 Illustrative example.

Data Readings	APCA Segment	Normalization	SAX Representation	Huffman Encoding
19.98 19.98 19.98	19.98	4.719	j	10110
19.3 19.3	19.3	1.129	i	10101
19.17 19.14 19.16	19.16	0.41	g	1001
19.11 19.06	19.09	0.154	f	1000
19.03 19.02 19.01 19 18.98	19.008	-0.563	c	0101
18.97 18.96 18.94 18.92	18.94	-0.871	b	0100
18.91	18.91	-0.871	b	0100

Algorithm 1 is applied on the data set of readings. After the segmentation, seven segments are created ([19.98, 19.98, 19.98], [19.3, 19.3], [19.17, 19.14, 19.16], [19.11, 19.06], [19.03, 19.02, 19.01, 19, 18.98], [18.97, 18.96, 18.94, 18.92], [18.91]). Then these segments are processed by the APCA method to produce a set of seven elements (19.98, 19.3, 19.16, 19.09, 19.008, 18.94, 18.91). These elements in the produced set are normalized using Algorithm 2 to produce the following set (4.719, 1.129, 0.41, 0.154, -0.563, -0.871, -0.871). The SAX representation is applied on the produced normalized set using Algorithm 4 to give a set of symbols (j, i, g, f, c, b, b) depending on breakpoints table and alphabet table (each index value in breakpoints table refer to index symbol in alphabet table). Finally, these set of symbols are encoded based on Fixed Code Dictionary derived from the Huffman encoding method that presented in Table 2 where the resulted codes are introduced in the following list (10110, 10101, 1001, 1000, 0101, 0100, 0100).

4.7 | Computational complexity

This section provides the analytical study of the computational complexity for the proposed DEDAR approach. Each sensor node i collects a series of data during each period, which is represented by R, where the total amount of readings during one period is T. The time complexity of the Algorithm 1, which is responsible for dimensionality reduction based on segmentation and APCA is O($MAX(Len(SoD)Log_2 Len(SoD), Len(SoD) * Len(\overrightarrow{set\_bfr}))$). The space requirement for Algorithm 1 is $\Theta(Len(SoG) * m + Len(SegLen) + Len(SoD) + Len(\overrightarrow{set\_bfr}))$. The time requirement for Algorithm 2 is $\Theta(SegLen^x)$, while it requires storage equal to $\Theta(Len(SoG^x) + Len(SoN^x))$. Algorithm 3 requires time and storage equal to $\Theta(SegLen^x * Len(B))$ and $\Theta(Len(SoN^x) + Len(\beta) + SegLen^x)$ respectively. The time and space complexities are required by Algorithm 4 are

$\Theta(Len(DF) * lags)$ and $\Theta(Len(DF) + Len(PD))$ respectively. Algorithm 5 requires $\Theta(Len(PD))$ of time complexity, whilst it requires storage equal to $\Theta(Len(PD) + Len(AD))$. The Huffman Encoding uses a fixed code dictionary (FCD) that includes the used symbols and their codes. The FCD will be saved at both sender (sensor node) and receiver (gateway). It is simple and requires a very small size of memory (121 bits). The encoding requires $\Theta(Len(SAX^{x}_{sm}))$ of time complexity. Hence, the time complexity for DEDaR approach is is $O(MAX(Len(SoD)Log_2 Len(SoD), Len(SoD) * Len(\overrightarrow{set\_bfr})))$ and the storage requirement is $\Theta(Len(SoG) * m + Len(SegLen) + Len(SoD) + Len(\overrightarrow{set\_bfr}))$. This computational complexity analysis shows the implementation cost of the proposed DEDaR approach from the time and storage requirements point of view.

5 | PERFORMANCE EVALUATION

The simulation results, analysis, and discussion in this part are used to assess and demonstrate the performances of the proposed DEDaR technique. The simulation experiments are carried out with the help of a custom simulator written in Python. In these simulation experiments, actual data from sensor nodes deployed at the Intel Berkeley Research Lab are used[7]. As illustrated in Figure 8 , the Berkeley database contains data on humidity, temperature, voltage, and light acquired every 31 seconds from 54 sensors. We will employ 10000 sensed data measures from three sensors in this work, and the focus will be met on one type of sensor measurements: humidity, for the purpose of simplicity (Sensor 1, Sensor 2, and Sensor 3). The sink (Gateway) is in the middle of the building.

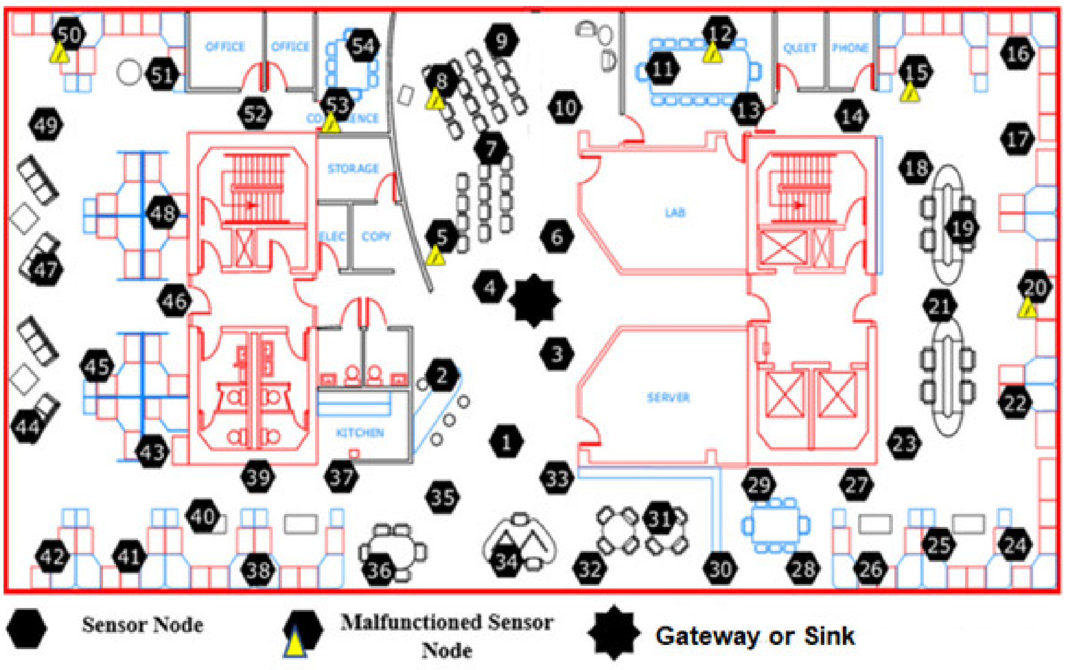


FIGURE 8 Intel Berkeley Research Lab

The energy consumption model called the "First Order Radio Model," proposed by Heinzelman[41]. was used. Figure 9 presents the First Order Radio Model.

The captured reading is of length 64 bits; therefore, the size of the packet is the number of readings multiplied by 64 bits. Table 4 presents the simulation parameters.

In order to show the high efficiency of the proposed DEDaR approach, it is compares with some related recent existing methods such as DPDR[8], DP LSTM[9], DRR-IoT[10], and LMS[11]. The comparison is achieved based on some performance metrics such as the percentage of data reduction, data accuracy, transmitted data size, energy consumption, and network lifetime.

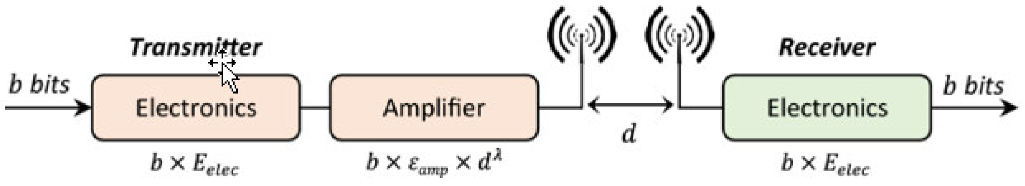


FIGURE 9 First Order Radio Model

TABLE 4 The Simulation Parameters Values.

Parameter	Value
T	10000
e_max	0.03, 0.05, 0.07, and 0.09
Eelec	$50 * 10^{-9}$
ϵamp	$100 * 10^{-12}$
Λ	2
Sim_th	70%

The threshold Sim_th is selected after achieving several experiments by the proposed DEDaR approach and by using different Sim_th ratios. Figures 10 , 11 , and 12 show the transmitted data, energy consumption, and accuracy for sensor1, sensor2, and sensor3 respectively.

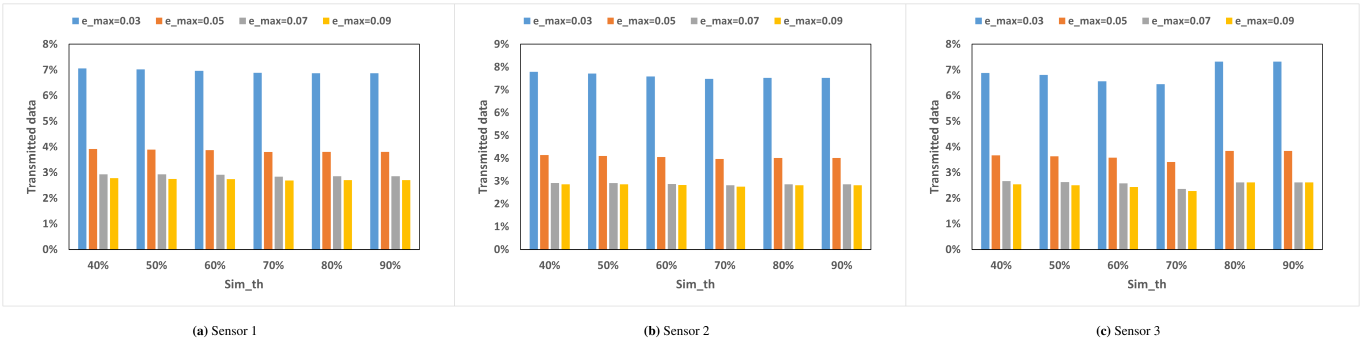


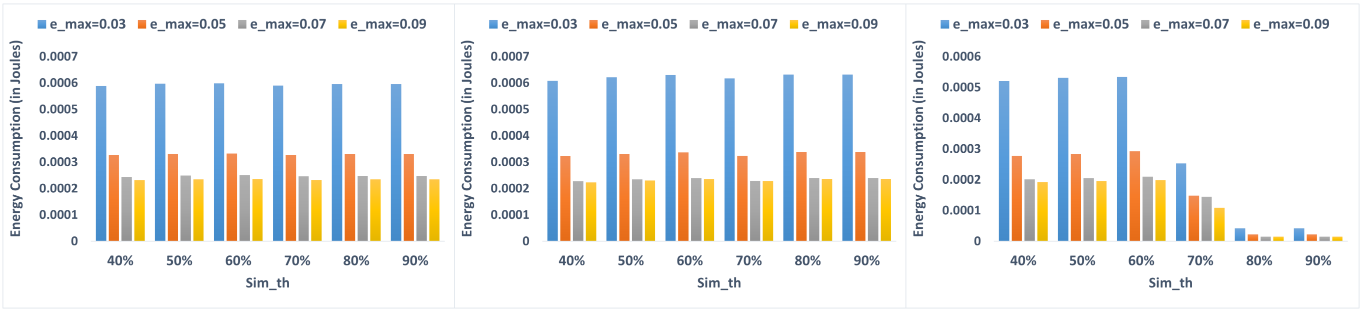
FIGURE 10 Transmitted data.

It can be seen from the results in the Figures 10 , 11 , and 12 that the suitable ratio for the threshold Sim_th is 70% because the DEDaR approach introduces better results from the energy consumption and accuracy point of view. The other ratios sometimes provide a sightly lower energy consumption but with lower accuracy. In this paper, the similarity threshold Sim_th ratio was selected, which balances between energy consumption, and data accuracy. In real world applications, this similarity threshold Sim_th ratio will be selected according to the application's need.

5.1 | Data Reduction

This experiment investigates the efficiency of the proposed DEDaR approach in removing redundant data and compressing sensed data before sending it to the Gateway.The data reduction percentage can be calculated as in Eq. (9).

$$TTR = TCR - NTR \tag{9}$$

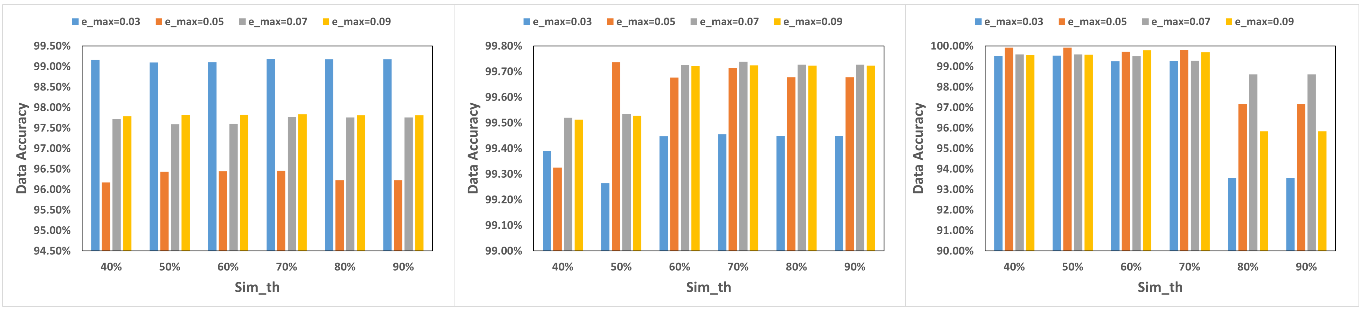


(a) Sensor 1

(b) Sensor 2

(c) Sensor 3

FIGURE 11 Energy consumption.



(a) Sensor 1

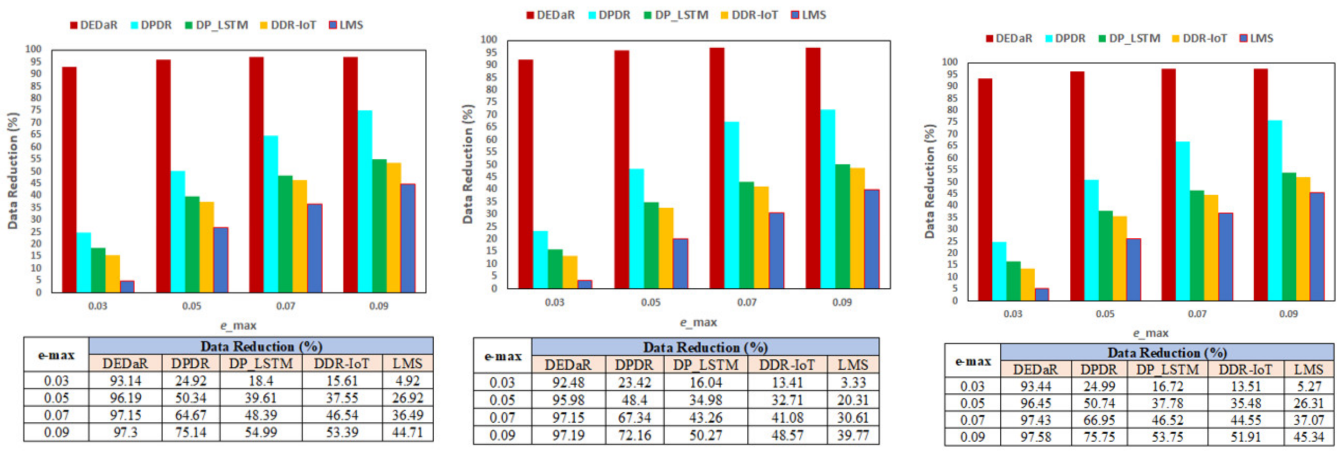
(b) Sensor 2

(c) Sensor 3

FIGURE 12 Data accuracy.

$$DPR = |(\frac{TTR}{TCR} * 100) - 100| \tag{10}$$

Where TTR, TCR, NTR, and DPR refer to the total collected readings, total transmitted readings, non-transmitted readings, and the data reduction percentage respectively. Figure 9 shows the data reduction percentage for different methods on the sensed data of sensors 1, 2, and 3 using different e_max values. Figure 13 shows the reduction percentage.



e-max	Data Reduction (%)				
	DEDaR	DPDR	DP_LSTM	DDR-IoT	LMS
0.03	93.14	24.92	18.4	15.61	4.92
0.05	96.19	50.34	39.61	37.55	26.92
0.07	97.15	64.67	48.39	46.54	36.49
0.09	97.3	75.14	54.99	53.39	44.71

(a) Sensor 1

e-max	Data Reduction (%)				
	DEDaR	DPDR	DP_LSTM	DDR-IoT	LMS
0.03	92.48	23.42	16.04	13.41	3.33
0.05	95.98	48.4	34.98	32.71	20.31
0.07	97.15	67.34	43.26	41.08	30.61
0.09	97.19	72.16	50.27	48.57	39.77

(b) Sensor 2

e-max	Data Reduction (%)				
	DEDaR	DPDR	DP_LSTM	DDR-IoT	LMS
0.03	93.44	24.99	16.72	13.51	5.27
0.05	96.45	50.74	37.78	35.48	26.31
0.07	97.43	66.95	46.52	44.55	37.07
0.09	97.58	75.75	53.75	51.91	45.34

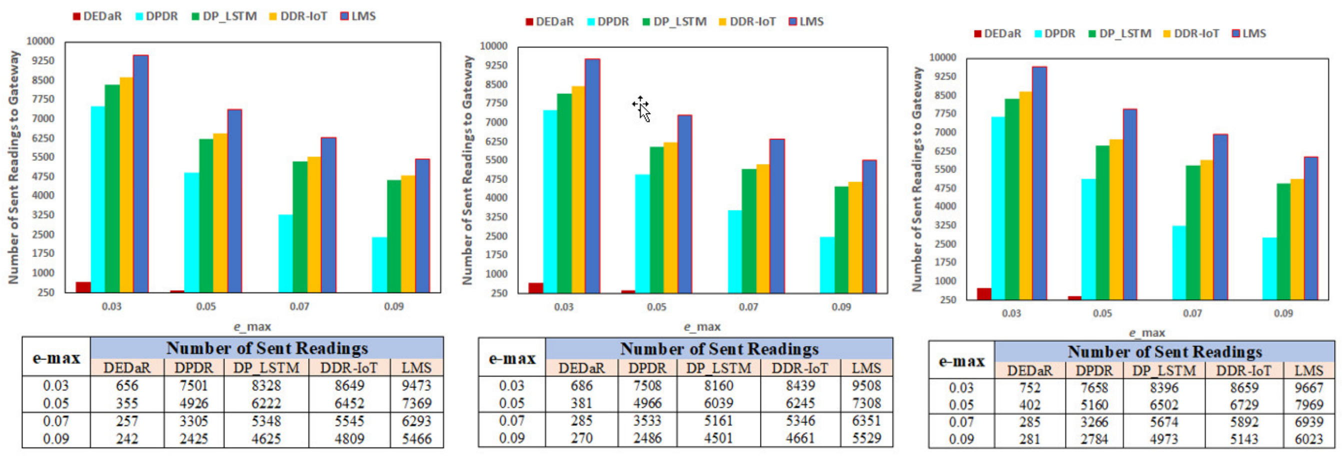
(c) Sensor 3

FIGURE 13 Data reduction percentage

It can be seen from the results of Figure 13 that the proposed DEDaR approach increased the percentage of reduction from 92.48% up to 97.58%, while the other methods introduced a percentage of reduction from 23.42% up to 75.75%, from 16.04% up to 54.99%, from 13.41% up to 53.39%, and from 3.33% up to 45.34% for DPDR, DP_LSTM, DDR-IoT, and LMS, respectively. The proposed DEDaR approach achieves better performances compared to other approaches, and it reduces the sensed data efficiently by removing the redundant data before sending it to the Gateway.

5.2 | Number of Sent Readings

In this experiment, the volume of sent data readings to the Gateway is studied. Figure 14 introduces the number of sent readings to the Gateway after applying the proposed DEDaR approach.



e-max	Number of Sent Readings				
	DEDaR	DPDR	DP_LSTM	DDR-IoT	LMS
0.03	656	7501	8328	8649	9473
0.05	355	4926	6222	6452	7369
0.07	257	3305	5348	5545	6293
0.09	242	2425	4625	4809	5466

(a) Sensor 1

e-max	Number of Sent Readings				
	DEDaR	DPDR	DP_LSTM	DDR-IoT	LMS
0.03	686	7508	8160	8439	9508
0.05	381	4966	6039	6245	7308
0.07	285	3533	5161	5346	6351
0.09	270	2486	4501	4661	5529

(b) Sensor 2

e-max	Number of Sent Readings				
	DEDaR	DPDR	DP_LSTM	DDR-IoT	LMS
0.03	752	7658	8396	8659	9667
0.05	402	5160	6502	6729	7969
0.07	285	3266	5674	5892	6939
0.09	281	2784	4973	5143	6023

(c) Sensor 3

FIGURE 14 Number of Sent Readings to Gateway

From the results in Figure 10, it can be observed that the proposed DEDaR approach reduces the transmitted data to the Gateway from 2.42% up to 7.52% compared with other approaches. The DPDR, DP_LSTM, DDR-IoT, and LMS reduced the transmission to the gateway from 24.25% up to 76.58%, from 45.01% up to 83.96%, from 46.61% up to 86.59%, and from 53.66% up to 96.67%, respectively. This can ensure that the proposed DEDaR approach is more powerful in removing the redundant data before sending it to the Gateway.

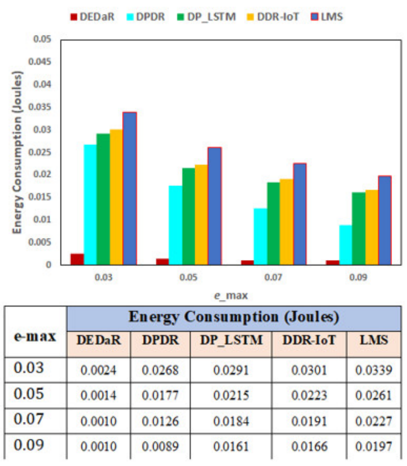
5.3 | Energy Consumption

Due to the limited resources of sensor nodes, energy consumption is critical in these devices of IoT networks. Figure 15 refers to the consumed energy inside each sensor node.

It can be noticed from the results in Figure 11 that the proposed DEDaR approach decreased the consumed energy from 90% up to 90.3%, from 91.1% up to 94.74%, from 91.4% up to 94.94%, and from 92.33% up to 95.56% in comparison with DPDR, DP_LSTM, DDR-IoT, and LMS, respectively. It can be concluded from the results that the proposed DEDaR approach saves more energy than other methods due to transmitting as little as few data as possible readings to the Gateway. This improves the performances of the network and extends its lifetime.

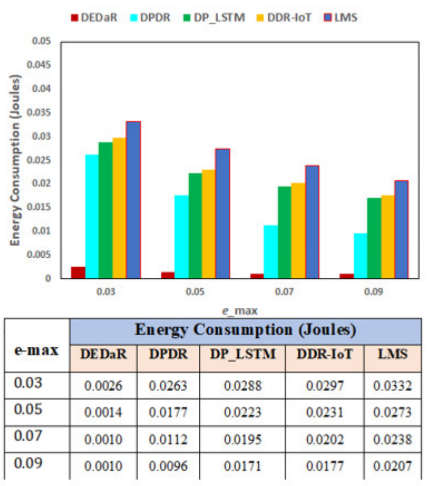
5.4 | Data Accuracy

Data accuracy can be calculated by computing the deviation between the gathered (input data) and transmitted data readings (obtained results) after employing the reduction algorithm, including the prediction algorithm. Equations (11) and (12) are used



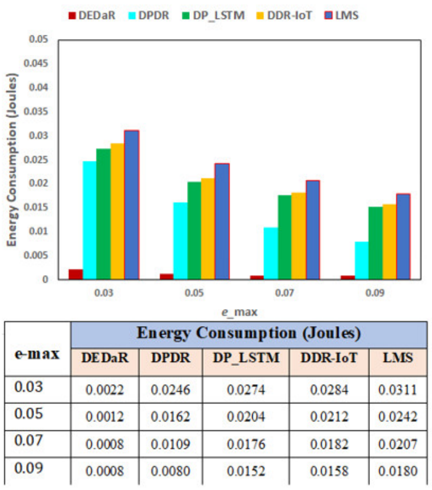
e-max	Energy Consumption (Joules)				
	DEDaR	DPDR	DP_LSTM	DDR-IoT	LMS
0.03	0.0024	0.0268	0.0291	0.0301	0.0339
0.05	0.0014	0.0177	0.0215	0.0223	0.0261
0.07	0.0010	0.0126	0.0184	0.0191	0.0227
0.09	0.0010	0.0089	0.0161	0.0166	0.0197

(a) Sensor 1



e-max	Energy Consumption (Joules)				
	DEDaR	DPDR	DP_LSTM	DDR-IoT	LMS
0.03	0.0026	0.0263	0.0288	0.0297	0.0332
0.05	0.0014	0.0177	0.0223	0.0231	0.0273
0.07	0.0010	0.0112	0.0195	0.0202	0.0238
0.09	0.0010	0.0096	0.0171	0.0177	0.0207

(b) Sensor 2



e-max	Energy Consumption (Joules)				
	DEDaR	DPDR	DP_LSTM	DDR-IoT	LMS
0.03	0.0022	0.0246	0.0274	0.0284	0.0311
0.05	0.0012	0.0162	0.0204	0.0212	0.0242
0.07	0.0008	0.0109	0.0176	0.0182	0.0207
0.09	0.0008	0.0080	0.0152	0.0158	0.0180

(c) Sensor 3

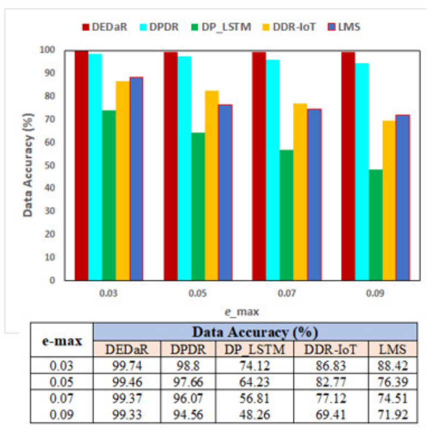
FIGURE 15 Energy Consumption

to produce the percentage of accuracy.

$$DD = \left| \frac{\sum TCR - (\sum TTR + \sum TER)}{TCR} \right| * 100 \tag{11}$$

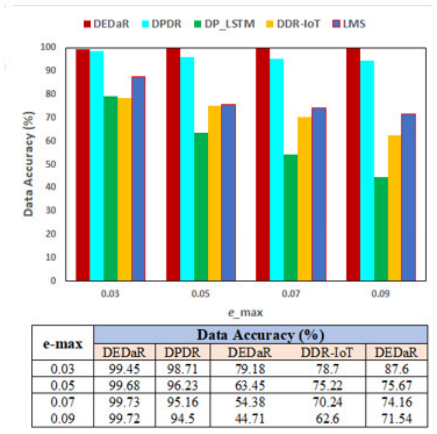
$$DA = (1 - DD) * 100 \tag{12}$$

The data deviation is DD, the total estimated readings of the sensor Si is TER, and the data accuracy is DA. The total transmitted readings are denoted by TTR, while the total collected readings are denoted by TCR. Figure 16 shows the data accuracy.



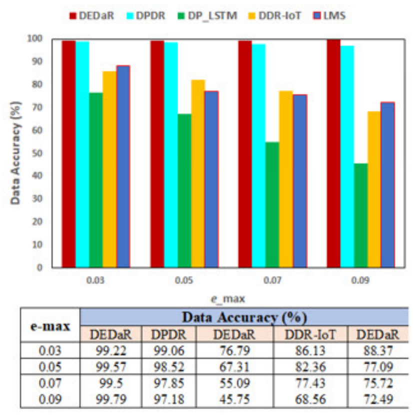
e-max	Data Accuracy (%)				
	DEDaR	DPDR	DP_LSTM	DDR-IoT	LMS
0.03	99.74	98.8	74.12	86.83	88.42
0.05	99.46	97.66	64.23	82.77	76.39
0.07	99.37	96.07	56.81	77.12	74.51
0.09	99.33	94.56	48.26	69.41	71.92

(a) Sensor 1



e-max	Data Accuracy (%)				
	DEDaR	DPDR	DEDaR	DDR-IoT	DEDaR
0.03	99.45	98.71	79.18	78.7	87.6
0.05	99.68	96.23	63.45	75.22	75.67
0.07	99.73	95.16	54.38	70.24	74.16
0.09	99.72	94.5	44.71	62.6	71.54

(b) Sensor 2



e-max	Data Accuracy (%)				
	DEDaR	DPDR	DEDaR	DDR-IoT	DEDaR
0.03	99.22	99.06	76.79	86.13	88.37
0.05	99.57	98.52	67.31	82.36	77.09
0.07	99.5	97.85	55.09	77.43	75.72
0.09	99.79	97.18	45.75	68.56	72.49

(c) Sensor 3

FIGURE 16 Data Accuracy

It can be seen from the results of Figure 16 that the proposed DEDaR approach introduces a suitable data accuracy of 92.48% up to 99.74% compared with other methods introduced of 23.42% up to 75.75%, from 16.04% up to 54.99%, from 13.41% up to 53.39%, and from 3.33% up to 45.34% for DPDR, DP_LSTM, DDR-IoT, and LMS, respectively. This will ensure that the proposed DEDaR approach provides a high reduction with suitable data accuracy at the Gateway.

The proposed DEDaR produces this high ratio of data accuracy due to losing a lower number of data readings at the sensor node. It uses efficient techniques to remove redundant data from the sensor node while maintaining adequate data quality at the gateway. The DEDaR approach uses a fixed code dictionary based on lossless Huffman encoding to compress the segments of the SAX technique, and this decreases the lost data and increases the accuracy of received data at the gateway.

5.5 | Further results and discussion

This section presents further results, analysis, and discussions about the DEDaR approach, and the findings are compared with some related works to show the efficiency of the proposed DEDaR approach. In the next experiments, the DEDaR approach uses the temperature readings during the simulation. It uses different sizes per period such as $T = 20$, 50, and 100 readings per period. Moreover, the *e_max* uses different values like 0.03, 0.05, and 0.07. The DEDaR approach is compared with DaReCA[23], Prefix Frequency Filtering (PFF)[25], and Aggregation and Transmission Protocol (ATP)[26]. Figure 17 shows the transmitted readings.

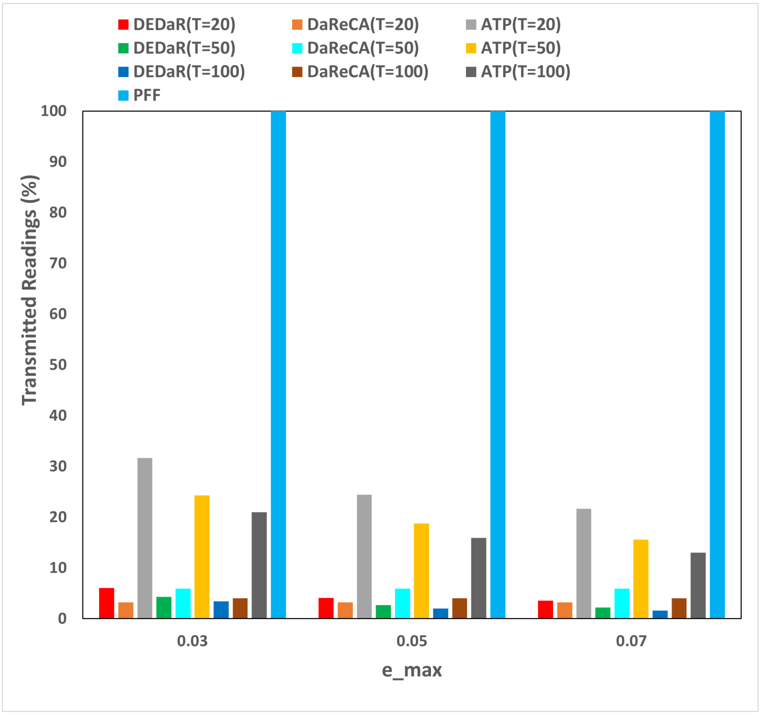


FIGURE 17 Transmitted readings

It can be observed from the results presented in Figure 17 that the DEDaR, DaReCA, ATP, and PFF approaches sent data from 1.63% up to 6.06%, from 3.2% up to 5.9%, from 13.03% up to 31.68%, and 100% respectively. It can be seen from Figure 17 that the proposed DEDaR approach reduced the transmitted data efficiently compared with other approaches. These findings ensure the efficiency of combining the prediction and compression methods in decreasing the volume of transmitted data by the sensor node.

Figure 18 refers to the energy consumed by the sensor node using the DEDaR approach compared with DaReCA, ATP, and PFF approaches.

It can be seen from the results in Figure 18 that the proposed DEDaR approach lowered the consumed energy by the sensor node from 92.82% up to 94.73%, from 97.77% up to 98.13%, and 98.15% up to 98.34% compared with DaReCA, ATP, and PFF methods. The high performance of the suggested DEDaR in reducing this volume of energy due to its use of an efficient data reduction method that reduces the sensed data before sending it by the sensor node (see Figure 17). This contributes in decreasing the consumed energy inside the sensor node. The impact of decreased data readings on the quality of collected readings at the fog gateway is investigated in this experiment. It is critical to lower the size of transferred data before sending

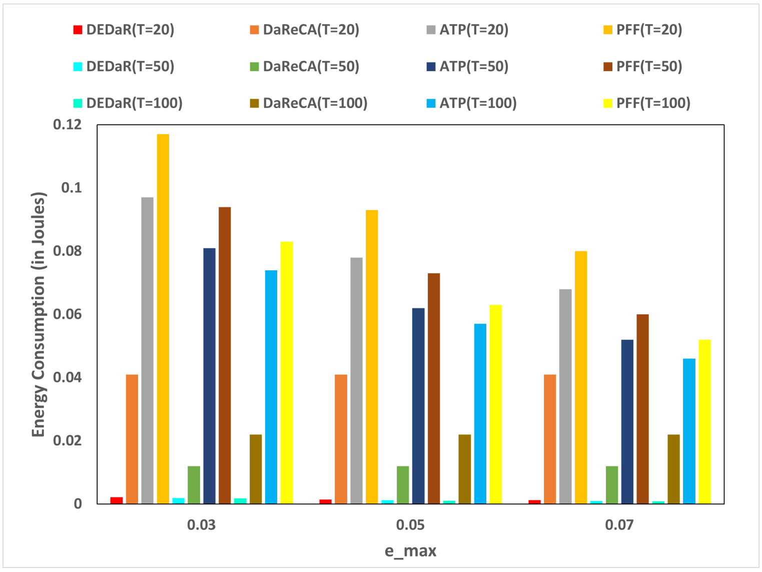


FIGURE 18 Energy consumption

them to the gateway in order to save energy, but it is also important to ensure an adequate rate of data quality at the gateway. The percentage of the data loss represents as an indicator for the data accuracy. Figure 19 refers to the percentage of the data loss.

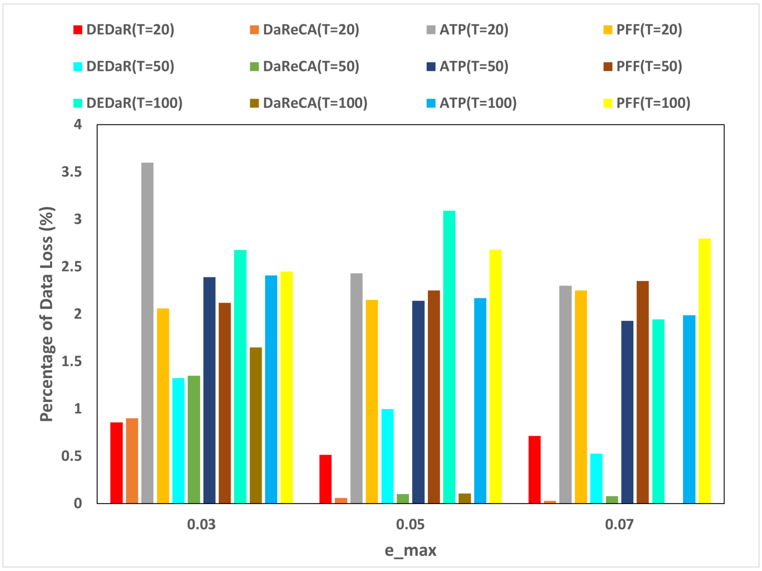


FIGURE 19 Data loss percentage

It can be noted that the DEDaR approach decreased the lost data from 0.52% up to 3.09%, while the DaReCA, ATP and PFF reduced the lost data from 0.03% up to 1.65%, from 1.93% up to 3.6% and from 2.06% up to 2.68% respectively. The proposed DEDaR approach introduced better data accuracy in most cases compared with other methods in spite of achieving higher data reduction at the sensor node. The DaReCA method introduces lower data loss in some cases because it sends a larger volume of data when it is implemented at the sensor node that participates in reducing the lost data. Hence, the proposed DEDaR could further decrease the sensed data and keep the energy of sensor nodes whilst preserving a suitable level of data quality.

6 | CONCLUSION AND FUTURE WORKS

In this paper, A Distributed Energy-efficient Data Reduction (DEDaR) Approach based on Prediction and Compression is presented to decrease data transmission in IoT Networks. The DEDaR divides time into periods. At each period, the AutoRegressive Prediction (ARP) is used to forecast the data for the following period and decide whether or not to submit the current period's data to the Gateway. When it comes to data transmission, redundant data are removed using an effective compression method based on "adaptive piecewise constant approximation" (APCA), "Symbolic Aggregate Approximation" (SAX), and lastly Fixed Code Dictionary based on Huffman Encoding. The suggested DEDaR approach outperforms other previous approaches in terms of data reduction, transmitted data size, energy consumption, and data accuracy, according to simulation findings based on real-world data. Future works will focus on reducing the transmitted data from the Gateway to the cloud using machine learning. In addition, we believe that it is possible to predict missing data at the Gateway in order to increase the accuracy of data so that it may be used in decision-making. Finally, we want to apply the suggested approach on a real network.

CONFLICT OF INTEREST

The authors declare that they have no conflict of interest.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are openly available in reference[7].

References

1. AlQurabat AKM, Idrees AK. Data gathering and aggregation with selective transmission technique to optimize the lifetime of Internet of Things networks. *International Journal of Communication Systems* 2020; 33(11): e4408.
2. Idrees SK, Idrees AK. New fog computing enabled lossless EEG data compression scheme in IoT networks. *Journal of Ambient Intelligence and Humanized Computing* 2021: 1–14.
3. Al-Qurabat AKM, Idrees AK, Abou Jaoude C. Dictionary-Based DPCM Method for Compressing IoT Big Data. In: IEEE. ; 2020: 1290–1295.
4. Al-Qurabat AKM, Abou Jaoude C, Idrees AK. Two tier data reduction technique for reducing data transmission in IoT sensors. In: IEEE. ; 2019: 168–173.
5. Akyildiz IF, Vuran MC. *Wireless sensor networks*. John Wiley & Sons . 2010.
6. Dargie W, Poellabauer C. *Fundamentals of wireless sensor networks: theory and practice*. John Wiley & Sons . 2010.
7. Madden S. Intel Berkeley Research lab. *http://db.csail.mit.edu/labdata/labdata.html* 2004.
8. Wang H, Yemeni Z, Ismael WM, Hawbani A, Alsamhi SH. A reliable and energy efficient dual prediction data reduction approach to WSNs based on Kalman filter. *IET Communications* 2021.
9. Jarwan A, Sabbah A, Ibnkahla M. Data transmission reduction schemes in WSNs for efficient IoT systems. *IEEE Journal on Selected Areas in Communications* 2019; 37(6): 1307–1324.
10. Ismael WM, Gao M, Al-Shargabi AA, Zahary A. An in-networking double-layered data reduction for internet of things (IoT). *Sensors* 2019; 19(4): 795.
11. Fathy Y, Barnaghi P, Tafazolli R. An adaptive method for data reduction in the internet of things. In: IEEE. ; 2018: 729–735.

12. Mejia J, Ochoa-Zezzatti A, Cruz-Mejía O, Mederos B. Prediction of time series using wavelet Gaussian process for wireless sensor networks. *Wireless Networks* 2020; 26(8): 5751–5758.

13. Zhang C, Liu Y, Wu F, Fan W, Tang J, Liu H. Multi-dimensional joint prediction model for IoT sensor data search. *IEEE Access* 2019; 7: 90863–90873.

14. Fathy Y, Barnaghi P. Quality-based and energy-efficient data communication for the internet of things networks. *IEEE Internet of Things Journal* 2019; 6(6): 10318–10331.

15. Almeida FR, Brayner A, Rodrigues JJ, Maia JEB. Improving multidimensional wireless sensor network lifetime using pearson correlation and fractal clustering. *Sensors* 2017; 17(6): 1317.

16. Liazid H, Lehsaini M, Liazid A. An improved adaptive dual prediction scheme for reducing data transmission in wireless sensor networks. *Wireless Networks* 2019; 25(6): 3545–3555.

17. Russo A, Verdier F, Miramond B. Energy saving in a wireless sensor network by data prediction by using self-organized maps. *Procedia computer science* 2018; 130: 1090–1095.

18. Karjee J, Kleinsteuber M. Data estimation with predictive switching mechanism in wireless sensor networks. *International Journal of Sensor Networks* 2017; 25(3): 184–197.

19. Wu M, Tan L, Xiong N. A structure fidelity approach for big data collection in wireless sensor networks. *Sensors* 2015; 15(1): 248–273.

20. Dhimal S, Sharma K. Energy conservation in wireless sensor networks by exploiting inter-node data similarity metrics. *International Journal of Energy, Information and Communications* 2015; 6(2): 23–32.

21. Alhussaini R, Idrees AK, Salman MA. Data transmission protocol for reducing the energy consumption in wireless sensor networks. In: Springer. ; 2018: 35–49.

22. Idrees AK, Alhussaini R, Salman MA. Energy-efficient two-layer data transmission reduction protocol in periodic sensor networks of IoTs. *Personal and Ubiquitous Computing* 2020: 1–20.

23. Idrees AK, Abou Jaoude C, Al-Qurabat AKM. Data reduction and cleaning approach for energy-saving in wireless sensors networks of IoT. In: IEEE. ; 2020: 1–6.

24. Idrees AK, Al-Qurabat AKM. Energy-efficient data transmission and aggregation protocol in periodic sensor networks based fog computing. *Journal of Network and Systems Management* 2021; 29(1): 1–24.

25. Bahi JM, Makhoul A, Medlej M. A two tiers data aggregation scheme for periodic sensor networks.. *Adhoc & Sensor Wireless Networks* 2014; 21(1).

26. Harb H, Makhoul A, Couturier R, Medlej M. ATP: An aggregation and transmission protocol for conserving energy in periodic sensor networks. In: IEEE. ; 2015: 134–139.

27. Zhu R, Yu M, Li Y, Wang J, Liu L. Edge sensing-enabled multistage hierarchical clustering deredundancy algorithm in WSNs. *Wireless Communications and Mobile Computing* 2021; 2021.

28. Rida M, Makhoul A, Harb H, Laiymani D, Barhamgi M. EK-means: A new clustering approach for datasets classification in sensor networks. *Ad Hoc Networks* 2019; 84: 158–169.

29. Loganathan D, Balasubramani M, Sabitha R. Energy Aware Efficient Data Aggregation (EAEDAR) with Re-scheduling Mechanism Using Clustering Techniques in Wireless Sensor Networks. *Wireless Personal Communications* 2021; 117(4): 3271–3287.

30. Alam M, Aziz AA, Latif S, Awang A. Error-aware data clustering for in-network data reduction in wireless sensor networks. *Sensors* 2020; 20(4): 1011.

31. Shahina K, Pradeep Kumar T. Similarity-based clustering and data aggregation with independent component analysis in wireless sensor networks. *Transactions on Emerging Telecommunications Technologies*: e4462.

32. Tsiropoulou EE, Paruchuri ST, Baras JS. Interest, energy and physical-aware coalition formation and resource allocation in smart IoT applications. In: IEEE. ; 2017: 1–6.

33. Idrees AK, Al-Qurabat AKM, Abou Jaoude C, Al-Yaseen WL. Integrated divide and conquer with enhanced k-means technique for energy-saving data aggregation in wireless sensor networks. In: IEEE. ; 2019: 973–978.

34. Hyndman RJ, Athanasopoulos G. *Forecasting: principles and practice*. OTexts . 2018.

35. Korstanje J. The Decision Tree Model. In: Springer. 2021 (pp. 159–168).

36. Zifan A, Saberi S, Moradi MH, Towhidkhah F. Automated ECG segmentation using piecewise derivative dynamic time warping. *International Journal of Biological and Medical Sciences* 2006; 1(3).

37. Wang Y, Wang P, Pei J, Wang W, Huang S. A data-adaptive and dynamic segmentation index for whole matching on time series. *Proceedings of the VLDB Endowment* 2013; 6(10): 793–804.

38. Malinowski S, Guyet T, Quiniou R, Tavenard R. 1d-sax: A novel symbolic representation for time series. In: Springer. ; 2013: 273–284.

39. Cassisi C, Montalto P, Aliotta M, Cannata A, Pulvirenti A. Similarity measures and dimensionality reduction techniques for time series data mining. *Advances in data mining knowledge discovery and applications'(InTech, Rijeka, Croatia, 2012,* 2012: 71–96.

40. Sayood K. *Introduction to data compression*. Morgan Kaufmann . 2017.

41. Heinzelman WR, Chandrakasan A, Balakrishnan H. Energy-efficient communication protocol for wireless microsensor networks. In: IEEE. ; 2000: 10–pp.

How to cite this article: Hussein A.M., Idress A.K., and Couturier R. (2022), Distributed Energy-efficient Data Reduction Approach based on Prediction and Compression to reduce data transmission in IoT networks, *Int. J. Commun. Syst.*, .