HAL
open science

K-Predictions Based Data Reduction Approach in WSN for Smart Agriculture

Christian Salim, Nathalie Mitton

To cite this version:

Christian Salim, Nathalie Mitton. K-Predictions Based Data Reduction Approach in WSN for Smart Agriculture. Computing, 2020, 10.1007/s00607-020-00864-z . hal-02975997

HAL Id: hal-02975997

https://hal.science/hal-02975997v1

Submitted on 23 Oct 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Noname manuscript No.
(will be inserted by the editor)

K-Predictions Based Data Reduction Approach in WSN for Smart Agriculture

Christian Salim and Nathalie Mitton

the date of receipt and acceptance should be inserted later

Abstract Nowadays, climate change is one of the numerous factors affecting the agricultural sector. Optimising the usage of natural resources is one of the challenges this sector faces. For this reason, it could be necessary to locally monitor weather data and soil conditions to make faster and better decisions locally adapted to the crop. Wireless sensor networks (WSNs) can serve as a monitoring system for these types of parameters. However, in WSNs, sensor nodes suffer from limited energy resources. The process of sending a large amount of data from the nodes to the sink results in high energy consumption at the sensor node and significant use of network bandwidth, which reduces the lifetime of the overall network and increases the number of costly interference. Data reduction is one of the solutions for this kind of challenges. In this paper, data correlation is investigated and combined with a data prediction technique in order to avoid sending data that could be retrieved mathematically in the objective to reduce the energy consumed by sensor nodes and the bandwidth occupation. This data reduction technique relies on the observation of the variation of every monitored parameter as well as the degree of correlation between different parameters. This approach is validated through simulations on MATLAB using real meteorological data-sets from Weather-Underground sensor network. The results show the validity of our approach which reduces the amount of data by a percentage up to 88% while maintaining the accuracy of the information having a standard deviation of 2 degrees for the temperature and 7% for the humidity.

Keywords Smart Agriculture; Data Correlation; Data Reduction; Data Prediction; Pearson coefficient; WSN

Christian Salim and Nathalie Mitton
Inria, France
E-mail: christian.salim@inria.fr; nathalie.mitton@inria.fr

1 Introduction

Modern agriculture is in need of new and improved methods. Climate change and water scarcity present some new challenges. Intelligent decision support tools are becoming primordial to address some challenges and maximise the usage of natural resources [1]. Wireless sensor networks (WSN) serve as low-cost monitoring systems in different domains (healthcare, industrial, video surveillance, environmental, etc...) [2]. In the agricultural fields, WSN can also be beneficial to survey the climate and soil parameters. In our scenario, we assume a WSN deployed for smart agriculture, periodically gathering environmental data from different sensor nodes and sending this data to a sink for further analysis [3]. This periodic cycle leads to a lot of redundant data sent to the sink especially if no changes occur in the monitored feature (e.g. if the temperature stays stable). For the same parameter, monitored by several sensor nodes, data redundancy can be present in time and space, on the same node for consecutive values, or between two different nodes for the same value sensed at the same time. This big amount of periodic data is even more challenging at the sensor node level. For instance, for the temperature, each node may capture and send a value every 30 minutes, 48 values per day as achieved for every node in the weather underground network around the globe[1]. And this is being performed by hundreds of nodes and not only for temperature but also for other parameters such as humidity, pressure, wind speed, etc... Thus, WSNs must take account of the medium occupancy of the network, the bandwidth usage and most importantly the limited energy resources of the sensor nodes to ensure a long system lifetime with minimum maintenance. To face those challenges, data reduction under different forms is one of the solutions that can be adopted and adapted to the situation under study.

In this paper, data reduction is executed by coupling two algorithms, the machine learning data reduction algorithm (MLDR) [4] and the Pearson Data Correlation and Prediction algorithm (PDCP) proposed in this paper to enhance the data reduction. In the former part, on a single node, the MLDR algorithm is a data reduction technique based on a machine learning process used to predict on a dual prediction basis [5], the next values of the monitored feature (e.g: temperature) both at the sensor node and at the sink simultaneously [5]. This algorithm uses the past values to compute the trend of variation of any monitored parameter in able to predict its next values. In the latter part, based on the predictions of the features from MLDR, the PDCP algorithm completes the prediction process by applying a data correlation technique based on the Pearson correlation coefficient which depends on numerous past values [6] between the same parameter on different neighbour nodes and between different parameters on a single node. If a high correlation is detected, less data is sent to the sink. In spatial or inter-nodes correlation where the correlation is computed between the same value sensed by different nodes, only one of the nodes may send the data, reducing interference at the

[1] https://www.wunderground.com

same time. In the different parameters case, e.g temperature/pressure, temperature/humidity, etc, where correlation is computed in a node (intra-node correlation), the node only sends one out of the 2 correlated data, and the sink estimates the other one by applying the same technique.

As such, MLDR and PDCP reduce the energy consumption of the communication process, since the nodes do not need to send all the captured values to the sink anymore while still ensuring high data accuracy at the sink. However a critical threshold between the estimated data and the real one is always present at the sensor node level to detect any absurd change in the values. If this threshold is surpassed by a new real value, the real value is sent and the process is repeated to compute a new trend or to measure the correlation. In this approach, while combining MLDR and PDCP algorithms, a triple prediction process takes place, 1 real value can ensure predicting several other values of different parameters. In the example stated below, the temperature and the humidity values are taken into consideration on S1 and S2 that are two neighbour nodes. We consider that the temperature on S1 is predicted through MLDR based on a real value which makes the first prediction. Based on PDCP, S1 and S2 temperature parameters are correlated so the temperature values on S2 are predicted through the values on S1 as the second prediction. The third prediction is based on the highly correlated temperature and humidity on S2, so the humidity values are predicted based on the temperature predictions. This process leads to a greater data reduction, thus, more energy consumption reduction. We conducted simulation using Matlab simulator and the results show the validity of our approach, by reducing the amount of sent data to the sink more than 88% outperforming other approaches. In those simulations, we studied the temperature and the humidity on 2 different sensor nodes in the same region while applying both algorithms.

The remainder of this paper is as follows: Section 2 introduces the state of the art, Section 3 presents the MLDR algorithm and Section 4 clarifies the behaviour of MLDR algorithm. In Section 5 the data correlation technique is explained. The PDCP algorithm is presented in Section 6. Section 7 presents evaluation results. Finally, Section 8 concludes the paper.

2 Background and Related Work

Data reduction for WSN has already received much attention in the last years. Different data reduction techniques are present in the literature. We can sum up the most popular methods as follows: *clustering* by detecting similarities in spatio-temporal data [7]. Adapting the sampling rate of the nodes is a solution to reduce the size of the data sent to the sink [8,9]. Machine learning for data prediction is widely used for data reduction [5,10]. However, all these methods can serve as complementary solutions for our data reduction technique which is based on *data correlation*. The main purpose of our work is to reduce the amount of transmitted data from the nodes to the sink. Data correlation is widely used in data reduction techniques, it tests the correlation between

several characteristics. In the literature, a lot of data reduction approaches based on data correlation have been proposed. Some papers investigated data correlation and data similarity between several features as in [11],[12],[13],[14] and [15]. Authors in [12] present a Bayesian Inference Approach to detect data with high spatio-temporal correlated data, to avoid transmitting data that can be reconstructed from another data such as temperature and humidity in some cases. However this method presents one level of prediction by predicting one parameter from another one, in our approach one parameter value can help predicting more than two other parameters values. Thus, predicting different parameters helps to furthermore reduce the amount of data sent to the sink. Machine learning for data prediction is widely used for data reduction as in [5, 16,17,10,4] and [8]. In the dual prediction model [5], the sensor node and the sink both predict the next values of the monitored feature simultaneously. In [5], the authors propose a machine learning technique AS+TR to predict the next values, while sending all the data in the learning phase to the sink. In this approach, they take the uniform trend into consideration but do not consider a changeable trend between consecutive values as we do. Also they do detect a trend directly after a single change, which can cause some problems for the learning process and send more values. In our method the evolution of the trend for several values is studied before establishing the prediction process of the next values for the same parameter. This limits the re-computation of the trend after every change and does not require sending the new real values unless a drastic change happened. It reduces the energy consumption for the processing and the transmission processes.

Other machine learning based methods have been proposed in the literature for data prediction for data reduction. A lot of approaches were interested in data correlation for this purpose, mainly using the Pearson correlation technique and its derivatives [6,18–20], the Auto Regression model [21–23], the long short-term memory (LSTM) networks [24,25] and the convolutional networks techniques [26–29]. The authors in [6] proposed a methodology to analyse data streams, based on spatio-temporal correlations using Pearson correlation. They evaluated their approach using smart cities traffic data collected from the road sensors in the city of Aarhus in Denmark. They created a spatio-temporal traffic model in smart cities with IoT data coming from road sensors. The data is collected via Bluetooth sensors that measure the number of cars in a road, attached to light poles. These sensors embed a GPS receiver that provides location data. The vehicle count was the parameter taken into consideration in their approach. In this method, the nodes are used to capture data, and the computation is run on a remote server. Nevertheless, the nodes in this method were not energy limited, they were attached to light poles for power and energy reasons. In our approach, the nodes might be in remote areas such as an agriculture field and the algorithms are executed at the sensor node level for quicker decision making. In [24], LSTM networks are proposed by the author to predict solar irradiance hourly day-ahead using weather forecasting data. The prediction model is trained by LSTM networks taking account of the dependence between consecutive hours of the same day. For single out-

put prediction, this algorithm outperforms several other approaches for solar irradiance prediction. Nevertheless, in order to predict and train the model, a big amount of data is needed, which makes it difficult to be done at the energy-limited sensor nodes. The authors in [26] proposed a Convolutional LSTM network to enhance the forecasting of the precipitation now-casting. This approach builds an end-to-end trainable model to solve the issue capturing spatio-temporal correlations by stacking multiple ConvLSTM layers and forming an encoding-forecasting structure. In our approach, we are interested to execute algorithms directly at the sensor node level, however, this type of convolutional layers for training would be time and energy consuming on energy-limited sensor nodes.

In all the stated techniques above, some authors proposed that the processing is to be done on the sink, others proposed complex solutions which can be harmful for powerless sensor nodes. Some proposed prediction techniques while not focusing a lot on how to increase the data reduction while maintaining the accuracy. Other approaches studied the correlation between several parameters to increase the data reduction while using complex systems and equations. In this paper, we are interested in the on-node processing since it is better to take actions instantly and locally if any abnormal situation occurs. We tackle the challenge of overall network efficiency through data reduction in wireless sensor networks deployed to monitor the environmental parameters for agricultural surveillance taking account of the limited energy resources of every sensor node. Our solution is to implement a light data reduction algorithm based on trend variation and data correlation using Pearson correlation. "Light data reduction algorithm" means that we are proposing an efficient algorithm with the least complexity and the maximum simplicity in equations while maintaining an acceptable accuracy. we compared our approach with two on-node processing techniques in [5] and [12]. The first part (trend variation) is compared to [5] where the authors present a technique similar but more complex than our approach to compute the trend and predict the next values of a predefined parameter. The data correlation part is compared to [12] taking the same example of data correlation between the humidity and temperature. The authors in [12] are interested in the data reduction by studying the correlation between parameters on the same node. Our technique studied the correlation differently in a less complex way and also the correlation on different nodes is added, which furthermore increases the data reduction in the overall network.

3 Machine Learning based Data Reduction algorithm (MLDR)

3.1 Overview

In this section, we present the Machine Learning based Data Reduction algorithm (MLDR) and its main principles. In MLDR, while capturing data, the nodes start a learning phase to detect the behaviour of the monitored value

(later referred as trend in our paper). A trend is detected once the variation of the given parameter over time is stable. This means when the next value differs from the last one by the same amount for several consecutive values (e.g temperature passing from 9 to 10 to 11 degrees, it means the trend is 1 degree per 1 time unit). Different forms of trends exist and are explained later in this section. Once a trend is detected, the sensor node sends this trend to the sink with the last value of the monitored value from the learning phase. Afterwards, all the real measurements are not sent to the sink unless a critical change is detected depending on predefined thresholds. The sink and the sensor node start a dual prediction mechanism to predict the next values based on the last sent value (and not the last sensed value) and the trend. In other words, they both compute an estimate of the monitored value by using same data and same equations. Meanwhile, the sensing process is not affected at the sensor node side. For each new captured value, the sensor node compares the predicted value with the real captured value. If a critical change is noticed (even in the learning phase), the sensor node sends the critical value to the sink as an information that a change happened, so in order to stop unnecessary predictions at the sink level since they would be erroneous. Then, the node enters the learning phase again, trying to detect a new trend. In the MLDR algorithm proposed in [4], the sensor node sends a hold message to the sink to stop the predictions and goes back to the learning phase trying to detect a new trend from the new values.

3.2 Machine Learning based Data Reduction Algorithm

In this section, we explain in detail the machine learning based data reduction algorithm (MLDR) (Algorithm 1). In this algorithm, the sensor node sends the first captured value val_0 to the sink. Then the sensor node enters a learning phase for n consecutive values to detect the changing trend tr between those values. To define n, we should take into consideration that a big number causes a huge processing which increases the energy consumption but at the same time increases the accuracy. However, a very small amount of values decreases the accuracy, we should find the best compromise.

To compute the trend, the sensor node must compare the differences between n consecutive values as follows:

$$tr_i = val_{i+1} - val_i \tag{1}$$

$$d_i = |tr_{i+1} - tr_i| \tag{2}$$

where vector $tr\{tr_0; tr_1; ...; tr_i\}$ represents the trends between values (the change of the values over time) and vector $d\{d_0; d_1; ...; d_i\}$ represents the differences between those trends.

3.3 Trend Behaviour

The trend behaviour is used to predict the next values. It is computed in parallel at the node and at the sink on the basis of the values sent by the sensor node.

We assume that predicting a weather value based on its history is like predicting a trajectory. We thus adopt the kinematics functions for this type of prediction replacing the movement, speed and acceleration by value (val), trend (tr) and evolution (e) respectively. In Equation 3, the trajectory function represents the new values of the feature in time:

$$val = \frac{1}{2}e \times t^2 + tr_0 \times t + val_0 \tag{3}$$

The trend is represented by the speed function which is the first derivative of the trajectory function.

$$tr = e \times t + tr_0 \tag{4}$$

The evolution e of the trend is represented by the acceleration which is the first derivative of the speed and the second derivative of the trajectory. It can be stable (evolution $e = 0$) or unstable ($e = constant$ or unstable e). Different cases can occur: stable trend, unstable trend with stable or unstable evolution.

3.3.1 Stable Trend

If the trend is stable, it means the evolution is null $e = 0$ or that the differences between consecutive trends are negligible and do not surpass a certain threshold. In this case, the node will send the nth value with the last trend computed to the sink. The stability is detected when Equation 5 holds.

$$d_i < th_{tr} \qquad \forall \quad 0 < i <= n \tag{5}$$

where th_{tr} is the threshold to define stability (it can be learnt through several days and/or multiple observations or predefined by specialists).

Once the trend is sent to the sink with the latest value, both the sensor and the sink can predict the next values using the kinematic functions defined in 3 and 4 with $e = 0$ and $t = 1$ for every period. In the case of an absurd change, the new value is sent directly to the sink even if the node is in the learning process. The difference between the captured value and the predicted value $pval_i$ at a period i must not surpass this threshold (Equation 6).

$$|val_i - pval_i| < th_{cr} \tag{6}$$

3.3.2 Unstable Trend

A trend is referred as unstable when it evolves over time, the evolution of this trend can be stable or unstable as explained in the sections below.
Stable Evolution: If the values of the feature are uniformly accelerated or decelerated, the evolution is constant but not null. In this case, in the learning phase, if the trend is not stable but its evolution is stable for n consecutive samples, the mote sends the last value, the trend and its evolution to the sink. Once the sink receives the data, the prediction process starts. The trajectory function represents the new values of the temperature with time based on the trend and evolution for each period ($t = 1$) as shown in Eq. 7.

$$\Delta val = \frac{1}{2}e + tr_0 \tag{7}$$

The trend is represented by the speed function which is the first derivative of the trajectory function and for $t = 1$ as set in Eq. 8.

$$\Delta tr = e \tag{8}$$

Unstable Evolution: If the values of the feature are accelerated or decelerated in a non uniform way, the evolution Δe is unstable. In this case, the sensor node remains in the learning phase until one of the above situations is detected (stable trend or unstable trend with stable evolution). Beside staying in the learning phase and taking off the prediction process, the sensor node will only send the values that exceed a certain threshold as shown in Eq. 6.

4 Case Study

In this section, we are interested to study the behaviour of MLDR algorithm once a change occurs. Our case study is done on the stable trend scenario between values. We consider that a trend is computed and the dual prediction process is taking place. In the Hold scenario, when a change occurs, the sensor node sends to the sink a hold message (to stop useless predictions) and the value where the change happened. The sensor then computes a new trend based on several consecutive captured values as shown in Fig. 1 and Algo. 1. In this scenario, we may lose some updates if the sensor needs time to detect the new trend or if a new trend does not exist. However, if a big change occurs, it will be directly detected and sent to the sink if it surpasses a certain predefined threshold even in the learning phase. Once a hold message is sent to the sink, the latter keeps the last value as a valid one, until a new value is received. The accuracy of the data is poorly affected since the node sends again a new value if a drastic change happened (a critical threshold th_{cr} is always present).

Algorithm 1 Machine Learning Data Reduction Algorithm MLDR on each node S_i

```
Send the first Value val_0
Set val_cmp=val_0, th_cr, th_tr, i = 1, tr = 0, e = 0        # Enter the Learning Phase
Set counter c = 0
while Energy > 0 do
   for each period i do
      if ||val_i − val_cmp|| > th_cr then
         Send val_i ; Set val_cmp=val_i
      end if
      Compute d_i, tr_i, e_i; i = i + 1
      if d_i > th_tr then
         Restart the Learning phase
         c = 0
      else
         c = c + 1
      end if
      if c = 2 then
         Go to the Prediction Process
      end if
   end for
   Send val_i, tr=tr_i, e=e_i                                # Start the Prediction Process
   Set pval_i=val_i and val_cmp=val_i
   for each period i do
      Compute pval_i                                         # using Eq. 3 and 4
      val_cmp=pval_i
      if ||val_i − val_cmp|| > th_cr then
         Send val_i and HOLD message
         Go to the Learning phase
      end if
   end for
end while
```

4.1 MLDR Behaviour

Fig. 1 shows the MLDR behaviour. In this figure, the minimal number of values to detect a trend is $n = 4$. Once the trend is detected, the sensor sends it to the sink with the last captured value. The prediction process starts at the node and at the sink levels simultaneously. As shown in Fig. 1, the sensor node re-enters a learning phase once a change occurs. While in the learning phase, the node sends a new critical value (surpassing the threshold). The second learning phase needed 7 to 8 values to detect the trend because of non stability in the captured values.

5 Data Correlation

To furthermore increase the data reduction in this type of networks on the sensor node level, the Pearson correlation method is used to detect correlations between parameters and in consequence to reduce the amount of redundant data sent from the nodes to the sink while maintaining the needed accuracy

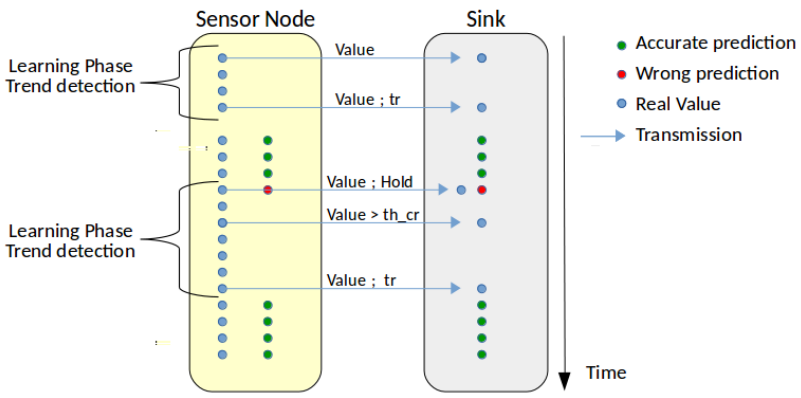

Fig. 1 Sensor node behaviour using MLDR algorithm

of the sensed data. We will run two data correlation based mechanisms: *1)* Inter-nodes data correlation; *2)* Intra-node data correlation. Fig. 2 illustrates the the difference between inter-nodes and intra-node.

In the inter-nodes data correlation, Pearson correlation coefficient [20] is used to compute the percentage of data similarity between a same type of data (e.g the similarity between the temperature) sensed by different neighbour nodes. If this similarity is considered high, the values offer redundant information. As a data reduction process, only one of the sensor nodes sends the value to the sink while maintaining the information depending on one or multiple criteria especially the residual energy as explained in the next section.

In the intra-node data correlation technique, data correlation is evaluated on the same node but between different types of data. The purpose is to try to extract one parameter value from another one by using effective and simple arithmetic operations taking account of the energy limitations on the sensor node level which denies us the usage of an auto-regression technique like ARMA [21] or LSTM technique [26] (e.g compute the correlation between the temperature and the humidity to send one of the two values to the sink and extract the other one based on the correlation and machine learning).

5.1 Inter-Nodes Data Correlation

In this part, the correlation coefficient between a same measure from different nodes is computed using the Pearson correlation coefficient method. However computing the Pearson correlation coefficient requires several values of the same data parameters from different sensor nodes (e.g temperature, wind speed, humidity, etc...). Several ways exist to compute this coefficient. The

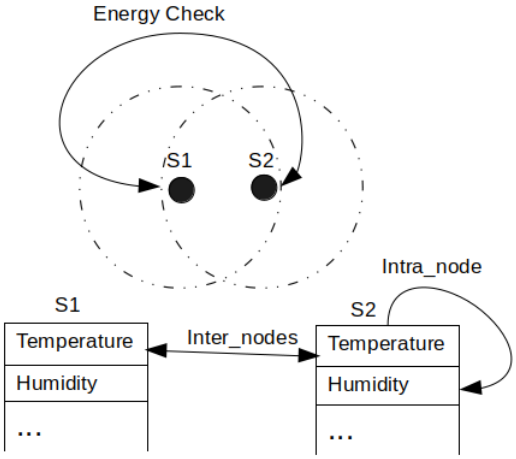

Fig. 2 Inter-Nodes and Intra-Node difference

normal and most known formula is presented in the equation below:

$$\rho_{X_1X_2} = \frac{\sum_{i=1}^{n}(X_{1_i} - \overline{X_1})(X_{2_i} - \overline{X_2})}{\sqrt{\sum_{i=1}^{n}(X_{1_i} - \overline{X_1})^2}\sqrt{\sum_{i=1}^{n}(X_{2_i} - \overline{X_2})^2}} \tag{9}$$

In our work, we used the geometric interpretation of the Pearson correlation. This technique needs several values to detect the percentage of correlation between two parameters of the same type on two different sensor nodes. This method does not take account of the case when the parameters under consideration are proportional or inversely proportional. For this purpose, Equation 10 is multiplied by a sign parameter s to have the right positive and negative values, it takes two values 1 or -1. As an example we consider two vectors of five values each X_1 and X_2: X_1=(2, 3, 6, 7, 9) ; X_2= (0.12, 0.13, 0.16, 0.17, 0.19). The correlation coefficient is computed as follows:

$$\rho_{X_1X_2} = \cos\theta = \frac{\overrightarrow{X_1}.\overrightarrow{X_2}}{\|X_1\|.\|X_2\|} \times s \tag{10}$$

where $\rho_{X_1X_2}$ represents the Pearson correlation coefficient. In this example, $\rho_{X_1X_2}$=0.99, however the percentage of correlation is the square of the correlation coefficient times 100 as shown in the equation below:

$$\rho_p = \rho^2_{X_1X_2} \times 100 \tag{11}$$

In this case the two vectors are 99.8% correlated.

This method is applied on every sensor node to compute the correlation coefficient with all its neighbour nodes in its communication range. We assume that the information needed to compute the correlation coefficient together

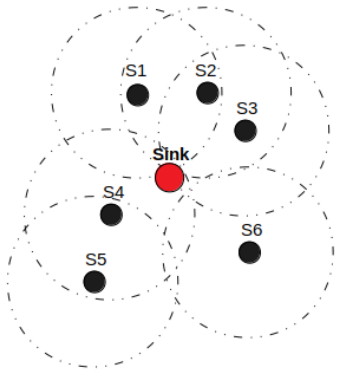

Fig. 3 Network Scenario

with the remaining energy of each node is piggybacked in the Hello messages used for each node to discover each other. A threshold of correlation is predefined on each node for each type of data. The whole process is executed at the very start of the network. The mean ratio of difference R_m is computed between the two parameters after n consecutive values as written below:

$$R_m = \frac{\sum_{i=0}^{n} \frac{y_i}{x_i}}{n} \tag{12}$$

Then, the sink computes the "missing" value of y_{n+1} based on the values of ρ, x_{n+1}, x_n, y_n, R_m and the sign value s as follows:

$$y_{n+1} = y_n + (x_{n+1} - x_n) \times R_m \times \rho^2 \times s \tag{13}$$

Once two nodes detect a high correlation on one or several parameters, they locally decide whether to send the message based on the residual energy of each other. Only the one with the highest energy sends the message. In case of ties, the message will be sent by the node with the smallest identifier. This does not require any additional exchange between nodes, the needed information (Id, battery level) being included in Hello messages.

Let's consider Fig 3 to illustrate our approach. S2 and S3 are neighbour nodes (they can directly communicate with each other) and can apply the Pearson correlation coefficient equation to detect the percentage of correlation between their parameters. Let's assume a high correlation, the sensor-node that has more residual energy between S2 and S3 will send the data to the sink. This same sensor sends the ratio R_m at the start. The sink computes the value of the other parameter based on R_m and the real values from the sending sensor node (one real value and one predicted value).

5.2 Intra-Node Data Correlation

Data correlation can be computed with different types of data on the same node. For example the correlation between humidity and temperature can be high and follows a certain shape, in this case we can extract one value from the other one. This method is applicable on every sensor node trying to find correlations between several types of parameters such as temperature, humidity or wind speed. If any important correlation is found, it will help reduce the amount of data sent from the node to the sink by sending only one of the two correlated parameters. The sink can extract the second parameter value based on the received one using some formulas as explained below.

The Pearson Correlation Coefficient is also applied in this kind of data reduction process. It is used to detect if the correlation between two different parameters is sufficient to start the prediction. The correlation coefficient ρ_{xy} between two different parameters is compared to a predefined threshold of correlation th_{cor}. Two parameters are considered correlated if Eq. 14 holds :

$$\rho_{xy} > th_{cor} \tag{14}$$

where x and y are two vectors of several values representing two different parameters such as the temperature and the humidity. We compute the Pearson correlation coefficient as mentioned in Equation 10. If Equation 14 holds, we compute the mean of the ratio R_m for the same number of values n that was needed to compute the correlation coefficient. We apply the equations 12 and 13 to compute R_m and the next value y_{n+1} based on the values of ρ, x_{n+1}, x_n, y_n and R_m.

Meanwhile the sensor nodes are always sensing the real values of all the parameters. The predicted value must be in a certain range based on the real value to be accepted and to continue the prediction process. This range is defined by and upped and lower thresholds based on the correlation coefficient as follows:

$$th_{up} = (\frac{1-\rho_{xy}}{2} + 1) \times y_r \tag{15}$$

$$th_{low} = (\frac{1-\rho_{xy}}{2} + \rho_{xy}) \times y_r \tag{16}$$

where R_y is the real value of parameter y. In this case, if $\rho_{xy} = 0.8$, it is considered at 0.2 far from the perfect correlation 1. The upper threshold $th_{up} = 1.1 \times y_r$ and $th_{low} = 0.9 \times y_r$

The node must be able to make a decision by comparing the real sensed value with the predicted one. For this purpose, the dual prediction model is adopted from [4]. In this scenario, the node and the sink predict the new values at the same time by applying the same equations. If the predicted value falls outside this range, the node sends the real value to the sink and the prediction process continues based on the new real value. However, the correlation coefficient is not recomputed unless we have 2 consecutive values falling outside of the range.

6 Scenario and Algorithm

In this section, we discuss the scenario where the Pearson Data Correlation and Prediction algorithm PDCP is implemented. Fig 3 shows several neighbouring sensor-nodes where different neighbour couples can be formed trying to find correlation between their parameters (inter-nodes). This is done while implementing the intra-node correlation on each node to detect the degree of correlation between different parameters on each node. The PDCP algorithm is detailed in 2. The intra-node correlation is always applicable on each sensor-node, however, the inter-nodes correlation depends on the neighbouring parameters (distance and radius of communication). E.g, in Fig 3, the whole algorithm of inter-nodes and intra-node correlation techniques can be applicable on several neighbour couples such as $\{S1, S2\}.\{S2, S3\}$ and $\{S4, S5\}$.

In Algorithm 2, the number of values needed to compute the correlation depends on the scenario and the studied application. Later on, in our experiments, we consider that $n = 48$ values, it means a full day of values is needed to compute the correlation. In this scenario and algorithm, for the intra-node correlation, we focused on the temperature and humidity correlation specifically which is represented by ρ_{hute} in the algorithm. For the inter-nodes correlation, we computed the correlation for the temperature in Sensors S1 and S2.

Algorithm 2 Pearson Data Correlation and Prediction Algorithm PDCP on each node S_i

```
Set th_cor_inter, th_cor_intra, th_up, th_low, n, i = 1, R_m, ρ_te, ρ_hute, RN
Compute ρ_te_{S_i S_j} for each neighbour S_j                    # Inter-Nodes phase
if ρ_te > th_cor_inter then
    Compare RN_{S_i} to RN_{S_j}
    if RN_{S_i} > RN_{S_j} then
        S_i continues sending the temperature values to the sink
    end if
end if                                                           # Intra-Node phase
Compute ρ_hute
if ρ_hute > th_cor_intra then
    Compute th_up, th_low and R_m                                # Equations 12,15,16
    Compute the next humidity value                              # Equation 13
end if
```

In the inter-nodes correlation, if the temperature parameters of S1 and S2 are correlated, one out of both sensors will not send its temperature values anymore and the sink considers the sent one as the data for both nodes (one real and one predicted). As mentioned before, the residual energy is the parameter taken into account between two neighbour nodes to decide which node will send the data. In the intra-node process, if the humidity is correlated to the temperature, each node will send one of the two values (e.g, temperature) to the sink with all the needed parameters (R_m,ρ,..) to predict the other value. However, the two levels of correlation can be related, while taking account of the predicted temperature from the inter-nodes correlation in the intra-node

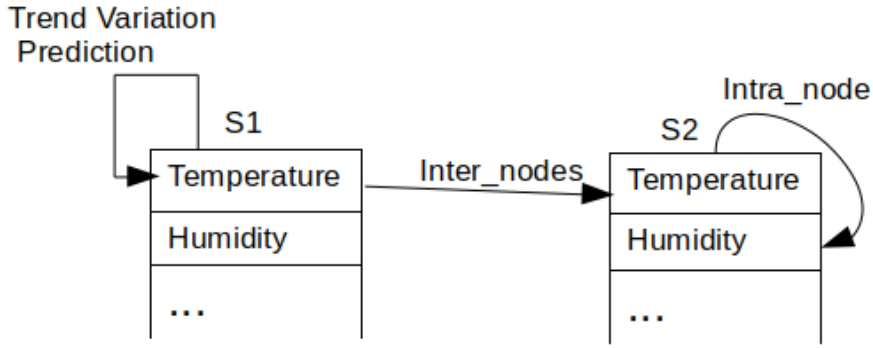

Fig. 4 The triple prediction process

level. In this case the humidity value can be predicted through the predicted value of the temperature which will reduce furthermore the amount of sent data to the sink. This data reduction technique can be extended to study n parameters on m different nodes for the inter-nodes and intra-node methods. To enhance the data reduction, combining both algorithms (MLDR and PDCP) provides us with a triple prediction process as shown in Fig 4 based on the scenario presented in Fig 2.

In this figure, S1 and S2 are two neighbour nodes, S1 is predicting its own temperature values. Temperatures on both nodes are correlated using the inter-nodes technique (Algo.2) and so the temperature on S2 is predicted based on S1 already predicted temperature. Then, the humidity and the temperature on S2 are correlated using the intra-node method (Algo. 2). In this case, the humidity on S2 is predicted based on the predicted value of temperature on S2.

7 Experimental Results

In this section, we compare MLDR to AS_TR from [5] since the authors in [5] compute the trend of the parameters to predict the next values using different techniques. The couple MLDR and PDCP algorithms are compared to a Bayesian approach in [12]. The authors in [12], predict the humidity from the temperature values. In our experiments we also predicted the humidity based on the temperature using different techniques. This comparison is based on the prediction accuracy and the amount of sent data from the nodes to the sink. We used a MATLAB simulator with a temperature data set for the 8th, 9th and the 10th of April 2020 from two sensor nodes deployed in Lille city, France (Lille airport and Lille city centre) from Weather_Underground website which gathers data from a sensor network of different weather stations

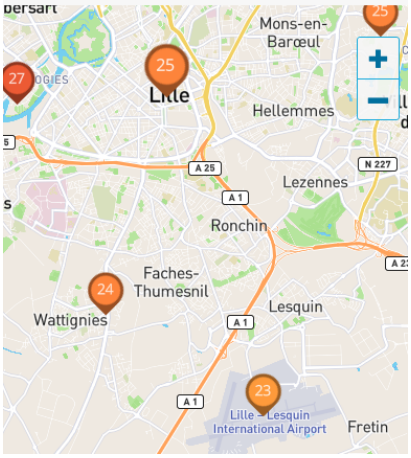

Fig. 5 Nodes real location

deployed around the globe[1]. In the remaining of the simulations we consider S1 as the airport node (Lesquin) and S2 as the city centre node as shown in Fig. 5. The sampling rate of the sensor node is set to 1 value each 30min. The temperature in those 3 days varied from 9 degrees Celsius as a minimum to 26 degrees Celsius as a maximum. 144 samples of each feature are measured at the node. The main goal is to reduce this amount of data sent from the sink while maintaining the information integrity. Table 1 presents the different values of the experimental parameters used in MLDR and PDCP algorithms.

Table 1 Experimental parameters

Parameter	MLDR	PDCP-inter	PDCP-intra
th_{tr}	1	-	-
th_{cr}	2	3	-
th_{hum}	-	-	10%
th_{cor}	-	0.9	0.75

7.1 MLDR

In those experiments, for a better usage of MLDR algorithm, the thresholds are set as th_{tr}=1 degree and th_{cr}=2 degrees. While applying the MLDR algorithm, the number of sent values is decreased from 144 to 25 and 27 respectively. Comparing the data reduction between MLDR and the approach proposed in [5], each node using MLDR sends only the critical data when it occurs.

[1] https://www.wunderground.com

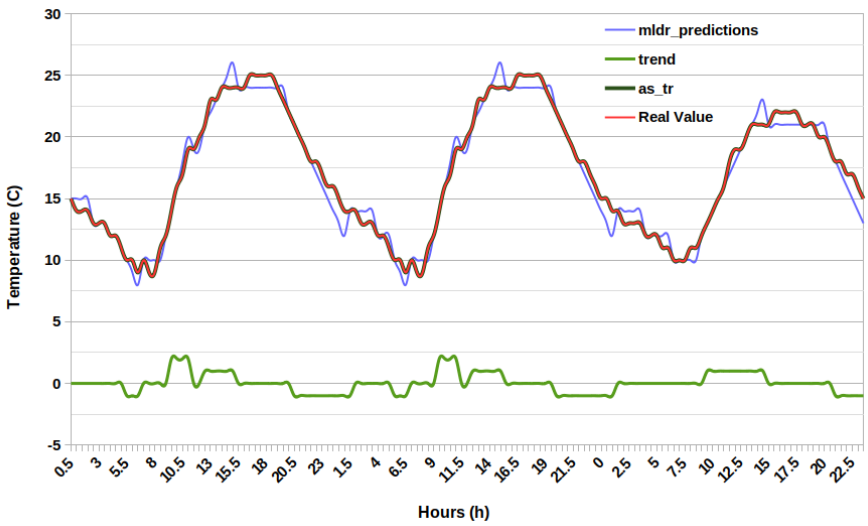

Fig. 6 MLDR Predictions Behaviour

The node waits until the end of the learning phase to sent the reference value and the trend, but not all the captured data in the learning phase. In Fig. 6 and Table 2, we draw a comparison between MLDR approach and the data reduction part in [5] using the same parameters.

The number of predicted values and their integrity for the 8th, 9th and the 10th of April are shown in Fig. 6. The maximum difference between the predicted values in MLDR and the real values varies between 0 and 2 degrees Celsius. The trend was 0 for several periods because of the stability of the temperature in different parts of the day. In other periods, there were no stability in the change so the trend could not be detected, the learning phase must run again to detect a stability of the change.

Table 2 Amount of transmitted data per day

Day	All data	MLDR	AS+TR[5]
8th	48	10	22
9th	48	12	33
10th	48	8	21
Total	144	30	76
Data Reduction	0%	79%	47%

AS+TR approach [5] has a high level of data prediction accuracy, however, as shown in Table 2, it sends three times more data than MLDR which still maintains a high accuracy as shown in Fig. 6, the prediction is always less than 2 degrees of difference from the real value. In [5], the node enters the learning

phase upon any change and the trend is computed after only 1 difference. This process leads to get directly real data from the sensor node because of the difference between the real value and the predicted value based on the changeable trend, which explains the big amount of sent data from the node to the sink as shown in Table 2.

7.2 Data Correlation

In this section, we start by experimenting PDCP algorithm and comparing it to a Bayesian approach in [12]. Later on in the next section a comparison of the double MLDR and PDCP algorithm with the same Bayesian approach takes place. For the inter-nodes correlation the temperature parameter was taken into account as the studied parameter between both sensors. For the intra-node correlation the temperature and humidity parameters in each sensor were selected to evaluate their correlation and test our approach. The sampling rate of the sensor node is set to 1 value each 30 minutes (by default). After conducting several simulations, the number of needed values to compute the correlation coefficient between two parameters is set to 48 values gathered in a whole day. The 8th of April is used for learning the values and thresholds. These parameters are used in the testing phase in the next two days. The temperature in those 3 days varied from 9 degrees Celsius as a minimum to 26 degrees Celsius as a maximum. The humidity varied from 30% to 100%.

7.2.1 Inter-Nodes

In this part, for the inter-nodes correlation, the temperature parameters from the two neighbour sensor nodes S1 and S2 is studied. The threshold of correlation th_{cor} is set to 0.9, since in this part we need a very high correlation to decide to stop sending one of the two values. In those 3 days, 144 values from each parameter are sensed at each sensor-node. The main purpose is to reduce the amount of sent data to the sink.

The simulations show that the correlation coefficient for the whole first day (April 8 2020) is equal to 0.91, greater than the predefined threshold of correlation (0.9) for the inter-nodes correlation, the mean ratio R_m is equal to 1.09.

This high correlation coefficient leads to send only one of the two temperature values to the sink (Value of S2) alongside the R_m value at the very start of the network. The sink computes the other value (Value of S1) as shown in equations 12 and 13. Fig 7 shows the difference between the real and the predicted values for the next 2 days (April 9 and 10). This difference does not surpass 1 degree Celsius.

While applying the inter-nodes correlation part of PDCP algorithm, the number of sent values is decreased from 192 values for the two nodes in two days to 96 values as shown in Table 3 (S1 is not sending the temperature values).

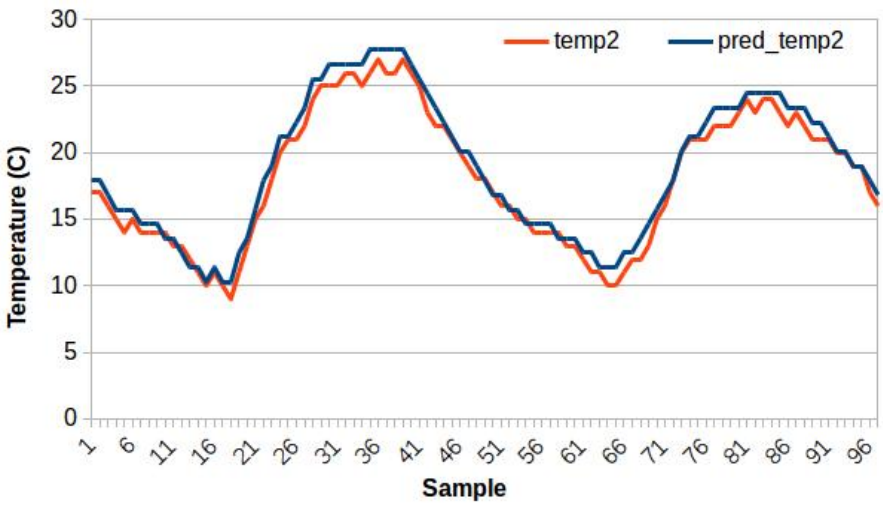

Fig. 7 Temperature prediction on the sink

Table 3 Amount of transmitted temperature values per day by S1 and S2

Day	All data	Inter-Nodes
9th	96	48
10th	96	48
Total	192	96
Data Reduction	0%	50%

7.2.2 Intra-Node

Different parameters are sensed by each sensor node. After conducting several simulations, we decided to study the correlation between the temperature and the humidity parameters on S1 (Airport Node) while applying the intra-node technique in the PDCP algorithm based on the Pearson Correlation coefficient method. The threshold of correlation th_{cor} for the intra-node part is equal to 0.75, since we are comparing different types of parameters. On April 8, 2020 the correlation coefficient was equal to -0.8, which is greater than 0.75, so the prediction process can take place based on the last value and the computed mean ratio R_m. In this case, $R_m = 4$. The humidity values are then predicted through the temperature values as explained above. The critical threshold for humidity is set to th_{hum}=10%. If the difference between the predicted value and the captured value exceeds this threshold, the real humidity value is sent to the sink and the prediction process continues taking this new value into account.

Fig 8 shows the prediction of the humidity for the same sensor node on the sink for April 9 and 10, 2020. In those 2 days, 96 humidity values were captured, however, the node sent 6 humidity values to the sink (surpassing the

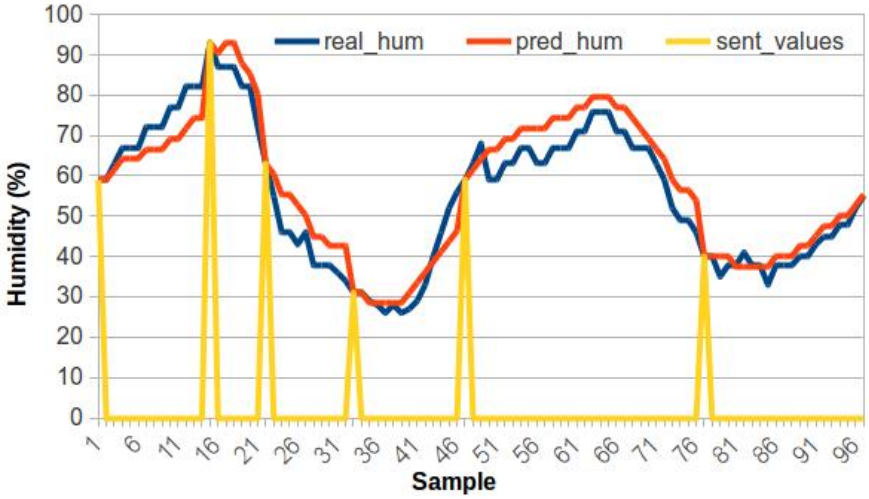

Fig. 8 Humidity prediction at the sink while applying the intra-node technique

thresholds) to enhance the prediction at the sink. Their were no consecutive sent values to recompute the correlation coefficient ρ and the mean ratio R_m. The difference between the real and the predicted values did not surpass 7% of humidity which affirms the integrity and the feasibility of the predictions in our approach.

Table 4 Amount of transmitted humidity values per day by S1

Day	All data	Inta-Node
9th	48	4
10th	48	2
Total	96	0
Data Reduction	0%	93%

Tables 3 and 4 show that the intra-node part in PDCP algorithm when applied reduces the amount of sent data to the sink while maintaining the integrity and the accuracy of the data as shown in the figures above. A comparison with another method is drawn in the section below.

7.2.3 Intra-Node and Inter-Nodes Combination

In the inter-nodes part, the temperature value of S1 is predicted on the sink. In the intra-node part, the humidity of S1 is predicted based on the temperature on the same node. However, combining both parts of PDCP algorithm increases the data reduction. This combination helps to predict at the sink the humidity values for S1 based on the already predicted temperature values

for the same node. Fig 9 shows the humidity prediction for S1, 7 values were sent (the threshold was surpassed). The maximum difference between the real humidity value and the prediction value did not exceed 7% of humidity, thus staying in the same humidity category. While applying the intra-node correlation on S2, the node sent only 7 humidity values to the sink in two days of predictions as shown in Table 5.

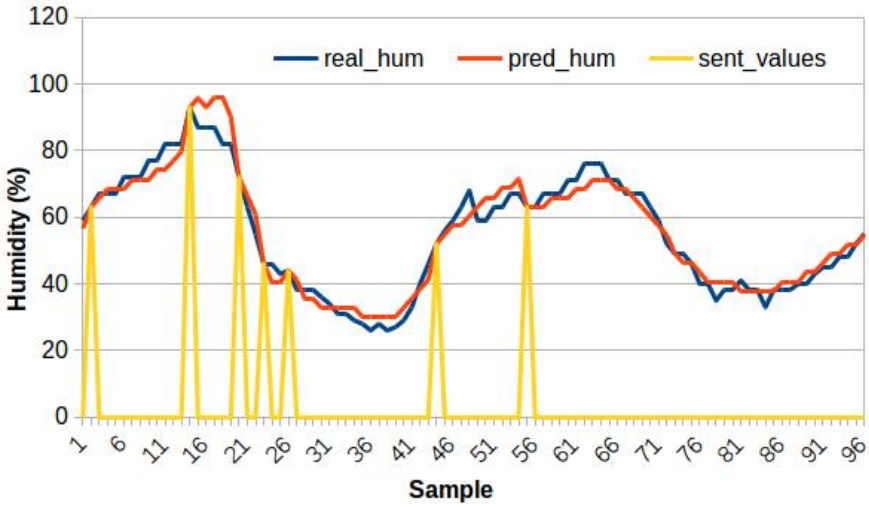

Fig. 9 Humidity prediction at the sink while applying PDCP algorithm

Table 5 draws the differences between our approach and a Bayesian inference approach from the literature [12] for the same scenario and parameters. As noticed from the numbers, they are neglecting any change in the humidity values which helps them to improve data reduction to 50% for the intra-node correlation but they lost some accuracy with a humidity standard deviation H_{SD} up to 10%. However, the inter-nodes correlation applied in PDCP gives us the edge to improve the percentage of data reduction to reach 69% with a better accuracy and a standard deviation H_{SD} of 7%.

Table 5 Amount of transmitted values per day by S1 and S2

Day	All data	PDCP	Bayesian[12]
$S1_T$	48	0	48
$S1_H$	48	7	0
$S2_T$	48	48	48
$S2_H$	48	5	0
Total	192	60	96
H_{SD}	0%	7%	10%
Data Reduction	0%	69%	50%

7.3 PDCP and MLDR combination

Combining both algorithms is like having a three prediction process. We take S1 as an example, MLDR algorithm helps the sink predict the next value of the temperature on S1. Based on this value, the temperature on S2 is predicted, from which the prediction of the humidity value on S2 takes place.

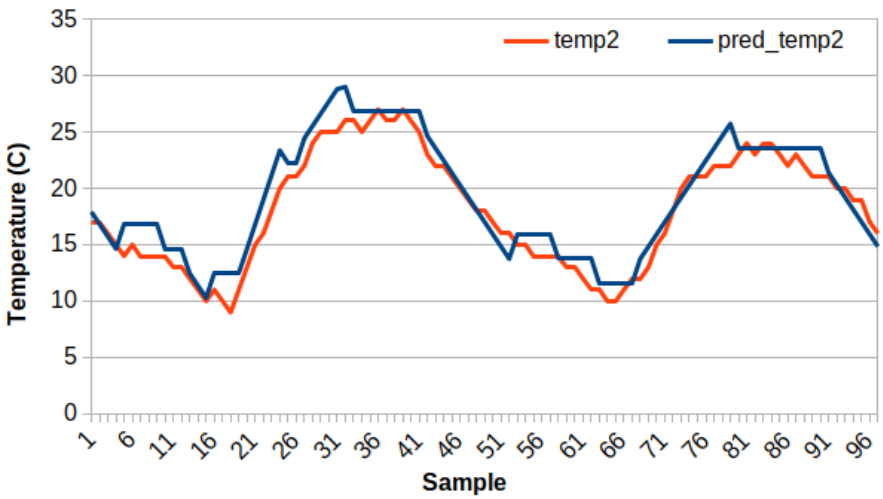

Fig. 10 Temperature prediction on the sink for S2 while applying PDCP and MLDR algorithms

Fig. 10 shows the prediction of the temperature value for S2 on the sink based on the predicted value of the temperature value on S1. The standard deviation does not surpass 3 degrees Celsius and is between 0 and 1 for more than 90% of the time, which keeps the high accuracy.

The triple prediction is shown in Fig. 11. In the continuity of Fig. 10, the sink predicts the humidity on S2 based on all the previous predicted values. The standard deviation is equal to 8% of humidity which is acceptable for the accuracy (keeping the humidity predicted value in the same range as the real values).

Table 6 compares the data reduction percentage and the standard deviation of the triple prediction process (MLDR + PDCP) and the approach proposed in [12]. Data reduction in our approach is way more important than in the Bayesian approach proposed in [12], having 88% of reduced data instead of 50%. For the temperature prediction as mentioned before the standard deviation does not surpass 3 degrees Celsius which is not the case in [12] since they do not predict the temperature, they send all the values. Considering the humidity, the standard deviation in our approach is about 7% of humidity which is better than the 10% in [12].

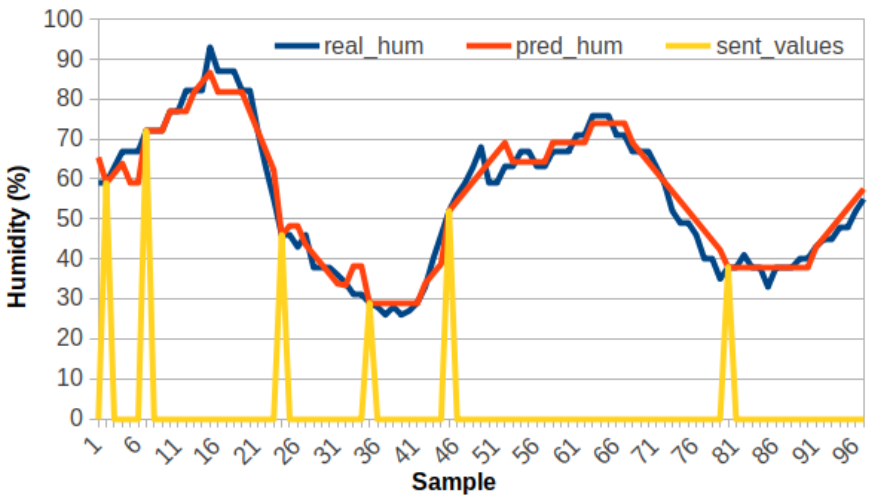

Fig. 11 Humidity prediction at the sink while applying PDCP and MLDR algorithms

Table 6 Amount of transmitted values per day by S1 and S2 using MLDR and PDCP

Day	All data	MLDR+PDCP	Bayesian[12]
$S1_T$	48	0	48
$S1_H$	48	6	0
$S2_T$	48	12	48
$S2_H$	48	5	0
Total	192	23	96
H_{SD}	0%	7%	10%
T_{SD}	0	3	0
Data Reduction	0%	88%	50%

The size of a sent packet is equal to 4 KB. To compare the energy consumption of the transmission, we assume that each sensor node is equipped with a CC2420 radio transceiver and an ARM7TDMI microprocessor as in [10]. According to their data-sheets, to send a byte, the node consumes $18.37 \times 10^{-7} J$. In this case, the nodes in our approach for the 2 days of monitoring consumes $0.173J$ compared to $0.722J$ if the algorithm in [12] is implemented and $1.44J$ if no algorithm is implemented. Reading those numbers, our approach consumes 4 times less energy than the technique in [12] and 8 times overall if no algorithm is implemented.

8 Conclusion and Future Work

In this paper, we proposed a data reduction technique based on a data correlation technique by applying the Pearson correlation coefficient functions and equations in WSN implemented for agriculture to detect any abnormal situation in the meteorological data.

Our simulations show a reduction of more than 88% of the overall transmitted data considering the temperature and the humidity of two sensor nodes. This reduction percentage surpasses other approaches from the literature by more than 35%. Not to forget that our approach does not lose data since all the values are predicted on the sink and the node levels. The data accuracy is maintained by good margins of standard deviations (3 degrees for the temperature and 7% for the humidity).

In the near future, this approach can be enhanced by applying correlation on more than two nodes at the same time. Other ideas can be taken into account such as studying the routing protocol for multi-hop scenarios. To further increase the data reduction, sampling rate adaptation is to be considered. It enables us to to reduce the amount of data sensed and processed by the sensor node, especially when the values are stable and not changing. Thus, we can reduce the energy consumption related to those two processes and not only the transmission one.

Acknowledgements

This work was partially supported by a grant from CPER DATA and by LIRIMA Agrinet project.

References

1. Tamoghna Ojha, Sudip Misra, and Narendra Singh Raghuwanshi. Wireless sensor networks for agriculture: The state-of-the-art in practice and future challenges. *Computers and Electronics in Agriculture*, 118:66 – 84, 2015.
2. Soledad Escolar Díaz, Jesús Carretero Pérez, Alejandro Calderón Mateos, Maria-Cristina Marinescu, and Borja Bergua Guerra. A novel methodology for the monitoring of the agricultural production process based on wireless sensor networks. *Computers and Electronics in Agriculture*, 76(2):252 – 265, 2011.
3. Kizito Patrick Musaazi, Tonny Bulega, and Stephen Mutaawe Lubega. Energy efficient data caching in wireless sensor networks: A case of precision agriculture. In Amos Nungu, Bjorn Pehrson, and Julianne Sansa-Otim, editors, *e-Infrastructure and e-Services for Developing Countries*, 2015.
4. Christian Salim and Nathalie Mitton. Machine learning based data reduction in wsn for smart agriculture. In *Advanced Information Networking and Applications*, Cham, 2020.
5. Gaby Bou Tayeh, Abdallah Makhoul, David Laiymani, and Jacques Demerjian. A distributed real-time data prediction and adaptive sensing approach for wireless sensor networks. *Pervasive and Mobile Computing*, 49:62 – 75, 2018.
6. Maria Bermudez-Edo, Payam Barnaghi, and Klaus Moessner. Analysing real world data streams with spatio-temporal correlations: Entropy vs. pearson correlation. *Automation in Construction*, 88:87 – 100, 2018.
7. S. Radhika and P. Rangarajan. On improving the lifespan of wireless sensor networks with fuzzy based clustering and machine learning based data reduction. *Applied Soft Computing*, 83, 2019.
8. C. Habib, A. Makhoul, R. Darazi, and C. Salim. Self-adaptive data collection and fusion for health monitoring based on body sensor networks. *IEEE Transactions on Industrial Informatics*, 12(6):2342–2352, Dec 2016.

9. V. Toldov, L. Clavier, V. Loscrí, and N. Mitton. A thompson sampling approach to channel exploration-exploitation problem in multihop cognitive radio networks. In *IEEE Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Sep. 2016.
10. Christian Salim, Abdallah Makhoul, Rony Darazi, and Raphaël Couturier. Similarity based image selection with frame rate adaptation and local event detection in wireless video sensor networks. *Multimedia Tools and Applications*, 78(5):5941–5967, Mar 2019.
11. Gaby Bou Tayeh, Abdallah Makhoul, Charith Perera, and Jacques Demerjian. A spatial-temporal correlation approach for data reduction in cluster-based sensor networks, 2019.
12. Cristanel Razafimandimby, Valeria Loscri, Anna Maria Vegni, Driss Aourir, and Alessandro Neri. A Bayesian approach for an efficient data reduction in IoT. In *Int. Conf. on Interoperability in IoT (InterIoT)*, November 2017.
13. C. Razafimandimby, V. Loscri, A. M. Vegni, and A. Neri. Efficient bayesian communication approach for smart agriculture applications. In *IEEE Vehicular Technology Conf. (VTC-Fall)*, Sep. 2017.
14. M. Azaza, C. Tanougast, E. Fabrizio, and A. Mami. Smart greenhouse fuzzy logic based control system enhanced with wireless data monitoring. *ISA Transactions*, 61:297 – 307, 2016.
15. A. Ghaddar, T. Razafindralambo, I. Simplot-Ryl, S. Tawbi, and A. Hijazi. Algorithm for data similarity measurements to reduce data redundancy in wireless sensor networks. In *A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, June 2010.
16. Mou Wu, Liansheng Tan, and Naixue Xiong. Data prediction, compression, and recovery in clustered wireless sensor networks for environmental monitoring applications. *Information Sciences*, 329:800 – 818, 2016. Special issue on Discovery Science.
17. Aya Ayadi, Oussama Ghorbel, M.S. BenSalah, and Mohamed Abid. Spatio-temporal correlations for damages identification and localization in water pipeline systems based on wsns. *Computer Networks*, 171:107134, 2020.
18. G. Rajesh and Ashvini Chaturvedi. Correlation analysis and statistical characterization of heterogeneous sensor data in environmental sensor networks. *Computer Networks*, 164:106902, 2019.
19. Hongling Zhao, Haibo Yang, and Zongmin Wang. Optimization of rainfall sensor network layout based on the correlation coefficient method. In *2016 International Conference on Artificial Intelligence: Technologies and Applications*. Atlantis Press, 2016/01.
20. Fernando R. Almeida, Angelo Brayner, Joel J. P. C. Rodrigues, and Jose E. Bessa Maia. Improving multidimensional wireless sensor network lifetime using pearson correlation and fractal clustering. *Sensors*, 17(6), 2017.
21. Yi Shen, Jinyun Guo, Xin Liu, Qiaoli Kong, Linxi Guo, and Wang Li. Long-term prediction of polar motion using a combined ssa and arma model. *Journal of Geodesy*, 92(3):333–343, 2018.
22. Feng Ding, Dandan Meng, Jiyang Dai, Qishen Li, Ahmed Alsaedi, and Tasawar Hayat. Least squares based iterative parameter estimation algorithm for stochastic dynamical systems with arma noise using the model equivalence. *International Journal of Control, Automation and Systems*, 16(2):630–639, 2018.
23. Khushboo Jain, Arun Agarwal, and Anoop Kumar. A novel data prediction technique based on correlation for data reduction in sensor networks. In Poonam Bansal, Meena Tushir, Valentina Emilia Balas, and Rajeev Srivastava, editors, *Proceedings of International Conference on Artificial Intelligence and Applications*, pages 595–606, Singapore, 2021. Springer Singapore.
24. Xiangyun Qing and Yugang Niu. Hourly day-ahead solar irradiance prediction using weather forecasts by lstm. *Energy*, 148:461 – 468, 2018.
25. E. Marchi, F. Vesperini, F. Weninger, F. Eyben, S. Squartini, and B. Schuller. Nonlinear prediction with lstm recurrent neural networks for acoustic novelty detection. In *International Joint Conference on Neural Networks (IJCNN)*, July 2015.
26. Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, Cambridge, MA, USA, 2015.
27. T. Shu, J. Chen, V. K. Bhargava, and C. W. de Silva. An energy-efficient dual prediction scheme using lms filter and lstm in wireless sensor networks for environment monitoring. *IEEE Internet of Things Journal*, 6(4):6736–6747, Aug 2019.

28. J. Abdullah, M. K. Hussien, N. A. M. Alduais, M. I. Husni, and A. Jamil. Data reduction algorithms based on computational intelligence for wireless sensor networks applications. In *2019 IEEE 9th Symposium on Computer Applications Industrial Electronics (ISCAIE)*, pages 166–171, April 2019.
29. H. Cheng, Z. Xie, Y. Shi, and N. Xiong. Multi-step data prediction in wireless sensor networks based on one-dimensional cnn and bidirectional lstm. *IEEE Access*, 7:117883–117896, 2019.