Investigating Data Similarity and Estimation Through Spatio-Temporal Correlation to Enhance Energy Efficiency in WSNs

Alia Ghaddar, Tahiry Razafindralambo, Isabelle Simplot-Ryl, David Simplot-Ryl, Samar Tawbi, Abbas Hijazi

Investigating Data Similarity and Estimation through Spatio-Temporal Correlation to enhance Energy Efficiency in WSNs

ALIA GHADDAR[1,2⋆], TAHIRY RAZAFINDRALAMBO[1], ISABELLE SIMPLOT-RYL[1], DAVID SIMPLOT-RYL[1], SAMAR TAWBI[2], ABBAS HIJAZI[2]

[1] *INRIA/CNRS/Univ. Lille 1, France*
[2] *Lebanese university Beirut, Lebanon*

Wireless sensor networks are of energy-constrained nature, which calls for energy efficient protocols as a primary design goal. Thus, minimizing energy consumption is a main challenge. We are concerned in how collected data by sensors, can be processed to increase the relevance of certain mass of data and reduce the overall data traffic. Since sensor nodes are often densely deployed, the data collected by nearby nodes are either redundant or correlated. One of the great challenges for the aforementioned problem is to exploit temporal and spatial correlation among the source nodes. Our work is composed of two main tasks: 1- A *predictive modeling task* that aims to capture the temporal correlation among collected data. 2- A *data similarity detection task* that measures the data similarity based on the spatial correlation.

Key words: Wireless sensor networks; Time series forecasting; Data aggregation; Data similarity; Redundancy; Spatial correlation; Temporal correlation;

1 INTRODUCTION

Wireless Sensor Networks have opened up new opportunities in many domains including environmental monitoring, agriculture, industrial, biological detection (On/In-Body sensor networks), home security and so on.

⋆ email: Alia.Ghaddar@lifl.fr

A main task of sensor networks is the regular collection and aggregation of data towards the base station. participating nodes in these networks are typically battery operated, and thus have access to a limited amount of energy and processing power. One node, called the leader, collects data from surrounding nodes and then sends the summarized information to upstream nodes (many-to-one flows). It happens that some collocated nodes notify the sink about the same event, at almost the same time and approximately the same values. This induces a propagation of redundant highly correlated data, which is costly in terms of system performance, and results in energy depletion, network overloading, and congestion. However, important ideas in last decade arise to lighten the importance of redundancy on data accuracy and sensing reliability in WSNs. Therefore, methodologies to decrease or even eliminate the redundancy are often needed to make a balance between the benefits and disadvantages while maintaining the system lifetime in WSNs. For this purpose, clustering and data aggregation approaches has been extensively studied [1], since the energy consumption of the network can be minimized if the amount of data that needs to be transmitted is minimized.

The idea behind data aggregation is to combine the data coming from different sensor nodes en route, eliminating redundancy, minimizing the number of transmissions and thus saving energy [21]. Many researchers have noted the importance of data aggregation in sensor networks such as [28]. Basically, there are two types of data aggregation techniques: Spatial data aggregation which aggregates data from different sources and temporal data aggregation which combines data from different periods of time.

Along with this, the performance of data collection and aggregation in WSNs can be enhanced by exploiting the data correlations . Sensor nodes are often densely deployed in sensor network[2], hence the data collected by nearby sensor nodes are either redundant or correlated. This data correlation can be exploited to reduce the amount of data transmitted in the network, resulting in energy savings.

Spatial Correlation occurs when observations from the sensor nodes which are in close proximity are highly correlated (the degree of correlation depends upon the distance between nodes). Therefore, information about an event is captured by many surrounding sensor nodes, which generate a large amount of traffic on the wireless channel and consumes a lot of battery energy. Furthermore, the nature of the physical phenomenon constitutes the Temporal Correlation between each consecutive observation of a sensor node.

In this paper, we propose an estimation-based algorithm to investigate temporal and spatial correlations among data in WSNs to detect redundancy and

reduce data transmission traffic. Our work is composed of two main tasks: A *predictive modeling task* and a *data similarity detection task*. The goal of a predicting modeling is to build a model that can be used to predict− based on known examples collected in the past, future values of some phenomena which will reduce the transmission rate from a sensor node to the base station. This model aims to capture the temporal correlation among collected data. While the data similarity detection task (relayed on kernel based methods) measures the similarity between collected data in order to improve the performance of data collection while preserving data accuracy. The use of these two tasks produces a considerable accuracy prediction and transmission rate reduction as we will show later in this paper.

The rest of the paper is structured as follows: Section 2 provides an overview of time series techniques, describes our prediction model and presents its efficiency in terms of communication traffic and rate. In Section 3, we present key concepts on similarity functions. We also propose data similarity detection algorithm and show its efficiency. Section 5 concludes our paper.

2 FORECASTING TECHNIQUES

To improve the performance of data aggregation, Times series forecasting was proposed as a means to reduce the amount of communication between the wireless sensor and the sink. The sink node exploits time series model to predict local readings instead of direct communication with sensors.

Some approaches [27, 19] use AR/ARMA models (AutoRegressive /AutoRegressiveMoving Average) [4] contained in both the sink and each sensor. Other approaches were based on Kalman filters [20]. Some have use relatively complex probabilistic models (e.g., multi-variate Gaussians [10] or generalized graphical models). Other works were more simpler like [27] in which the framework relies mostly on local probabilistic models computed and maintained at each sensor. This model is similar in character to our prediction model, in that each sensor continuously maintains its local model, and notifies the sink only of significant changes. However, it has more heavyweight learning phase than our model as we mention later. Recent work in [19], gives nodes, additional task over environmental monitoring. Every node, has to calculate an Adaptive-ARMA model (A-ARMA) from a history of samples to discover the time series correlation between measurements. Although it uses ARMA, the work proposed in [19], is close to ours since it uses recent readings to predict future local readings. When a reading is not properly predicted by the model, the sensor choose to re-learn the model, and notify the sink by sending new model parameters and a certain number of

recent readings samples. Predictions would then begin again, and continue, until an error tolerance is violated.

2.1 Modeling time series

A time series is a set of observations X_t, each of which is recorded at time t, representing a phenomena evolving over time. An important part of the analysis of time series is the description of a suitable uncertainty data model to predict future values using some recent history of readings. AR/ARMA models are principal models for time series. Being more simple than ARMA model due to its lower computational cost and memory requirements, AR model becomes popular in many domains (such as in finance, communication, weather forecasting, and a variety of other domains [4]). The time series (AR) of order k is represented as follows:

$$AR(k) : X_{i+k} = a_0 + a_1 X_{i+k-1} + \cdots + a_k X_i \quad (1)$$

where a_1,...,a_k are the model parameters, X_t is the time series. The work in [27] shows that AR models, while simple, still offer excellent accuracy in sensor networks for monitoring applications, making such models practical on many current-generation sensor networks. We will adopt this model due to its simplicity, which leads to lower computational cost and memory requirements (unlike the fully general ARMA models).

2.2 Our Estimation approach

Motivation

Our prediction model design is motivated by the need to reduce communication overhead between sensors communicating on one hop path, in order to increase sensor lifetime and the monitoring operations. Since sensor nodes are energy-constrained and they are difficult to replace every time when consumed: such as implantable body sensors (pacemakers and cardioverter-defibrillators), disaster or battle field monitoring sensors, etc. Moreover, since it was indicated by empirical studies [13], that the transmission of one bit over 100 meters would cost about the same level of energy as executing 3000 instructions in sensor node. We try to achieve our goals using prediction model without loosing accuracy by exploring temporal correlation among data.

Prediction Model Overview

Basically, the communication between a sensor and a sink is caused by an error threshold violation. Regularly, when a sensor collects a new observation, it computes the error value e between this new observation and the predicted

value from the model. If the prediction error becomes bigger than some pre-specified error tolerance, the prediction is not acceptable; Then sensor node re-computes the parameters of the model from a number of recent samples, and sends information about the changes to the sink as notification to update its model. In most cases, the information sent to the sink are the model parameters and optionally a list of measures such as in [27] and [19]. In [15], we proposed two algorithms with a pre-specified error bound. We reduced the communication overhead by sending only one previous measure (Algorithm 1 denoted by EEE) then a list of recent errors (Algorithm 2 denoted by EEE^+) for more accuracy and energy-saving. We assume having a normal data series with no anomalous values and we demonstrated the efficiency of EEE^+ algorithm in terms of communication traffic and energy consumption.

Preliminaries

In this part, we rely on cluster-based data aggregation mechanisms in which each node can reach its corresponding sink (e.g Cluster Head) directly in one-hop (such as In/On-Body sensors for personal health monitoring,..) and we assume having a proper routing protocol.

We employ an AR-based model for data prediction. The model is contained in both: the sink and each source sensor node, to predict values instead of direct communication. As [26], we ignore the trend and seasonal components of the time series and we set a narrow prediction window, denoted by $k = 3$ in order to decrease the complexity of learning and adapting the model.

The choice of a narrow prediction window is made for two reasons: (1) to simplify our model similarly to [26, 27], (2) Our experiment results depicted in Figure 1 show that a large prediction window does not really increase the percentage of accepted values. In fact, a too large window means that we take into account very old information that may have changed, so the algorithm reaction to change may be too slow. This could explain the deviation in number of accepted values when the window size increases. This could require more re-learning phases. Moreover, Figure 2 shows that the percentage error is limited ($\sim$5%) while changing the prediction window size for this data type. So typically, we consider a value of 3 a reasonable compromise.

2.3 Predictive modeling task-EEE^* algorithm

The prediction task is based on the work presented in [15]. Regularly, when a sensor collects a new observation N_t at time t, it computes the error value e between this new observation and the predicted value X_t from the model. If the prediction error becomes bigger than some pre-specified error tolerance th (X_t is mispredicted value), the sensor *re-learns* the prediction model

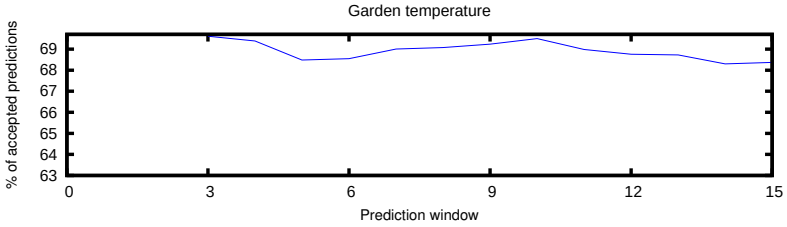


FIGURE 1
Influence of the prediction window size on the estimation precision using EEE^{+} and $th_{err} = 0.05$.

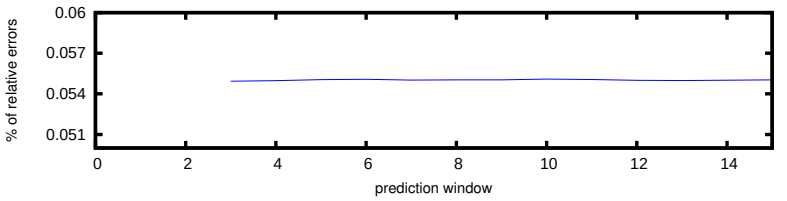


FIGURE 2
Influence of the prediction window size on the estimation error using EEE^{+} and $th_{err} = 0.05$.

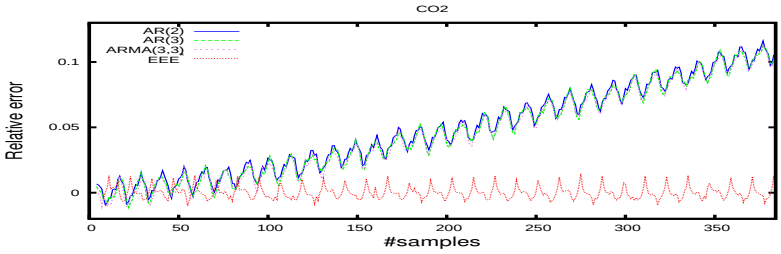
from a number of recent samples, and sends information about the changes to the sink as notification to update its model. It also generates a new error prediction bound as follows:

$$th = \sum_{i=1}^{k} (\frac{|N_i - N_{i-1}|}{k}) + r_{rand} \tag{2}$$

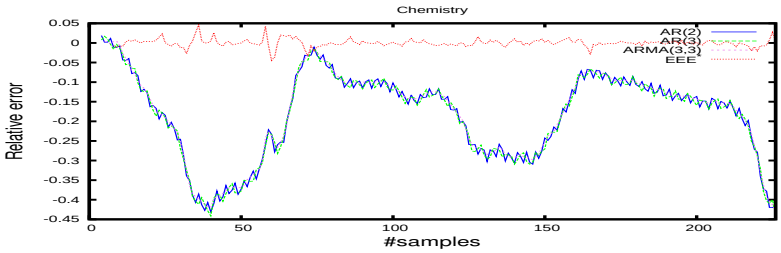
r_{rand} is random value $\in [\frac{-c\sigma}{\sqrt{k}}, \frac{c\sigma}{\sqrt{k}}]$ to add some reliability the choice of th. We denote by σ the standard deviation of the differences between the latest data samples N_i. c indicates the level of uncertainty (for a confidence interval of 95% we choose $c = 1.96$). The prediction would then begin again and continue until the new error prediction bound is violated.

2.4 Our prediction model Efficiency

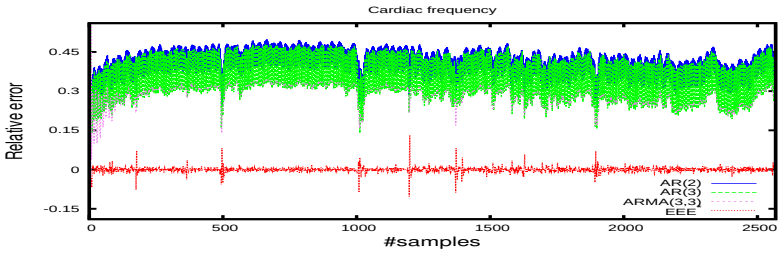
Performance of the models is statistically evaluated in a panel of different tables such as Data point statistics (RMSE, correlation coefficient,..), Relative error statistics (Mean value of the relative error,..) and others. To evaluate the performance in estimation of our model and AR/ARMA models, we use the



(a) CO_2



(b) $Chemical\ process\ T°$



(c) $Cardiac\ Frequency$

FIGURE 3
Relative error produced by EEE^* and different other models such as AR(2), AR(3) and ARMA(3,3)

relative error measure between the real values and the estimated ones, the residual values and the correlation coefficient.

Prediction Accuracy

Information about the absolute error is little use in the absence of knowledge about the magnitude of the quantity to be measured. So, in order to determine if the estimation error produced is too high or accepted according to the dynamic threshold change, we choose to use the relative error between the estimated and the real values, as follows: $\frac{N_i - X_i}{N_i}$. Figure 3, shows the relative error obtained using EEE^* on real data series such as: the chemical process temperature readings taken every minute, the monthly measurements of carbon dioxide above Mauna Loa* , and cardiac frequency measurements† .

Data type	AR(2)	AR(3)	ARMA(3,3)	EEE^*
Garden (T°)	up to 43%	up to 50%	Divergence	up to 7%
Chemical process (T°)	up to 40%	up to 43%	up to 41%	up to 6%
Carbon dioxide	up to 10%	up to 11%	up to 11%	up to 2%
Cardiac frequency	up to 50%	up to 50%	up to 35%	99% of errors were $\leq$ 5%
Radiosity	up to 60%	up to 60%	up to 60%	99% of errors were $\leq$ 8%
Female (T°)	$\sim$5%	$\sim$5%	$\sim$4%	$\sim$3%

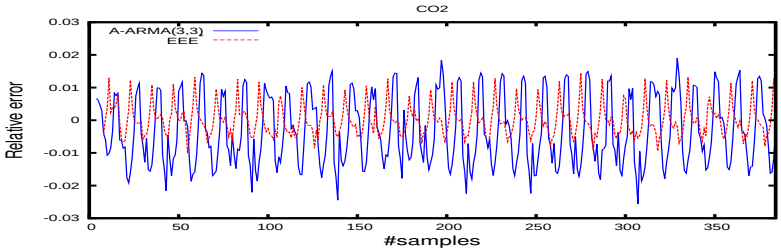
TABLE 1
The relative error produced by each model estimations: AR(2), AR(3), ARMA(3,3) and EEE^*

Figure 3 shows the relative error produced by EEE^* and different other prediction models such as AR(2), AR(3), ARMA(3,3). In fact, predictions generated by AR/ARMA models using different estimation methods such as MLE, Burg, OLS and Yule-Walker, did not produce better relative errors compared to our prediction model. AR/ARMA models tend to be divergent in some cases such as Figures 3(a). This is due to the effects of looseness of accuracy along the prediction process.

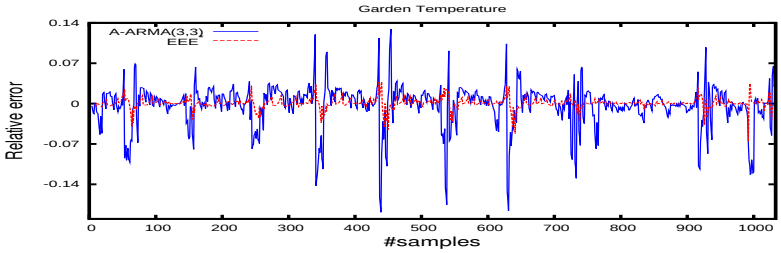
As shown in Table 1, the relative error produced by EEE^* did not exceed 8% for most data types used. In cases such as cardiac frequency, we can notice that 99% of errors were $\leq$ 5%, and that about 12 of 2565 predictions was increased up to 15%. We consider that this could be perturbed by different factors (sudden emotions or actions, etc..). Note that in this paper, we do not detect or take into consideration such events or even data intrusions that may occur in other WSNs applications.

Figure 4, shows the relative error produced by Adaptive-ARMA(3,3) pro-

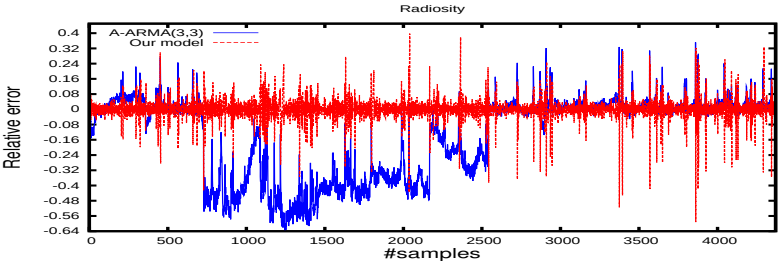
* http://www.robjhyndman.com/TSDL/
† http://www.inrialpes.fr/Xtremlog/



(a) CO_2



(b) $Garden$



(c) $Radiosity$

FIGURE 4
Relative error produced by Adaptive-ARMA(3,3) with a bound=0.05 and EEE^*.

posed in [19] and EEE^* model on the different data types. For the sake of fairness, we used A-ARMA(3,3) with a bound=0.05. We can notice that A-ARMA(3,3) yields the predictions better then classical AR/ARMA models, due to its model adjustment during the prediction process. However, it produces less accuracy then EEE^*. Hence, as for the precision and data transmission traffic, we deduce that EEE^* is an appropriate algorithm for

	Garden Temperature	Chemical process (T°)	Carbon dioxide	Cardiac Freq.
AR(2)	∼1.956 e-08	∼1.586e-06	5.58e-06	3.343e-06
AR(3)	∼0.01523	∼3.363e-05	6.90e-07	2.752e-08
ARMA(3,3)	0.00439	0.018	0.00012	0.00012
EEE^*	0.996	0.9889	0.983	0.964

TABLE 2
RV value measuring the correlation between real values and predictions of AR(2), AR(3), ARMA(3,3) and EEE^*

slow variation data series measurements. We can notice that EEE^* does not provide greatest estimation quality for the radioactivity, as shown in Figure 4(c). However, we note that $\sim 90\%$ of relative errors produced for the radioactivity were around 8%.

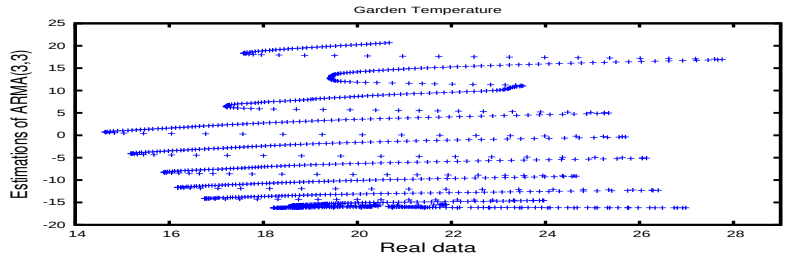
RV coefficient

We also refer to the correlation coefficient to observe the relationship between values. The purpose of this is to see how much the real data and the samples produced by the discussed models are correlated. In other word, if two variables are correlated, we can predict one based on the other. We can notice in Table 2 that the correlation coefficient tends to 1 in EEE^* and is greater then other models correlation coefficient values. This indicates a strong linear relationship between the value produced by our model and the real data. Combining these results with the one produced in Table 3, we consider that our model is a good prediction model and produces more accurate estimations then AR/ARMA models.

Scatterplots

Having a correlation coefficient value that tends to zero (such as for AR/ARMA in Table 2), does not mean the absence of relation. There may be a non linear relation between the models outputs and the real data.

The possibility of such non-linear relationships is another reason why examining $scatterplots$ is a necessary step in evaluating every correlation. Figure 5 shows the relationships between the ARMA(3,3) and EEE^* models and $CO2$ data values. It indicates that EEE^* is a suitable algorithm to predict data studied here then ARMA(3,3) model. As the scatterplots for other models revel similar results using the different data types studied, we did not show them.



(a) $Garden$: $ARMA(3,3)$

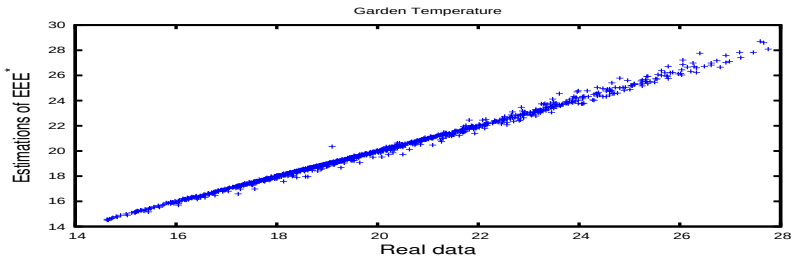


FIGURE 5
Scatterplots for some data types. (b) $Garden$: EEE^*

3 DATA SIMILARITY MEASUREMENTS

3.1 Role of similarity measurements

Similarity measures play a central role in reasoning in many applications such as bioinformatics, natural language processing (NLP), image processing, pattern recognition and different other problems of information retrieval. The similarity functions (called also affinity functions) and denoted by $s : X \times X \longrightarrow R$ are in some sense the converse to dissimilarity functions: meaning that the similarity between two objects should grow if their dissimilarity decreases. In particular, a similarity function is supposed to increase the more similar the points are. Different methods was introduced to mesure the similarity between two objects, we present below, a brief introduction to a special type of similarity functions: the "Kernel functions".

3.2 Kernel based-methods

Kernel functions are one of the most popular tools in Machine Learning and this has by now reached full maturity as evinced by the number of publication and books related to it. Kernel-based learning algorithms [8] work by embedding the data points into a Hilbert space, and searching for linear relations in

such a space. The embedding is performed implicitly, by specifying the inner product between each pair of points rather than by giving their coordinates explicitly. This approach has several advantages, the most important deriving from the fact that often the inner product in the embedding space can be computed much more easily than the coordinates of the points themselves.

Given an input set X, an embedding space F and a map $\phi : X \rightarrow F$. Having two points, $x_i \in X$ and $x_j \in X$, the function that returns the inner product between their images in the space F is known as the $kernel function$.

Definition 3.1 *: A kernel k is a function, such that k(x, z) =* $\prec \phi$ *(x),* ϕ *(z)*$\succ$ *for all x, z* $\in X$*, where* ϕ *is a mapping from* X *to an (inner product) feature space* F*.*

Kernel methods can handle different problems of classification [3, 23, 12], data compatibility, data integration, and data completion. They are based on measures of similarity (kernel functions) that allow us to perform classification, regression and related tasks (for a complete introduction refer to [3]). In-fact, many generic kernels (e.g. Gaussian kernels), as well as specific kernels (e.g. Fisher kernels), describe different notions on similarity of objects.

The gaussian kernel (3) is a popular and powerful kernel used in pattern recognition. Theoretical statistical properties of this kernel can be employed for different techniques such as fuzzy aggregation techniques [11].

$$k(x, y) = \exp \frac{- \parallel x - y \parallel^2}{2 * \sigma^2} \tag{3}$$

where $k(x, y) \in [0, 1]$, σ determines the width of the Gaussian kernel (Note that we have $x = y$ when $k(x, y) = 1$). In what follows, we will adopt the gaussian kernel function in (3) with $\sigma = 1.74$ such as [9].

3.3 Data Aggregation scheme

Achieving energy efficiency to prolong the network lifetime is an important design criterion for Wireless Sensor Networks. Since communication between nodes is the main source of energy consumption [25], different techniques have been used such as *Data Aggregation* to reduce the communication cost There are a large number of existing mechanisms which make data aggregation more efficient. One of these focuses on establishing a proper routing schemes. These schemes organize the sensor nodes into chain, a tree or clusters. As a brief description, *Chain-based data aggregation algorithms* organize sensor nodes as a shortest chain along which data is transmitted to the sink. While *Tree-based data aggregation algorithms* organize sensor

nodes into a tree. Data aggregation is performed at intermediate nodes along the tree and a concise representation of the data is transmitted to the root node which is usually the sink [22][18]. *Cluster-based data aggregation algorithms*, organize sensors into clusters. Each cluster has a designated sensor node as the cluster head which aggregates data from all the sensors in the cluster and directly transmits the result to the sink (such as [29, 17]). In this paper, we adopt cluster-based data aggregation schemes. The adoption of other schemes is discussed in our future work.

3.4 Related works

A significant challenge in WSNs is to prolong the monitoring operation of sensor nodes by efficiently using their limited energy, bandwidth and computation resources. Due to the high density in the network topology, sensor observations are highly spatially correlated. By allowing the nodes to corporate to carry out joint data from aggregation, the amount of data communicated within the network can be reduced. Recent techniques for processing multiple sensor streams in an energy efficient manner has been proposed. These techniques make use of both spatial and temporal correlations as well as clustering approaches to perform data reduction.

The work in [16] proposes an algorithm that manages spatio-temporal data in a distributed fashion, it performs in-network regression using kernel functions assuming rectangular $regions\ of\ support$. The network is assumed to contain multiple overlapping regions and in each region significant spatial correlations are expected to be observed. The authors introduce the usage of kernel functions; a region's kernel function maps a point x to a number, depending on the position of x in the region. Other techniques propose the use of statistical models of real world processes to reduce the cost of sensing and communication in sensor networks such as [10]. The prototype built, BBQ, consists of a declarative query processor and an underlying probabilistic model and planner, based on time-varying multivariate Gaussians.

Here, we extend the previous part to integrate spatial similarity measurement. This work relays on the idea of decreasing the communication overhead between nodes towards the base station by: 1- detecting and reducing redundancy by exploiting data similarity measurements. 2- reducing the number of communicated bits since it was indicated that the transmission of one bit over 100 meters would cost about the same level of energy as executing 3000 instructions in sensor node [13]. An aggregator will not just perform temporal aggregation, but also checks for data correlation during aggregation to reduce his data transmission amount.

Due to the spatial correlation in the sensed data, aggregation techniques have been incorporated into the routing protocols. Different routing strategies have involved data compression via coding in correlated data aggregation, to reduce data traffic. These strategies (like aggregation with Distributed source coding strategy-DSC [7]) have been based on lossless coding schemas (such as Slepian-wolf coding [6]) in data aggregation. In this paper, we do not propose a clustering/routing methods or discuss a coding schema and its dependence of optimal cluster-sizes on spatial correlations. We suppose that clusters and clusters heads (CH) are determined using distributed or centralized methods such as LEACH, HEED [17, 29] and others [24]. We assume having a suitable routing protocols that do not interfere with the spatial aggregation described in section 3.5. Each non cluster head node sends data to the CH node in its own cluster instead of to the base station (BS). The approach of clustering has the following advantages: 1) non-CH sensor nodes can save energy consumption because the nodes can avoid long-distance communication and have only to send data to its own CH being nearby and 2) the amount of data to be sent to BS can be reduced, which also saves the energy consumption. In what follows, we propose an algorithm based on time series estimations to investigate temporal and spatial correlations to reduce the communication overhead and ensure estimation accuracy. Our experiment results show that the relative errors between estimations made by the source sensors and the ones deduced by the sink, are very small.

3.5 Data similarity detection task

Given a typical WSNs in which each node records information from its vicinity and transfers this information to a centralized base station. Nodes which are close to each other, eventually sense similar information. Hence the information is geographically correlated. So, before sending it to the central agent, a huge saving in data transmission costs may be achieved by aggregating information from nearby nodes, removing redundancy and keeping data transmission to a minimum. We will call these geographical regions: "similarity regions", denoted by R_j and we consider that these regions are pre-defined by the base station at the deployment time. Since clusters are determined based on nodes' battery level, their coverage capabilities, the communication cost and the node density such as [17, 29, 5] it may happen that two neighboring clusters may share spatial data correlation. Hence a similarity region may contain different clusters spatially correlated as shown in Figure 6. We denote by Λ_j the number of clusters in a similarity region R_j.

Our goal is to detect data similarities during aggregation to keep data trans-

mission and overhead to a minimum. Thus, during aggregation (intra-/inter-clustering aggregation), an aggregator verifies the source of each received data (if it is a source node inside its own cluster or a CH node of a neighboring cluster) as well as the degree of similarity to reduce redundancy and overhead communication as possible. Along this line of thoughts, event detection/anomalies can be also observed. Let us describe some notations used

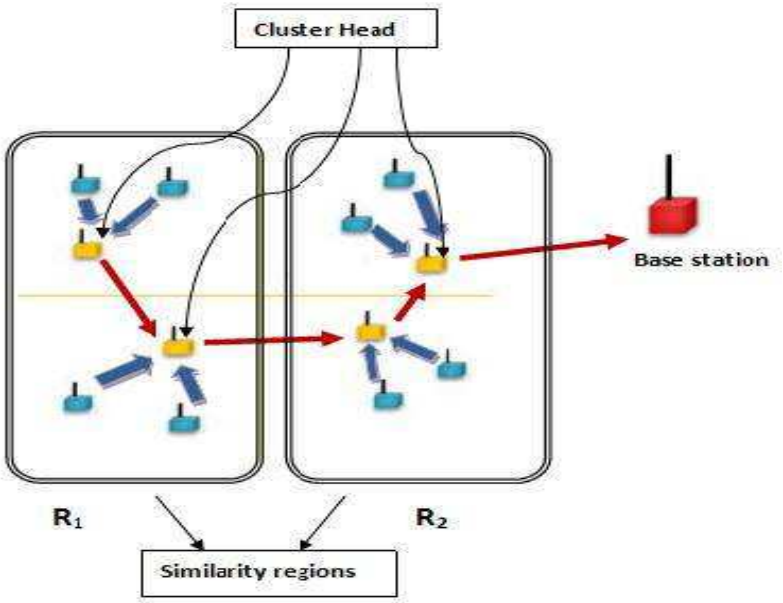


FIGURE 6
The concept of similarity region.

in this section:

- δ: the spatial similarity degree threshold. More specifically, δ_{R_j} and δ_c are respectively the degree of similarity thresholds inside a similarity region R_j and cluster (determined during clustering and similarity regions decomposition).

- Γ: the data similarity threshold. Precisely, Γ_{R_j} and Γ_c refers respectively to the degree of data similarity inside a region R_j and a cluster (The choice of Γ values could be a user defined threshold when clusters and similarity regions are determined).

- $d_{n_1 n_2}$: the distance between two nodes n_1 and n_2. For simplicity, we refer to the euclidean distance.

For every pair of nodes n_1 and n_2 belonging to the same cluster, $d_{n_1 n_2} \leq \delta_c$. While belonging to the same similarity region, $d_{n_1 n_2} \leq \delta_{R_j}$ (We denote by d_{SCH} the distance between a source S and a cluster head CH). If $\Lambda_j = 1$, the similarity region R_j is a cluster with $\delta_{R_j} = \delta_c$ and $\Gamma_{R_j} = \Gamma_c$.

- We choose to capture the (dis)similarity between two arrays of data A and B of length p by $K(A, B) = \Pi_{i=0}^{p} k(A_i, B_i)$.

We capture the temporal data correlation using our proposed model in section 2.3, each sensor S uses a prediction model to estimate future environment values and communicates with the sink only when a prediction threshold violation occurs. the values to be transmitted are no more the model parameters or recent data raw, but a recent number p of error values e_i^S where $i \in \{0, .., p\}$. An aggregator, while monitoring the environment may receive an array of data error values (e_i^S) from a source sensor in its cluster or a CH neighboring node, and it may combine these values with its own error values e_i^{agg}, if it has, then routes these values to the base station.
During aggregation, distinguishing between fused data is important. Simply speaking, fusing similar data can ensure redundancy and leads to huge communication cost. In fact, two nodes n_1 and n_2 located in the same similarity region or cluster, may have their values ($e_i^{n_1}$) and ($e_i^{n_2}$) highly correlated, which produces redundancy if they are sent both to the sink. Also, the choice of a similarity region can ensure reliable event detection/malicious tasks or anomalies, based on similarity measurement during inter-clustering fusion, for example when a CH node sends to another CH (resided in the same similarity region) an information that is not similar to the latter one, while it should be, one can deduce that something interesting has happened.

We assume that neighbor nodes monitor the same event, the position of each sensor is predetermined. Our algorithm is presented as follows: When a sensor node S sends data array (e_i^S) to a CH node, the latter -before starting processing these data- calculates the spatial degree of similarity to ensure if they are in the same cluster or similarity region:

- If $d_{SCH} \leq \delta_c$: The source and the CH nodes are in the same cluster. Then, the aggregator uses the Gaussian kernel function to calculate the degree of data similarity $K(e^S, e^{CH}) = \Pi_{i=0}^{p} k(e_i^S, e_i^{CH})$ between his array of data (e_i^{CH}) and (e_i^S) of the source.
  - if $K(e^S, e^{CH}) \geq \Gamma_c$, the values are highly spatially correlated and redundancy occurs. Then the aggregator routes its own data

values (e_i^{CH}) toward the sink. But, to ensure reasonable data quality and accuracy by the sink while updating its prediction model, the CH sends in addition to its own values, the array of similarity measures $k(e_i^S, e_i^{CH})$ between its data values and the sensors' ones. These values belong to $[0, 1]$, and we consider that sending their decimal part (integer values) instead of the main error values (e_i^S) (float values) can help reducing the data traffic in terms of number of bits. Note that the choice of a node CH to send its (e_i^{CH}) values is more energy saving especially in case of data fusion coming from different sensors.

  - if $K(e^{CH}, e^S) \prec \Gamma_c$. This indicates that an anomaly may occur, sensors are misbehaving or that something interesting has happened (e.g., the sensor became hot because a fire started nearby), these cases of which the sink should be aware. In this case, the aggregator decides to send both values e_i^{CH} and e_i^S toward the sink.

- Otherwise, if sensors are in the same similarity region ($d_{SCH} \leq \delta_{R_j}$) , the CH node follows the same process mentioned above by changing Γ_c to Γ_{R_j}

- Note that if both sensors are not in the same similarity region, the aggregator decides to send both values e_i^{CH} and e_i^S.

In the following section, we apply our methodology using a simple example, and we try to figure out the communication overhead and data prediction accuracy between the sink and the source sensors (note that a sensor and a sink use the same time series prediction model with order $k = p = 3$). In the experiment below, we considered two values as similar when their degree of similarity $k(e_i^S, e_i^{CH}) \geq 0.3$ hence $K(e^S, e^{CH}) \geq (0.3)^p$. We then defined $\Gamma_c = 0.027$.

4 EXPERIMENTATION AND ACCURACY RESULTS

We applied our algorithm on a simple 2-hop network topology composed of a sensor, an aggregator and a sink. We assume that the source sensor and the CH nodes are in the same similarity region R_j ($\Lambda_j = 1, \Gamma_{R_j} = \Gamma_c = 0.027$). Since our prediction model in [14] is suitable for a slow variation measurements, we applied our algorithm on different real data values[‡] : Wind speed at Lille (aggregator at Paris) (see Figure 7), Humidity average at Limoges

[‡] http://www.wunderground.com/global/stations/

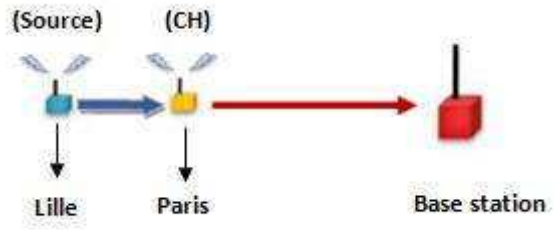


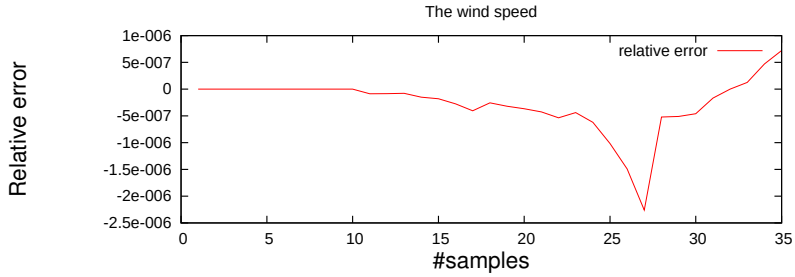
FIGURE 7
Wind speed measurement: the source sensor is located at Lille while the cluster head at Paris.

and Sea Low Level pressure at Limoges (aggregator at Lyon). We measure the data prediction accuracy between the sink and the source estimations after training the sink prediction model based on the aggregated data.

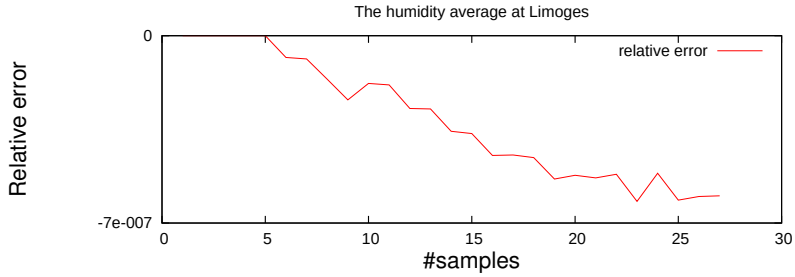
Figure 8 shows that the relative error values between source and sink estimations $\in [-7\times10^{-7}, 10^{-6}]$, which maintain good data quality and accuracy. In addition, Table 3 shows the number of data aggregated and communicated to the base station before and after using the similarity measurement. The data could be the error values (float numbers) and/or the similarity degree (integer values) according to the similarity examination results. We can see that the number of floats communicated after introducing the similarity measurement is reduced about 41% for wind speed, $\sim$ 20% for humidity and $\sim$ 39% for sea low level pressure which can increase energy saving since the number of transmitted bits is reduced. Table 4 represents the total overhead in terms of bytes before and after using the similarity measurements. if we consider that an integer is represented on 4 bytes and a float on 8 bytes. Our experiment shows a reduction in terms of bits of about $\sim$ 20% for wind speed, $\sim$ 10% for humidity and $\sim$ 20% for sea low level pressure.

	Using sim. meas.		**Without sim. meas.**	
Data traffic	*# float(a)*	*# int.*	*# float(b)*	$\frac{(a)}{(b)}$
Wind speed	51	36	87	$\sim$ 0.586
Humidity	24	6	30	0.8
Pressure	69	42	111	$\sim$ 0.62

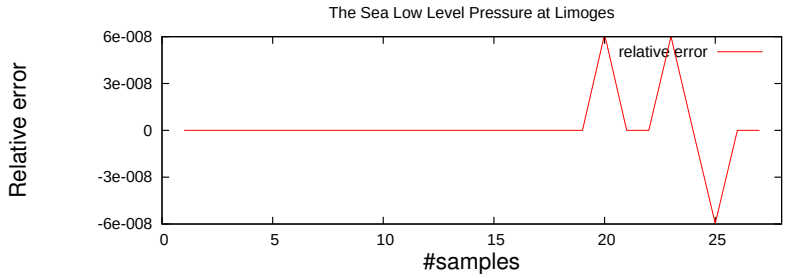
TABLE 3
The data traffic produced before and after introducing the similarity measurements. Here $\frac{(a)}{(b)}$, is the fraction between the data traffic (in terms of floats) before and after using similarity measurements .



(a) Wind speed at Lille



(b) Humidity average at Limoges



(c) Sea Low Level Pressure at Limoges

FIGURE 8
The relative error between the sensor and sink estimations.

Data traffic	***Without sim.(a)***	***Using sim.(b)***	$\frac{(b)}{(a)}$
Wind speed	696 bytes	552 bytes	~ 0.79
Humidity	240 bytes	216 bytes	0.9
Pressure	888 bytes	720 bytes	~ 0.81

TABLE 4
The data traffic represented in terms of bytes before and after introducing the similarity measurements.

5 CONCLUSION

The batteries on today's wireless sensor barely last a few days, and nodes typically expend a lot of energy in computation and wireless communication.

Hence, the energy efficiency of the system is a major issue. Data collection process and redundancy might have their negative impact on wireless network (e.g., waste of energy and bandwidth) due to high data communication traffic and rate. We have adopted time series forecasting techniques and we have proposed an algorithm based on the AutoRegressive model (AR), to predict local readings and reduce the data communication rate generated by sensors. We also integrated data similarity measurements based on kernel methods to reduce the overall communication load and avoid the transmission of redundant messages. Our experiments show that it's possible to reduce the communication overhead between nodes while ensuring a reasonable data quality and accuracy. Our future work is to enhance our algorithm and experiment it on complex topologies with clustering/routing methods since our experiments has focused on a simple case topology.

REFERENCES

[1] Ameer Ahmed Abbasi and Mohamed Younis. (2007). A survey on clustering algorithms for wireless sensor networks. *Comput. Commun.*, 30(14-15):2826–2841.

[2] Ian F. Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and Erdal Cayirci. (2002). Wireless sensor networks: a survey. *Computer Networks*, 38:393–422.

[3] A. Smola. B. Scholkopf. (2002). Learning with kernels. In *Learning with Kernels*.

[4] P. J. Brockwell and R. A. Davis. (March 2002). *Introduction to Time Series and Forecasting*. Springer.

[5] Haowen Chan and Adrian Perrig. (2004). Ace: An emergent algorithm for highly uniform cluster formation. In *Proceedings of the First European Workshop on Sensor Networks (EWSN)*, pages 154–171.

[6] T. M. Cover and J. A. Thomas. (1991). *Elements of Information Theory*. John Wiley.

[7] R. Cristescu, B. Beferull-Lozano, and M. Vetterli. (January 2004). Networked slepian-wolf: Theory and algorithms. In *Proceedings of the First European Workshop on Sensor Networks (EWSN)*, Berlin, Germany.

[8] N. Cristianini and J. Shawe-Taylor. (2000). An introduction to support vector machines. In *An Introduction to Support Vector Machines*.

[9] Scott D. (1992). *Multivariate Density Estimation: Theory Practice and Visualization*. NY: John Wiley and Sons, New York.

[10] A. Deshpande, C. Guestrin, S. Madden, J. M. Hellerstein, and W. Hong. (2004). Model-driven data acquisition in sensor networks. In *VLDB*, pages 588–599.

[11] Paul F. Evangelista, Piero Bonnisone, Mark J. Embrechts, and Boleslaw K. Szymanski. (August 2005). Fuzzy roc curves for the 1 class svm: Application to intrusion detection. In *Proceedings International Joint Conference on Neural Networks, IJCNN*.

[12] Maria florina Balcan, Avrim Blum, and Nathan Srebro. (2006). On a theory of learning with similarity functions. In *International Conference on Machine Learning*, pages 73–80.

[13] R. K. Ganti, P. Jayachandran, H. Luo, and T. F. Abdelzaher. (2006). Datalink streaming in wireless sensor networks. In *SenSys '06*, Boulder, Col., USA. ACM.

[14] Alia Ghaddar, Isabelle Simplot-Ryl, David Simplot-Ryl, Tahiry Razafindralambo, and Samar Tawbi. (june 2009). Algorithmes pour l'estimation des donnes dans les rseaux de capteurs. *11emes Rencontres Francophones sur les Aspects Algorithmiques de Tlcommunications (AlgoTel)*, pages 93–96.

[15] Alia Ghaddar, Isabelle Simplot-Ryl, David Simplot-Ryl, Tahiry Razafindralambo, and Samar Tawbi. (October 2010. To appear.). Towards energy-efficient algorithm-based estimation in wireless sensor networks. In *Proceedings of the 6th International Conference on Mobile Ad-hoc and Sensor Networks (MSN'10)*, Hangzhou, China.

[16] Carlos Guestrin, Peter Bodík, Romain Thibaux, Mark A. Paskin, and Samuel Madden. (April 2004). Distributed regression: an efficient framework for modeling sensor network data. In *IPSN*, pages 1–10.

[17] Wendi B. Heinzelman, Anantha P. Ch, IEEE, Anantha P. Chandrakasan, Member, and Hari Balakrishnan. (2002). An application-specific protocol architecture for wireless microsensor networks. *IEEE Transactions on Wireless Communications*, 1:660–670.

[18] Chalermek Intanagonwiwat, Ramesh Govindan, and Deborah Estrin. (2000). Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *MOBICOM*, pages 56–67. ACM.

[19] M. Dohler J.-L. Lu, F. Valois. (July 20010). Optimized data aggregation in wsns using adaptive arma. *SensorComm*.

[20] A. Jain, E. Y. Chang, and Y.-F. Wang. (2004). Adaptive stream resource management using kalman filters. In *SIGMOD '04*, Paris, France. ACM.

[21] B. Krishanamachari, D. Estrin, and S. Wicker. (July 2002). The impact of data aggregation in wireless sensor networks. In *International Workshop of Distributed Event Based Systems (DEBS)*, pages 575–578, Vienna, Austria.

[22] Samuel Madden, Michael J. Franklin, Joseph M. Hellerstein, and Wei Hong. (2002). Tag: a tiny aggregation service for ad-hoc sensor networks. *SIGOPS Oper. Syst. Rev.*, 36(SI):131–146.

[23] John Shawe-Taylor and Nello Cristianini. (June 2004). *Kernel Methods for Pattern Analysis*. Cambridge University Press.

[24] Noritaka Shigei, Hiromi Miyajima, Hiroki Morishita, and Michiharu Maeda. (March 2009). Centralized and distributed clustering methods for energy efficient wireless sensor networks. In *Proceedings of the International MultiConference of Engineers and Computer Scientists (IMECS)*, volume I, Hong Kong.

[25] M. Stemm and R. H. Katz. (August 1997). Measuring and reducing energy consumption of network interfaces in hand-held devices. *IEICE Transactions on Communications*, E80-B(8):1125–1131.

[26] D. Tulone. (October 2004). A resource-efficient time estimation for wireless sensor networks. In *DIALM-POMC'04*, pages 52–59.

[27] D. Tulone and Samuel Madden. (2006). Paq: time series forecasting for approximate query answering in sensor networks. In *EWSN*, pages 21–37.

[28] R. van Renesse. (April 2003). The importance of aggregation. In *Future Directions in Distributed Computing*, volume 2584 of *LNCS*, pages 87–92. Springer-Verlag.

[29] Ossama Younis and Sonia Fahmy. (2004). Heed: A hybrid, energy-efficient, distributed clustering approach for ad-hoc sensor networks. *Mobile Computing, IEEE Transactions*, 3(4):366–379.