Reducing The Data Transmission in Wireless Sensor Networks Using The Principal Component Analysis

Amirmohammad Rooshenas [1], Hamid R. Rabiee [2], Ali Movaghar [3], M. Yousof Naderi [4]

Department of Computer Engineering, Sharif University of Technology
Tehran, Iran

[1] rooshenas@ce.sharif.edu
[2] rabiee@sharif.edu
[3] movaghar@sharif.edu
[4] mynaderi@ce.sharif.edu

***Abstract*—Aggregation services play an important role in the domain of Wireless Sensor Networks (WSNs) because they significantly reduce the number of required data transmissions, and improve energy efficiency on those networks. In most of the existing aggregation methods that have been developed based on the mathematical models or functions, the user of the WSN has not access to the original observations. In this paper, we propose an algorithm which let the base station access the observations by introducing a distributed method for computing the Principal Component Analysis (PCA). The proposed algorithm is based on transmission workload of the intermediate nodes. By using PCA, we aggregate the incoming packets of an intermediate node into one packet and as a result, an intermediate node merely sends a packet instead of relaying all the incoming packets. Consequently, we can achieve considerable reduction in data transmission. We have analyzed the performance of the proposed algorithm through numerical simulations. The experimental results show that our algorithm performs better than the existing state of the art PCA-based aggregation algorithms such as PCAg in terms of accuracy and efficiency.**

I. Introduction

In the recent years, there have been a considerable amount of research on the area of data reduction in wireless sensor networks. As the radio board of a sensor node consumes most of the available power, data reduction techniques could be used to prolong the lifetime of wireless sensor networks. Among these techniques, which are classified in [1], data prediction and data aggregation are the most relevant categories to our work. In data prediction, sensor nodes do not continuously send their measured values to the base station and let the base station predict the required values. Thus, the base station often maintains the estimation of an observation instead of its actual value. In data aggregation, each sensor node sends the same number of packets in different periods, and the size of these packets are the same. Indeed, each packet carries the result of the in-network computation called partial value for the next intermediate node in the aggregation tree. In fact, data aggregation techniques are used for reducing the number of data transmissions by distributing the function which needs all sensors' observations throughout the network, so each node shall apply the function to its received values. Recent works [2], [3], [4] have focused on reducing the number of transmitted bytes by performing in-network data aggregation.

In this paper, we use a combination of data predication and data aggregation techniques. Since we separate observations into a base and some projected values, and use one base for more than one observation, such that the reconstructed signals at sink bear approximation errors, we classify our method in the data predication category. On the other hand, each node sends the Principal Components(PCs) of its received values to the next node, and the PCs have the same size. In addition, the duty of the base station is to compute the PCs of all sensors, and hence we have a in-network aggregation service for the PC computation.

Although most of the applications of principal component analysis (PCA) are not in the field of sensor networks, some recent work have used PCA to extract features out of wireless sensor data. In [5], a data aggregation method using PCA compression techniques is proposed to fuse the information from multiple sensors. Borgne et al. [6] propose a decentralized distributed PCA in wireless sensor networks to aggregate the transmitted data called PCAg. The main problem of PCAg [6] is the frequent sending of updated eigenvectors from a base station to its sensor nodes. In [7] a method called DPCA is proposed to solve this problem. By using DPCA, each sensor can calculate the needed elements of the eigenvector without any communications to the base station. The DPCA uses the broadcast nature of wireless communications and assumes that sensor nodes which are not in the radio range of each other have not any data correlation.

In this paper, we propose a new data aggregation algorithm in which intermediate nodes aggregate the incoming packets by means of PCA into a single packet and transmit the aggregated results to their parents. This approach extends the important previous works on PCA-based data aggregation. Besides the main algorithm, we propose an aggregation service to compute the reconstruction error at the base station. By using this aggregation service, we enhance the PCAg algorithm, and we compare the enhanced method with our proposed method. Our numerical results which are based on a real-world temperature measurement illustrate that the proposed method has better performance in terms of accuracy and efficiency.

The rest of the paper is organized as follows. A brief description of the PCA algorithm is provided in Section 2. We introduce the proposed method in Section 3. Section 4 provides the simulation results. Finally, we conclude the paper in Section 5.

II. Background

A. Principal Component Analysis

Let $\mathbf{x}[t] = [x_1[t], \cdots, x_n[t]]^T$ represent the observations of n sensor nodes and $t = \{1, 2, 3, \cdots\}$ the sampling period (epoch) at which the sensor measurements are collected. Therefore, $x_i[t]$ shows the sensed observation of sensor i at time t.

The vector $\mathbf{x}$ can be transformed into a new space by

$$\mathbf{y}[t] = \mathbf{P}^T \mathbf{x}[t] \tag{1}$$

Where $\mathbf{P}$ is an orthonormal transformation matrix. If we Assume this matrix is available, the value of $\mathbf{x}[t]$ can be reconstructed using $\mathbf{y}$ and the following equation:

$$\hat{\mathbf{x}}[t] = \mathbf{P}\mathbf{y}[t] \tag{2}$$

Therefore, instead of sending $\mathbf{x}[t]$, a source node may only send $\mathbf{y}[t]$. hence, if the dimensions of $\mathbf{y}$ is less than the dimensions of $\mathbf{x}$, the amount of the transmitted bytes will be reduced and more energy will be saved. This dimension reduction is lossy, therefore, $\hat{\mathbf{x}}[t]$ is an approximation of $\mathbf{x}[t]$. To measure the accuracy of $\hat{\mathbf{x}}[t]$, the reconstruction error is defined as:

$$e = \left\| \hat{\mathbf{x}[t]} - \mathbf{x}[t] \right\| \tag{3}$$

Let q be the number of the dimensions of $\mathbf{y}[t]$, then the kth dominant eigenvector of the covariance matrix of x is defined as the eigenvector corresponding to its kth largest eigenvalues. If the first q dominant eigenvectors are used as the columns of $\mathbf{P}$, the reconstruction error will be minimized and $\mathbf{y}[t]$ will represent the principal components(PCs) of $\mathbf{x}[t]$. Dominant eigenvectors show the direction in which data has maximum variance, therefore if all the eigenvectors are selected, total variance of data in all directions will be saved. Otherwise, the variance are removed in the directions corresponding to the discarded eigenvectors. As eigenvalues show the amount of variance conserved by eigenvectors, their sum is equal to total variance of the original data:

$$\sum_{i=1}^{n} \lambda_i = E[(\mathbf{x} - E[\mathbf{x}])^T (\mathbf{x} - E[\mathbf{x}])] \tag{4}$$

where λ_i is the eigenvalue corresponding to the ith dominant eigenvector. Therefore, to measure the accuracy of a dimension reduction method, the retained variance metric is used as:

$$H(q) = \frac{\sum_{k=1}^{q} \lambda_k}{\sum_{k=1}^{n} \lambda_k} \tag{5}$$

where q and n are the sizes of the new and original dimension, respectively.

For computation of the dominant eigenvector, we use the Power Iteration Method (PIM), which has been introduced in [8]. In PIM, the following iteration converges to the dominant eigenvector.

$$\mathbf{v}^{k+1} = \mathbf{C}\mathbf{v}^k \tag{6}$$

where $\mathbf{C}$ is the covariance matrix of the observations. If the eigenvalue corresponding to the dominant eigenvector is larger compare to the other eigenvalues, iteration (6) converges to the dominant eigenvector. The larger this eigenvalue is, the faster the PIM method converges. The PIM method is suitable only for the dominant eigenvectors, therefore FastPCA[9], as an extension of PIM, can be used to find the other eigenvectors.

B. Collective Principal Component Analysis

Suppose sensors 1 to n gather data $\mathbf{x}_1[t]$ to $\mathbf{x}_n[t]$, and we want to calculate the principal components (PCs) of $\mathbf{x}[t] = [\mathbf{x}_1^T[t], \cdots, \mathbf{x}_n^T[t]]^T$. Generally, to calculate the principal components, each sensor i sends its vector $\mathbf{x}_i[t]$ to the base station. Then $\mathbf{x}[t]$ will be constructed in the base station and its PCs will be computed by means of the eigenvectors of $x[t]$.

In [10], the Collective PCA (CPCA) has been proposed in which sensors send PCs ($\mathbf{y}_i[t]$) instead of vector $\mathbf{x}_i[t]$ to the base station. Since principal components have less dimensions in comparison to $\mathbf{x}_i[t]$, the number of transmitted bytes are reduced. The CPCA method is derived based on a property which states that principal components are invariant under orthogonal transformations[11].

By using CPCA, we can calculate the principal components of $\mathbf{x}[t]$ as the following. First, the $\mathbf{z}[t]$ is created as

$$\mathbf{z}[t] = [\mathbf{y}_1^T[t]; \mathbf{y}_2^T[t], \cdots]^T \tag{7}$$

Then, the principal components of $\mathbf{x}[t]$ is computed from the following equation if $\mathbf{v}_i$ for $1 < i < n$ represent eigenvectors of $\mathbf{z}[t]$.

$$\mathbf{y}[t] = \sum_{i=1}^{n} \mathbf{v}_i^T \mathbf{z}[t] \tag{8}$$

To reconstruct data in the base station, the eigenvectors are needed. Let $\mathbf{P}_1$ to $\mathbf{P}_n$ be the eigenvectors of $\mathbf{x}_1[t]$ to $\mathbf{x}_n[t]$. Therefore, the eigenvectors of $\mathbf{x}[t]$ are obtained according to the following equation:

$$\mathbf{w}_i = \mathbf{A}\mathbf{v}_i \quad where \quad \mathbf{A} = \begin{bmatrix} \mathbf{P}_1 & 0 & \dots & 0 \\ 0 & \mathbf{P}_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{P}_n \end{bmatrix} \tag{9}$$

and $\mathbf{v}_i$ is the ith dominant eigenvector of the covariance matrix of $\mathbf{z}[t]$. Because all the eigenvectors of $\mathbf{x}_i[t]$ do not contribute to the computation of $\mathbf{y}_i[t]$, the CPCA may have errors as described in [12].

III. The Proposed Method

A. Problem Definition

We assume a general multi-hop network which consists of n sensor nodes and one base station called sink. A node can either send or receive observations or principal components. Given the limited communication range of sensor nodes, sending a data from a node to the sink typically results in a series of hops through the network. Therefore, the intermediate nodes consume more energy for relaying packets, and the nodes located in the vicinity of the base station significantly have a shorter lifetime. In order to extend their lifetime, the intermediate nodes can aggregate all the receiving packets and forward only aggregate values toward the base station. We assume the aggregation is performed over an aggregation tree which is a directed tree formed by the union of all the paths from the sensor nodes to the base station. These paths may be arbitrarily chosen and necessarily are not the shortest paths. We consider a simple type of an aggregation tree which have a two-level structure. In this structure, the base station is root of the tree and the sensor nodes have the maximum of two hops to reach the base station. The aggregation tree is synchronized and our algorithm takes the structure of the aggregation tree. In addition, the intermediate nodes send the first principal component of the incoming data which includes observations as well as the received principal components from the preceding intermediate nodes.

B. LocalPCA

The goal of LocalPCA algorithm is to prolong the lifetime of sensor network by reducing the number of transmitted bytes in the intermediate nodes. In this algorithm, both simple and complex aggregations are used. The simple aggregation process is done on data which is coming from the leaf nodes to their parents. But in complex aggregation, instead of the observation data, the intermediate nodes process on aggregate vectors which are coming from other intermediate nodes. In the proposed algorithm, each intermediate node at the first level of routing tree receives observations from leaf nodes and constructs vector $\mathbf{x}[t]$ as described in SectionII-A. Then it reduces the dimensions of $\mathbf{x}[t]$ by means of PCA. The number of the new dimensions (q) can be determined by using the retained variance formula (5). To calculate the PCA, an intermediate node(in each epoch), computes the PCs $\mathbf{y}[t]$, of the vector $\mathbf{x}[t]$ by using equation (1). At the commencement of the computation, the eigenvectors, $\mathbf{P}$, are selected randomly.

In the next step, the reconstruction error is derived based on (3). This error can be obtained by using $\hat{\mathrm{x}}[t]$ or directly from (12) that will be described in Section III-C. If reconstruction error exceeds a predefined threshold, eigenvectors become invalid and must be updated. The updated version of $\mathbf{P}$ shall be sent to the base station and $\mathbf{y}[t]$ must be recalculated with the new $\mathbf{P}$. Finally, each intermediate node sends the new $\mathbf{y}[t]$ to its parent node. Actually, the PIM method for $q = 1$ and FastPCA method for $q > 1$ are used to update the eigenvectors. It is worth to mention that threshold of the tolerable error must be set according to the the application of sensor networks. Generally, iteration (6) starts with a random vector but if the previous invalidated eigenvectors are used, the iteration will converge faster.

When aggregation tree is multi-level, there are some intermediate nodes which receives data not only from leaf nodes but also from one or more of the other intermediate nodes. Therefore, the aggregation process becomes more complex. For this case, we use CPCA method as described in Section II-B. Actually, each vector $\mathbf{z}[t]$ of the intermediate node i, includes observations from itself and its leaf nodes as well as the PCs of the other intermediate nodes. The $\mathbf{z}[t]$ can be defined as

$$\mathbf{z}[t] = [\mathbf{y}_1^T[t], \cdots, \mathbf{y}_n^T[t], x_1[t], \cdots, x_m[t], s[t]]^T \qquad (10)$$

Where $\mathbf{y}_1[t]$ to $\mathbf{y}_n[t]$ are PCs of other intermediate nodes, $x_1[t]$ to $x_m[t]$ are leaf observations, and s[t] is the sensed value of current intermediate node on epoch t. In complex aggregation, the process of calculating PCs and updating eigenvectors are the same as simple aggregation except that $\mathbf{z}[t]$ shall be used instead of $x[t]$. Therefore, each intermediate node i sends $\mathbf{y}_i[t]$ which has the same size among all the intermediate nodes.

The computation of PCs can be carried out in the form of an aggregation service. In this way, after each node receives the observations and PCs from its children, it computes and send PCs of the received values to the next node by using the aggregation service. The computed PCs represent all principal components of all observations of the downstream sensors in the routing tree. In the base station, the final eigenvectors are computed by using of the equation (9), and then all sensed observations can be reconstructed by using the received PCs and the measured final eigenvectors.

To clarify the computation process of the final eigenvectors in the base station, let i represent an intermediate node with two children, k and j, and let the base station be its parent, and let j and k have $n-1$ and $m-1$ child nodes, respectively. When the base station receives p_i , p_k and p_j dominant eigenvectors from nodes i, j, and k, it can measure the final dominant eigenvector by

$$\mathbf{p} = \begin{bmatrix} p_{i1} * \mathbf{p}_j \\ p_{i2} * \mathbf{p}_k \\ p_{i3} \end{bmatrix} \qquad (11)$$

Where p_{il} shows the lth element of vector $\mathbf{p}_i$. If $\mathbf{p}_j$ and $\mathbf{p}_k$ have n and m elements respectively, the final reconstructed vector $\hat{\mathrm{x}}[t]$ would have $n + m + 1$ elements whose first n elements belong to the node j and its children, and the next m elements pertain to node k and its children. The last element of $\hat{\mathrm{x}}[t]$ is the sensed value of node i. The location of each sensor at $\hat{\mathrm{x}}[t]$ can be determined when routing tree is constructed or eigenvectors are updated.

```
Algorithm 1: LocalPCA method
Input: data packets from the precedent nodes : $D$
Input: principal packets from the precedent nodes : $Y$
Input: dominant eigenvector from previous execution : $p$
Output: aggregated data for the following node : $y_{aggr}$

create vector $x$
for $y$ in $Y$
    i = corresponding index to the child node
    $x_i = y$
for $d$ in $D$
    i = corresponding index to the child node
    $x_i = d$
$y_{aggr} = \mathbf{p}^T * x$
$\hat{\mathbf{x}} = \mathbf{p} * y_{aggr}$
$e = \|\hat{\mathbf{x}} - \mathbf{x}\|$
If e > threshold
    while $p$ coverges
        $A = x * x^T$
        $\mathbf{p} = A * \mathbf{p}$
    $y_{aggr} = \mathbf{p}^T * \mathbf{x}$
    new $\mathbf{p}$ should be sent for the following node
$y_{aggr}$ should be sent for the following node
```

C. An Aggregation Service for Reconstruction Error

In both LocalPCA and PCAg[13] methods, all eigenvectors and principal components are available at the base station, and $\hat{\mathbf{x}}[t]$ can be computed by using the equation (2). However, the base station cannot calculate the reconstruction error because it doe snot have access to the original value of $\mathbf{x}[t]$ in the sensor nodes. Thus the base station cannot measure the accuracy of the algorithms. Moreover, the accuracy parameter can be used to determine the error threshold of LocalPCA or updating rate of PCAg. To solve this problem, we propose an aggregation service in which the base station will be able to compute the reconstruction error. For this purpose, we expand the square of the reconstruction error:

$$\begin{aligned}\|\hat{\mathbf{x}} - \mathbf{x}\|^2 &= \mathbf{x}^T\mathbf{A}\underbrace{\mathbf{A}^T\mathbf{A}}_{I}\mathbf{A}^T\mathbf{x} - 2\mathbf{x}^T\mathbf{A}\mathbf{A}^T\mathbf{x} + \mathbf{x}^T\mathbf{x}\\ &= \mathbf{x}^T\mathbf{x} - \mathbf{x}^T\mathbf{A}\mathbf{A}^T\mathbf{x}\\ &= \mathbf{x}^T\mathbf{x} - \mathbf{y}^T\mathbf{y}\\ &= \sum_{i=1}^{n} x_i^2 - \sum_{i=1}^{q} y_i^2 \qquad q \ll n \end{aligned} \tag{12}$$

Where n is the number of the sensor nodes and q is the number of the principal components. Therefore, each sensor shall send the square of its observation to the next intermediate node. Then the next intermediate node should square its own observation and sum it with the received values and send the result to the next node. Therefore, the base station can calculate the reconstruction error by using PCs and sum of all squared observations. If we send the squared observation value along with partial value of the aggregation services used in either PCAg or LocalPCA, the number of the transmitted packets will not be increased. As a result, the base station can improve the performance of those methods by using the reconstruction error.

IV. Experiments

In this section, we investigate the performance of the proposed LocalPCA algorithm in terms of efficiency and accuracy in a multi-hop network via simulation and numerical analysis.

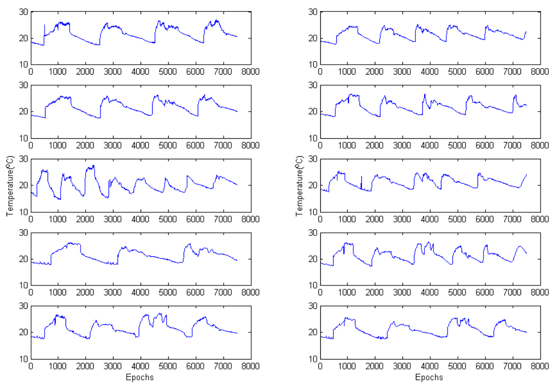

Fig. 1. Temperature Readings

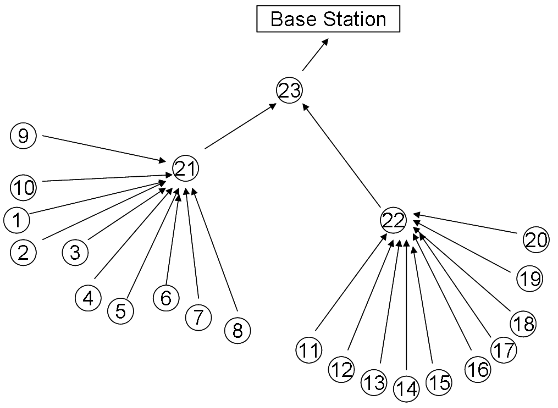

Fig. 2. Temperature Readings

The efficiency of the algorithm is evaluated by the number of transmitted bytes at the intermediate nodes. we assess the accuracy by means of the reconstruction error as defined in (3), because the base station uses an approximation of observations.

A. Simulation Settings

In our experiments, a trace of sensor data from Intel Berkeley Research lab is used. This data set contains observations of temperature, light, humidity and voltage collected every 31 seconds from 54 sensors. Among these observations, we selected the temperature data for our simulations.

Due to missing observation of the sensors, instead of interpolating the missing data, the order of recorded observations was considered. Actually, the consecutive observations were mapped to successive epochs and this approach is used throughout the experiments. An example of resulting temperature readings obtained from 10 Mica2Dot sensor sets is given in Figure 1.

For this data set, we analysed the accuracy of reducing data dimensions to one dimension using the equation (5). For this analysis, the data of 50 sensor nodes in 7500 epochs were used. The results show 100% accuracy for all temperature, humidity, voltage and light cases. Based on this analysis, we used the dominant eigenvector and the first principal component for the

TABLE I
RESULTS FOR 7500 EPOCH

Method Name	Transmitted Bytes	Error Mean	Error Variance	Error Max	Parameters
PCAg	200340	0.82	0.96	8.34	update:11 epoch
LocalPCA	168114	0.82	0.07	1.42	threshold at 23: 1.0 threshold at 22,21: 0.35
EAPCAg	215216	0.82	0.23	6.16	threshold: 1.42

proposed LocalPCA algorithm.

We use a hierarchical scenario as a sample of the real world WSNs. Figure 2 shows our hierarchical scenario. In this case, we used a tree topology in which 23 nodes were connected to one base station. The location of sensor nodes in Figure 2 only indicates the structure and relation between them and does not show the exact positions of the sensor nodes.

B. Comparison with Alternative Methods

1) PCAg: Let v_i be the ith element of eigenvector $\mathbf{v}$ corresponding to the sensor i, and let $x_i[t]$ represent observations of sensor i at time t. First, in PCAg [13], the base station gathers all $x_i[t]$ of sensors in predefined periods. Then it computes $\mathbf{v}$ and sends v_i to the corresponding sensor i. Consequently, each leaf sensor sends its partial value $x_i[t] * v_i$ to an intermediate sensor node. The intermediate node sums all the received partial values with its partial value and sends the result to the next node. Therefore, the base station receives $\sum_{i=1}^{n} x_i * v_i$ which is equal to the principal component for all of n sensors' observations. The base station, by using principal component and eigenvectors, can reconstruct all the observations. Due to unavailability of original observations in the base station, this method can not measure the reconstruction error.

2) Error Aware PCAg(EAPCAg): We enhance the PCAg algorithm by applying the proposed aggregation service of the reconstruction error that was described in Section III-C to the PCAg. In this way, the base station is able to update eigenvectors based on the measured reconstruction error. Actually, when the reconstruction error exceeds a predefined threshold, the base station asks all sensors to send their observations. Then the base station updates the eigenvectors and sends them back to the corresponding nodes in the network.

3) Distributed PCA(DPCA): Although DPCA[7] is also trying to distribute the computation of PCA, we did not choose this method in our comparison studies, because it assumes that only the observations of sensors which are in the radio range of each other are correlated. Therefore, DPCA divides the correlation matrix into sub-matrices and computes the eigenvectors and PCs of each sub-matrix independently. Since this correlation assumption is not sensible for temperature data, the error of this method is significant. This error is more considerable when only one PC is used [7].

C. Simulation Results

Table I summarizes our simulation results. Based on the results, LocalPCA is more efficient than the competing algorithms. For example, for node 23, the nearest intermediate node to the base station, the LocalPCA algorithm has the least

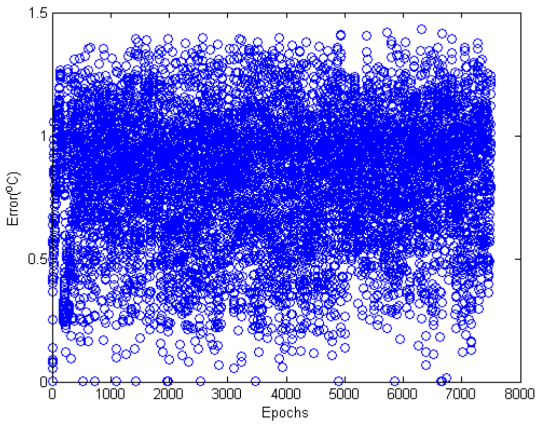

Fig. 3. Reconstruction Error in LocalPCA

number of the transmitted bytes followed by the PCAg and EAPCAg. At sensor node 23, EAPCAg method has sent about 15K bytes more than PCAg algorithm and PCAg method has sent about 32K bytes more than the LocalPCA. The average updating interval in EAPCAg is about 13.84 epochs which is around 3 epochs more than PCAg. Therefore, for updating eigenvectors, the EAPCAg has sent about 21K bytes less than PCAg. The main reason which causes EAPCAg to be less efficient than the PCAg is the fact that 4 bytes are added to the each PC packet of EAPCAg, for the computation of reconstruction error. Therefore, the size of each PC packet is increased from 14 bytes to 18 bytes. These packets is sent in each epoch, therefore EAPCAg sends about 15K bytes more than PCAg. In LocalPCA, the sensor node 23 has sent about 32K less than PCAg, because it uses reconstruction error, and the observations are not sent to the base station. Moreover, the length of eigenvector packets, in LocalPCA algorithm, only depends on the degree of each node, but for PCAg and EAPCAg, the length depends on the total number of sensor nodes in the network. Therefore, in PCAg and EAPCAg methods, when the number of sensor nodes are increased, the eigenvector packets should be divided into smaller packets. This adds some overhead to each packet, and increases the total number of transmitted bytes. But in LocalPCA, by increasing the number of sensor nodes usually the depth of routing tree and the average degree of tree will be increased, thus the total number of transmitted bytes will not change. The error of LocalPCA is bounded because when it exceeds a predefined threshold, the eigenvectors will be updated and

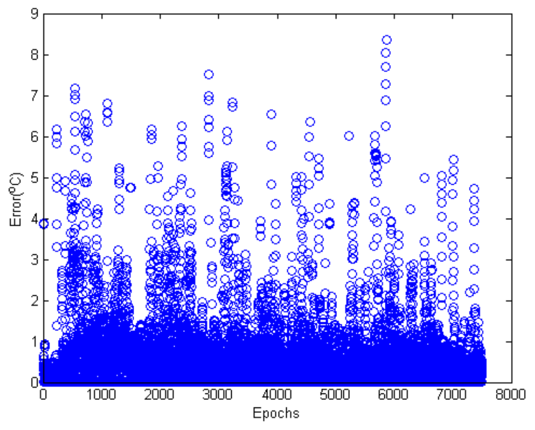

Fig. 4. Reconstruction Error in PCAg

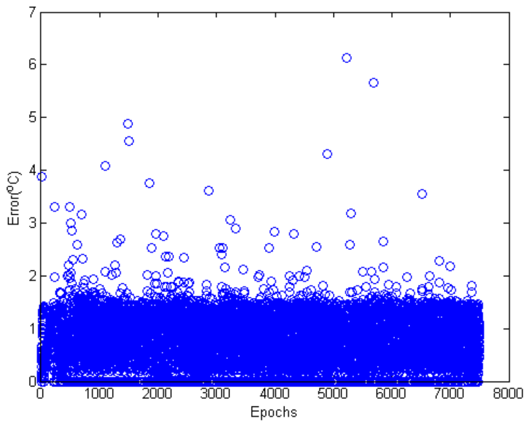

Fig. 5. Reconstruction Error in EAPCAg

the PCs will be recalculated with the new eigenvectors. It can be easily shown that the maximum reconstruction error for LocalPCA is always less than the sum of all thresholds. Actually this bound is very pessimistic because in LocalPCA each intermediate node updates its eigenvectors independently, and these updates may not occur at the same time.

According to Figure 3, we usually have errors in LocalPCA, but most of the errors are distributed near the average value. In PCAg, the error becomes zero after each updating process, but again it starts to increase until the next update. Because, the base station cannot measure the reconstruction error, and updating is done independent of the error. Therefore, changes in the environment cause PCAg to encounter unbounded errors. Figure 3 shows the variation of the reconstruction error for PCAg.

In EAPCAg, although the base station can measure the error, the eigenvectors are updated at the base station and will not be available to nodes until the next epoch. Therefore, the error can exceed the threshold, but after updating the eigenvectors, the error will be corrected. The error of this algorithm is shown in figure 5 In PCAg, in order to achieve the low average error, the updating intervals shall be reduced, and as a result, the frequency of updating will be increased. The smaller the updating interval, the more reduction of efficiency the algorithm have.

V. Conclusions

In this paper, we introduce a distributed method for computing the PCA through a new data aggregation algorithm which let the base station access the observations. The proposed algorithm is based on the transmission workload of the intermediate nodes. Using PCA, the incoming packets of an intermediate node are aggregated into one packet, so the intermediate node is able to just send one packet instead of relaying all the incoming packets. In this way, the number of transmitted bytes is reduced, and this reduction is considerable for the nodes located in the vicinity of the base station. In addition, we provided the PCAg method with an aggregation service to compute reconstruction error at the base station. This aggregation service helps PCAg to monitor the accuracy of the algorithm and, in turn, it can tune its update rate dynamically. Our extensive simulation results based on the accuracy and efficiency performance metrics illustrate the superior performance of the proposed algorithm comparing to other similar approaches.

References

[1] G. Anastasi, M. Conti, M. D. Francesco, and A. Passarella, "Energy conservation in wireless sensor networks: A survey," *Ad Hoc Networks*, vol. 7, pp. 537–568, 2009.

[2] Y. Yao and G. J., "The cougar approach to in-network query processing in sensor networks," *SIGMOD Record*, vol. 31, no. 3, pp. 9–18, 2002.

[3] C. Intanagonwiwat, D. Estrin, G. R., and J. Heidermann, "Impact of network density on data aggregation in wireless sensor networks." *ICDCS*, p. 457, 2002.

[4] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "Tag: a tiny aggregation service for ad-hoc sensor networks," *SIGOPS Oper. Syst. Rev.*, vol. 36, no. SI, pp. 131–146, 2002.

[5] G. Bontempi and Y. L. Borgne, "An adaptive modular approach to the mining of sensor network data," in *Proc. SIAM International Conference on Data Miningg*, 2005.

[6] Y. L. Borgne and G. Bontempi, "Unsupervised and supervised compression with principal component analysis in wireless sensor networks," in *Proc. First International Workshop on Knowledge Discovery from Sensor Data, 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2007.

[7] Y. L. Borgne, S. Raybaud, and G. Bontempi, "Distributed principal component analysis for wireless sensor networks," *Sensors Journal*, vol. 8, pp. 4821–4850, 2008.

[8] W. K. Nicholson, *Linear Algebra with Applications*, 3rd ed. PWS Pulishing Company, 1995.

[9] A. Sharma and K. K. Paliwal, "Fast principal component analysis using fixed-point algorithm," *Pattern Recognition Letters*, vol. 28, no. 10, pp. 1151–1155, 2007.

[10] K. Hillol, W. Huang, S. Krishnamoorthy, P. Byung-Hoon, and S. Wang, "Collective principal component analysis from distributed, heterogeneous data," in *Proc. the 4th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD '00)*. Springer-Verlag, 2000, pp. 452–457.

[11] I. Jollife, *Principal Component Analysis*, 2nd ed. Springer, 2002.

[12] H. Kargupta, W. Huang, K. Sivakumar, and E. Johnson, "Distributed clustering using collective principal component analysis," *Knowl. Inf. Syst.*, vol. 3, no. 4, pp. 422–448, 2001.

[13] Y. L. Borgne, J. Dricot, and G. Bontempi, "Principal component aggregation for energy-efficient information extraction in wireless sensor networks," *Knowledge Discovery from Sensor Data*, pp. 55–80, 2008.