Efficient Energy Management and Data Recovery in Sensor Networks using Latent Variables Based Tensor Factorization

Bojan Milosevic, Jinseok Yang, Nakul Verma, Sameer S. Tilak, Piero Zappi, Elisabetta Farella, Luca Benini, Tajana S. Rosing

University of Bologna, UCSD, Qualcomm Inc.

ABSTRACT
A key factor in a successful sensor network deployment is finding a good balance between maximizing the number of measurements taken (to maintain a good sampling rate) and minimizing the overall energy consumption (to extend the network lifetime). In this work, we present a data-driven statistical model to optimize this tradeoff. Our approach takes advantage of the multivariate nature of the data collected by a heterogeneous sensor network to learn spatio-temporal patterns. These patterns enable us to employ an aggressive duty cycling policy on the individual sensor nodes, thereby reducing the overall energy consumption. Our experiments with the OMNeT++ network simulator using realistic wireless channel conditions, on data collected from two real-world sensor networks, show that we can sample just 20% of the data and can reconstruct the remaining 80% of the data with less than 9% mean error, outperforming similar techniques such as distributed compressive sampling. In addition, energy savings ranging up to 76%, depending on the sampling rate and the hardware configuration of the node.

Categories and Subject Descriptors
G.3 [Probability and Statistics]: Multivariate statistics; I.5.1 [Pattern Recognition]: Statistical models

Keywords

1. INTRODUCTION
Modern applications of WSNs typically require measuring several variables (such as temperature, humidity, etc.) for extended periods of time over a large area. To meet the application requirements, the design and deployment of a WSN has to carefully balance between two competing goals: (1) high spatio-temporal resolution to ensure the accuracy of the collected data, and (2) minimal energy consumption to maximize the network lifetime and limit node maintenance. The amount of data that a node collects and processes directly affects both its power consumption and the accuracy of the information obtained [8, 22].

Extending the uptime of a sensor node - especially when the nodes are deployed in difficult to access remote locations - is an active topic of research in WSNs. Common approaches try to enhance the battery life directly by harvesting energy from the environment and employing low-power hardware architectures [4, 5], or using improved wireless protocols and distributed computation for data processing [27]. More recently, researchers are optimizing the battery life indirectly by reducing the overall amount of sensed data [24]. Here, the data is selectively sampled according to a predetermined protocol, reducing the total amount of samples collected by the individual sensor nodes, thus minimizing the energy consumption [11]. To maintain an acceptable amount of total measurements, the missing data is inferred according to statistical models that capture how the data evolves. In addition to enhancing the battery life, these approaches are also able to estimate any lost or corrupted data, making them a popular choice.

In this paper we propose an energy efficient, data driven technique to estimate missing data within a heterogeneous sensor network. Our approach extends the standard latent variable factorization model - which typically considers only dyadic interactions in data - to multivariate spatio-temporal data, by applying tensor decomposition techniques [15]. The key advantage of using a latent variable model is that it provides a compact representation of the gathered data that can be used to recover the missing samples. In order to perform well under extreme sampling conditions, we extend the standard technique to explicitly incorporate the spatial, temporal, and inter-sensor correlations. This study focuses on the trade-off between the accuracy in recovering the missing data and the energy consumption when sensor nodes duty cycle to save energy. The proposed technique drastically reduces the amount of sampled data at
each node, thus allowing the nodes to spend more time in a low-power sleep mode and save energy. The lower amount of sampled data implies a lower amount of data to transmit from the node to a central gathering station, reducing also the power consumptions associated with the radio communications. To test our approach, in a realistic setting (e.g., realistic channel conditions) we simulated our sensor network deployments using the OMNeT++ [21] network simulator. We used two different deployments: one is an environmental monitoring WSN with weather stations displaced across the state of California [9], and the another is a smaller scale WSN used for a structural health monitoring application [1]. Our results show that we can effectively reconstruct the un-sampled data with under 9% error even when we are sampling only about 20% of the CIMIS dataset. The 3ENCULT dataset achieves a mean reconstruction error of 6.3% when using 4-hop networks. We compared our technique with a state-of-the-art approach based on Distributed Compressive Sensing (DCS) [13], which has a comparable approach and reconstructs individual signals from sub-sampled versions. The result shows how our approach is better in the case of higher percentage of missing samples, and with signals that show lower temporal correlations. We observed up to 50% of energy saving across the network of sensors, depending on the sampling frequency. Lower energy savings when using a longer sampling period are expected because the sensor nodes spend most of the time in a low power sleep state. Thus, the energy consumption within the sleep state dominates the overall savings.

2. RELATED WORK
Selectively sampling the data in order to improve the overall energy consumption is an active area of research. Various compression [6, 3], subsampling [11, 18] and data recovery techniques [16] have been proposed in the literature, based on application characteristics. Compressed Sensing (CS) has emerged as a powerful framework to summarize (compress) and recover missing values when the data is sparse. It has recently been applied in WSNs to reduce transmission bandwidth and improve the energy consumption [6, 3, 13]. In [6], for instance, a random basis is used to compress data gathered from temperature sensors. A feedback mechanism allows the network nodes to adapt their sampling rate dynamically while minimizing the data acquisition and transmission costs. The distributed compressed sensing (DCS) framework [13] exploits both intra-signal and inter-signal correlations through the concept of joint sparsity. The work in [9], presents a distributed algorithm, robust to increases in the number of network, that optimizes the transmission bandwidth via DCS. Research has also optimized the energy consumption by minimizing data acquisition and various subsampling schemes have been proposed [11, 18, 16]. The work in [11] treats sensor networks as a database, adopting an interactive sensor querying approach enriched with statistical modeling techniques. They demonstrate that such models, coupled with the appropriate selection of sensors, can help provide answers that are both more meaningful and, by introducing approximations with probabilistic confidences, significantly more efficient for computing. In [18] similar types of nodes are clustered and data is sampled from one node in each cluster at a time. The missing values are predicted using a linear combination of the sampled entries. A Bayesian approach to find patterns in data is used in [16]. The missing data is reconstructed by using these patterns. They use simple spatial and temporal correlations (last values and one-hop neighbors) to improve the final prediction. Latent variable and decomposition techniques have also been proposed in literature, for various applications. Matrix factorization learning of latent variables has been used for recovering missing data in sensor networks in [25]. Tensor decomposition techniques have been applied on WSNs only in [21], where the learned models are used to find the damages in a structural health monitoring application. Other applications of tensor decomposition techniques include data classification and recovery in multichannel electroencephalograms [10] or network analysis [2].

The previously presented algorithms only consider homogeneous sensor streams, dealing with one sensor at a time, and do not consider varying sampling costs across sensor types. In our work we correlate data from different types of sensors to estimate the missing samples. Our approach focuses on the multivariate nature of the collected data. It expands the tensor factorization techniques by employing spatio-temporal and intra-sensor correlations for more robust and better results than the existing methods. The DCS has a similar approach when considering the network nodes, but adopts different reconstruction techniques. We included a comparison with our work to highlight the benefits of a correlation-based approach, that has better reconstruction capabilities with lower percentages of sampled data.

3. LATENT VARIABLES AND TENSOR FACTORIZATION
Latent variable based factorization is a simple yet powerful framework for modeling data, and has been successfully applied in several application domains [17]. The main idea behind this framework is to model the large number of observed variables (the observed data) in terms of a much smaller number of unobserved variables (the latent variables). The latent variables are learned from the observed data and are used to estimate the missing samples. Our approach focuses on the multivariate nature of the collected data. More specifically, given some multivariate data that is collected by a heterogeneous WSN in a large field over time, we can naturally organize it in a three dimensional data array (or a 3-tensor). Each of the three dimensions corresponds to a different variate of a particular measurement (e.g. the time, the location and the sensor type associated with each reading). Once the data is organized in this way, we can then associate a low-dimensional latent variable with each unique location, time slice and sensor type. We can thus model a particular observation (at a given location, time and type) as a noisy combination of the associated latent variables. In many scenarios, a multiplicative combination of these latent variables is able to capture intricate dependencies in the data [15, 2]. The goal then is to learn a good set of latent variables (that is, find a factorization) that can efficiently represent our observed data.

3.1 Modeling Details
We now define formally our model. For each unique time instance \( t \), sensor type \( s \), and node location \( n \), we associate a unique \( K \)-dimensional vector \( a_{tsn}, b_{tsn} \) and \( c_{tsn} \), respectively.

These unobserved vectors are called the latent factors or
variables, and are assumed to control the location-, time- and sensor-specific interactions present in the observed data. Then, given a \( [T \times S \times N] \) tensor \( \mathcal{X} \) of sensor readings from \( S \) different sensor types collected at \( N \) different nodes and \( T \) different time instances, with possible missing entries, we model \( \mathcal{X} \) as follows. We assume that each reading \( x_{tsn} \) (reading at time \( t \), for sensor type \( s \), at node location \( n \)) is a noisy realization of the underlying true reading that is obtained by the interaction of the time specific latent variable \( a_t \), the sensor specific latent variable \( b_s \), and with the location specific variable \( c_n \). That is,
\[
x_{tsn} = \sum_{k=1}^{K} a_{tk} b_{sk} c_{nk} + \epsilon,
\]
where \( \epsilon \) is modeled as independent zero-mean Gaussian noise (\( \epsilon \sim N(0, \sigma^2) \)). Observe that under this model once all the latent variables are known, one can recover the true readings of all sensors at all locations and times. Thus the goal is to find the most predictive set of vectors \( a_t, b_s \) and \( c_n \) for all \( t = 1, \ldots, T, s = 1, \ldots, S, n = 1, \ldots, N \). Such a representation models the entire data size of \( [T \times N \times S] \) by just \([K \cdot (T + N + S)]\) modeling parameters. The choice of the free parameter \( K \) provides a key trade-off: a large \( K \) increases the number of modeling parameters and thus can help model the observed data exactly. But this lacks the capability on predicting unobserved/missing data due to overfitting. A small \( K \), on the other hand, escapes the overfitting problem, but the corresponding model lacks sufficient richness to capture salient data trends. The exact choice of a good \( K \) is typically application dependent and is derived empirically.

### 3.1.1 Learning the latent variables
Finding the optimal set of \( K \)-dimensional latent variables given the observations is equivalent of factorizing the given tensor into three matrices each of rank at most \( K \) [15]. Thus, assuming that all the data is known (that is, every entry in the tensor is observed), we can find the latent factors by employing the CanDecomp/Parafac (CP) tensor factorization [15]. This is simply a higher-order generalization of the matrix singular value decomposition (SVD), and decomposes a generic third order tensor in three matrix factors \( A, B, \) and \( C \). By restricting the ranks of each of the matrix factors to at most \( K \), yields the best rank \( K \) approximation. Algorithmically, the matrix factors are found by an alternating least squares approach (ALS) [2], which starts from a random initialization and iteratively optimizes one matrix factor at a time, while keeping the other two fixed.

This technique can be generalized to work with tensors that have missing entries. Since sensor nodes can periodically go offline due to duty-cycling or run out of energy (preventing all sensors on a node from collecting any data for an extended period of time), we need to extend our basic model to deal with data missing from multiple sensors or nodes, resulting in entries and rows of missing data in the collected tensor. In order to do well in this regime we extend the basic tensor factorization model to explicitly incorporate spatial, temporal and sensor specific information from neighboring observations by explicitly learning and enforcing the corresponding correlations.

### 3.2 Incorporating correlations
To successfully interpolate the sensor interactions to contiguously missing blocks of data, we need to explicitly model spatial, temporal and sensor-specific trends within each of our latent variables \( a_t, b_s \) and \( c_n \). Such trends ensure that the latent variables \( a_t \) and \( a_{t'} \) (respectively \( b_s \) and \( b_{s'} \), and \( c_n \) and \( c_{n'} \)) take similar values when times \( t \) and \( t' \) are “similar” (respectively sensors types \( s \) and \( s' \), and locations \( n \) and \( n' \)). Note that similarity can mean anything based on the context. For locations, it can mean that variables associated two locations that are close in distance should have similar characteristics, while for time, it can mean that variables associated with times that are same hour of the day or same day of the week should have similar characteristics. Here we take a data driven approach to infer the best notion of similarity using correlations directly computed from the data. We use a limited sample of data collected from the network to learn the correlations, which are then applied to the subsequent slot of data to reconstruct the unsampled values.

The similarity constraints are modeled in the same way for all the three sets of latent variables, and here we illustrate the case for the \( a_t \)'s. Since each \( a_t \) is a \( K \)-dimensional variable, let \( a_{tk} \) denote its \( k \)-th coordinate. We model \( a_{tk} \) (independently for each coordinate \( k \)) as
\[
a_{tk} = \mu_{ak} + \alpha_{tk}
\]
\[
\alpha_{tk} \sim N(0, \Sigma_k).
\]
Here \( a_{tk} \) represents the collection of all \( a_t \)'s (across \( t = 1, \ldots, T \)) in the \( k \)-th coordinate and \( \mu_k \) represents their mean value. The distributional constraint over \( \alpha_{tk} \) (as \( N(0, \Sigma_k) \)) enforces the similarity constraints via the \( T \times T \) covariance matrix \( \Sigma_k \). By changing the \( t, t' \) entry of \( \Sigma_k \) we can encourage/discourage the corresponding \( a_t \) and \( a_{t'} \) to take similar values – a high positive value at \( \Sigma_k(t, t') \) encourages a positive correlation, a high negative value encourages negative correlation, while a value close to zero does not encourage any correlation. We enforce similar constraints on \( b_s \) and \( c_n \) as well. To get the right similarity constraints \( \Sigma_k \), \( \Sigma_k \) and \( \Sigma_k \) (for latent variables \( a_t, b_s \) and \( c_n \)), we compute them from the data we are considering. For spatial similarity constrains, we computed the averaged pairwise Pearson correlation coefficients between data from different pairs of locations, and averaged over different time and sensors. We do the same to approximate inter-sensor and temporal similarities. We can have only one global correlation constraint along each of the dimensions.

### 3.3 Parameter learning
We can learn the underlying latent variables in a probabilistic framework using a maximum a posteriori (MAP) estimate. In particular, let \( \theta \) denote all the model parameters (\( i.e., \theta = \{(a_t), (b_s), (c_n), \sigma\} \)), then the optimum choice of parameters \( \theta_{MAP} \) given the data \( \mathcal{X} \) is obtained by:
\[
\theta_{MAP}(\mathcal{X}) := \arg\max_{\theta} \log p(\mathcal{X} | \theta) p(\theta)
\]
\[
= \arg\max_{\theta} \sum_{t,s,n} \log p(x_{tsn} | a_t, b_s, c_n, \sigma) + \sum_{k=1}^{K} \log p(a_k) + \sum_{k=1}^{K} \log p(b_k) + \sum_{k=1}^{K} \log p(c_k).
\]
The first term (the likelihood) takes the form of Eq. (1), and the other terms represent the priors for each latent variable and each one of them takes the form of Eq. (2). We take a
uniform prior over $\sigma$, the standard deviation of the residuals in Eq. (1) so it doesn’t explicitly show in the equation. This optimization does not have a closed form solution. Standard gradient based techniques can be used to get a locally optimal solution. Here we can do an alternating hill-climb approach by optimizing the value of one variable while keeping all others fixed to get a good solution. As for the standard factorization technique, this iterative approach uses a random initialization of the latent variables.

4. HARDWARE, NETWORK AND POWER MODELS

To evaluate the proposed technique we use real-world data from two sensor networks, with different hardware configurations and spatio-temporal resolutions. The composition of the two networks is summarized in Table 1.

Our first data set comes from an environmental monitoring WSN from the California Irrigation Management Information System (CIMIS) [9]. It is a program that manages a network of 232 automated weather stations (128 functioning at the moment), displaced across the state of California. Each station provides hourly readings of 12 different measurements from its embedded sensors. The second data set comes from a case study of the 3ENCULT European project [1], deployed by our research group. This is a structural health monitoring application where a network of 23 low-power sensor nodes is deployed across the three floors of a historic building at the University of Bologna.

The two data sets have different sampling periods (one hour for CIMIS and ten minutes for 3ENCULT) and different spatial coverage and distribution (state-wide coverage for CIMIS and indoor coverage of a 3-store building for 3ENCULT). Table 2 summarizes the variables reported by each station for the two datasets. Even if some variables are shared among the two datasets, the network characteristics, the nature of the captured signals and the resulting correlations are different because of the differences in the sensed environment. As an example, we can observe the Figure 1 where a block of 512 samples of temperature readings from the two data sets is reported.

### 4.1 Hardware

We used the hardware configuration of the nodes in the two data sets to estimate each node’s power consumption. For the environmental monitoring stations in the CIMIS dataset we used the Vaisala WHT250 sensor station [23] which has detailed power consumption information available. This type of station is used at several locations in the CIMIS network. Since the WHT250 does not embed any radio, we assume that the radio communication is performed through a commercially available Zigbee wireless transceiver (XBee PRO [12]). The node employed in the 3ENCULT network is a custom wireless node (W25TH) that is based on an NXP system on chip module with an 32-bit RISC processor and a 24GHz IEEE802.15.4-compliant transceiver. Sensors include a Sensirion SHT21 for temperature and humidity, and a BH1715 for Light. Timing reference and power information is obtained either using the values reported in the datasheet (CIMIS) or direct measurements using a GPIO trigger connected to an oscilloscope (3ENCULT). Data on power consumption of the various subsystems is not reported due to lack of space, but the reader can refer to Tables 3 and 4 for key power parameters, including the measured currents in the case of the 3ENCULT nodes.

### 4.2 Network model

In a sensor network we can consider that each node samples the signals for a period of time $T$, called acquisition period, ideally gathering $N = T \cdot f_s$ number of samples at a $f_s$ sampling frequency, before sending the data to the collection point. If each node adopts a sub-sampling policy with an under-sampling ratio $\alpha$ then the number of samples actually gathered by the node is $M = \alpha N$.

The under-sampling pattern is locally generated by each node using its own ID and the timestamp as seed for the randomization of the sampling pattern, which has inter-measurement intervals that are always multiple of the sampling period $T_\delta = T/N$.

We used the OMNET++ network simulator to simulate the packet loss characteristics of a realistic wireless channel [20]. We leveraged the commonly used log-normal path loss model, which is defined as:

$$ P_r(d)[dBm] = P_t[dBm] - PL(d_0) - 10nlog(d/d_0) - X_\sigma $$

where $P_r(d)$ is the received signal strength (in dBm), $P_t$ is transmitted power, $PL(d_0)$ is the measured path loss at the reference distance $d_0$, $n$ is the path loss exponent, and $X_\sigma$ is a normal random variable with a zero mean and variance $\sigma$ (in dB).

The path loss exponent measures the rate at which the received signal strength decreases with distance [19], while the term $X_\sigma$ models the shadowing effect, which introduces signal variations because of various types of artifacts in the
4.3 Sensor node power models

We introduce an architecture level power model to evaluate the energy consumption of the node when the subsampling parameters are changed. Using this power model with data from real hardware and measurements, we can easily evaluate how changing the parameters influences the energy consumption and the lifetime of the network.

Starting from the assumption reported in the previous section, the average energy consumption in each period of duration $T_k$, for a sub-sampling factor $\rho$, is:

$$E_k = \rho (E_{\text{setup}} + E_{\text{sample}} + E_{\text{store}}) + E_{\text{sleep}} + \frac{1}{N-1} (E_{\text{nv}} + E_{\text{send}})$$

(4)

where $E_{\text{sleep}}$ is the energy spent in sleep mode, $E_{\text{setup}}$ is the energy used for waking up and setting up the device, $E_{\text{sample}}$ is the energy for sampling each sensors, $E_{\text{store}}$ is the energy used to send the acquired data, $E_{\text{nv}}$ is the energy to store the acquired sample in non volatile memory and $E_{\text{nv}}$ is the energy spent during the recovery of the data from non volatile memory. We used the hardware configuration of the nodes in the two data sets to estimate each node’s power consumption.

The energy consumption of the environmental monitoring sensor stations is summarized in Table 3. This data is obtained from the components’ data sheets, considering the average consumption for each operation, and using the World Meteorological Organization (WMO) specifications for environmental data gathering [26]. The wind measurements (speed and direction) need to be reported in 1 minute increments, thus costing a lot of energy. Pressure, temperature and humidity sensors have the same power consumption (9.6mW) as they are part of the same module, but each sensor has to be read individually. With an 1 hour sampling period, the energy consumed at each interval is 5.62J, 75% of which is spent in the sleep mode. Of the active energy, 96% is consumed for the sampling of the sensors, 3.5% for data transmission and 0.5% by the CPU. Our model considers all the contributions even if the sampling energy is the dominating component in this scenario.

The energy consumption of the nodes of the 3ENCULT network is summarized in Table 4. In this case the radio transmission represents the main source of energy consumption. There is 24.1mJ of energy per each 10 minutes sampling period, of which almost 82% is for the sleep mode. Of the active power, 86% is consumed by the radio transmission of the acquired packets, 13% by the CPU and 1% by the sensors.

<table>
<thead>
<tr>
<th>Sensor or State</th>
<th>Power [mW]</th>
<th>Time [s]</th>
<th>Energy [mJ]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sleep (all node)</td>
<td>1.2</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CPU Active</td>
<td>42.9</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Wind speed &amp; direction</td>
<td>20</td>
<td>60</td>
<td>1200</td>
</tr>
<tr>
<td>Pressure</td>
<td>9.6</td>
<td>5</td>
<td>48</td>
</tr>
<tr>
<td>Temperature</td>
<td>9.6</td>
<td>5</td>
<td>48</td>
</tr>
<tr>
<td>Humidity</td>
<td>9.6</td>
<td>5</td>
<td>48</td>
</tr>
<tr>
<td>Rain</td>
<td>0.84</td>
<td>10</td>
<td>8.4</td>
</tr>
<tr>
<td>Data transmission</td>
<td>158.4</td>
<td>0.05</td>
<td>8</td>
</tr>
</tbody>
</table>

Table 3: Power consumption of the different components of an environmental station used by CIMIS. The energy consumption refers to a single sampling event or a single packet transmission.

<table>
<thead>
<tr>
<th>Sensor or State</th>
<th>Current [mA]</th>
<th>Power [mW]</th>
<th>Time [ms]</th>
<th>Energy [mJ]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sleep (all node)</td>
<td>0.01</td>
<td>0.578</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CPU Active</td>
<td>8.08</td>
<td>24.7</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Temperature</td>
<td>0.3</td>
<td>0.495</td>
<td>0.002</td>
<td>0.001</td>
</tr>
<tr>
<td>Humidity</td>
<td>0.3</td>
<td>0.495</td>
<td>0.002</td>
<td>0.001</td>
</tr>
<tr>
<td>Light</td>
<td>0.3</td>
<td>0.495</td>
<td>0.002</td>
<td>0.001</td>
</tr>
<tr>
<td>Data transmission</td>
<td>18</td>
<td>102.3</td>
<td>0.05</td>
<td>5.115</td>
</tr>
</tbody>
</table>

Table 4: Power consumption of the different components of the sensor node used in 3ENCULT.

In both cases the influence of the energy spent in the sleep mode increases with an increase of the sampling period, since the node spends more time waiting for the next interval. With longer periods, even aggressive duty cycling policies will have a smaller benefit in terms of energy saving, since they reduce only the active energy spent for sampling and data transmission.

5. EXPERIMENTAL RESULTS

5.1 Data preprocessing

Our statistical model treats data from each heterogeneous sensor equally. Since raw data from different sensor types are at widely different scales (e.g. the temperature ranges from about 10 to 40°C, while relative humidity measurements range from 0 to 100%), we preprocess all the variables to ensure they have zero mean and unit variance. This normalization brings all measurements to the same scale and allows us to apply our multivariate tensor factorization technique. The normalized prediction can easily be translated back into the original scale by rescaling and adding back the mean value. The reconstruction error is evaluated in the original scale for each variable, adopting the Normalized Root Mean Squared Error (NRMSE) defined as:

$$NRMSE = \sqrt{\frac{1}{T} \sum_{t=1}^{T} (\hat{x}_t - x_t)^2} / (x_{\text{max}} - x_{\text{min}}),$$

where $x$ and $\hat{x}$ are respectively the actual and predicted data for each variable, $x_{\text{max}}$ and $x_{\text{min}}$ are the maximum and minimum values for the same variable.

5.2 Tensor factorization

Here we evaluate the effectiveness of our tensor-based latent variable factorization technique to model our datasets. In our approach, the data collected from the sensor network is organized in a $[T \times S \times N]$ tensor $X$ and is factored in three matrix factors $A$, $B$ and $C$ of size $[T \times K]$, $[S \times K]$ and $[N \times K]$. This leads to a model with $K(T + S + N)$ parameters for a dataset with $T \cdot S \cdot N$ entries.

The latent variables dimensionality, $K$, regulates the com-
plexity of the model. To evaluate the impact of $K$, different models with increasing complexity (increasing values of $K$) were learned from the data, using the standard CP tensor factorization technique. The whole dataset is reconstructed using the learned model and the $NRMSE$ between the real data and the reconstructed one is evaluated. This operation is repeated for different fractions of sampled data $\alpha$ used to learn the model. The $NRMSE$ shows how well the collected data can be summarized by a few latent variables, thus exploring the compression capabilities of our approach. The results for the two datasets are illustrated in Figure 2. In both cases we used blocks of 512 samples for the temporal duration, which were sub-sampled to test the algorithm. We can see a saturation effect on the quality of the reconstruction, with the increase of $K$. The exact choice of this parameter is application dependent and depends on the dimension of the network and the number of sensors. In our case $K$ was set to 24 in the CIMIS and 12 in the 3ENCULT case.

5.3 Data recovery

In order to save energy, sensor nodes can apply aggressive duty cycling strategies, collecting just a portion of the original data. When applying a duty cycling policy, a node avoids sampling its sensors, resulting in an entire rows of missing entries in the data tensor. To analyze the ability of the proposed method to reconstruct the missing samples, we removed increasing percentages of rows from the collected data. The remaining data (training data) was used to learn our latent variable model, while the removed entries (testing data) were used to evaluate the recovering capabilities of the algorithm.

Figure 3 shows the reconstruction error of the missing data ($NRMSE$) as a function of the fraction of the sampled data ($\alpha$) for the two analyzed datasets. We compare the two implementations of the tensor factorization algorithm: (1) the standard factorization technique (CP) that does not include correlations from the data, and (2) our approach that does incorporate the temporal, spatial, and inter-sensor correlation, where the variables are learned using the MAP estimation. In both cases we use a block of 512 samples from the whole network and set the dimension of the latent variables as indicated in the previous section. The correlations are learned directly from the data, using a block of 64 samples, taken just before the data blocks are considered for the reconstruction evaluation. The mean $NRMSE$ across all the variables is reported on the testing data for the two datasets.

In order to investigate the effect of the wireless transmission, we simulate the network conditions with different path loss exponents ($\eta$) and shadowing variances ($\sigma$) as described in Section 4.2 with the log-normal path loss model. Figure 3.(a) shows that in the CIMIS case our approach achieves a reconstruction error below 9%, with up to 80% of the missing samples under the worst wireless channel condition (i.e. $\eta=2.5$, $\sigma=3$). With the same sampling policy, Figure 3.(b) shows that for the 3ENCULT dataset we achieve a mean reconstruction error of 6.3%, when using a 4-hop network. The results show that the number of the network hops has a negligible effect on the reconstruction accuracy when the wireless channel conditions are stationary over the deployed area. Increasing the number of network hops by four de-
increases the reconstruction accuracy by just 1%. Our technique consistently outperforms the standard tensor factorization approach with up to 90% of the missing samples. If there are more than 90% missing samples, our approaches still achieve less than 20% error. The results outlined above are the average reconstruction errors among all sensors. If we look at the individual sensors, we see that the single variables exhibit different recovering proprieties. The ones characterized by strong correlation proprieties and smooth temporal transitions, such as temperature and solar radiation, are better reconstructed compared to the ones that have high variance or little periodicity, such as wind speed and direction.

5.4 Comparison with compressive sensing
To compare our approach with a state of the art technique, we used the same data with a compressive sampling reconstruction algorithm. To match our scenario, where just a portion of the desired data is collected due to duty cycling policies, we employed a sparse measurement matrix as in the Distributed Compressive Sampling (DCS) approach illustrated in [13]. A Discrete Cosine Transform (DCT) is used as the basis with a Basis Pursuit solver available in the SparseLab Matlab Toolbx to independently reconstruct the signals. The results of the comparison with the 3ENCULT data set are shown in Figure 4. Our technique, labeled MAP, performs better when a smaller portion of data is collected, thanks to the use of the inter-node correlations and the contribution from the whole network to reconstruct the data. For the reconstruction of the humidity signals we have comparable results for the two techniques. For the temperature our approach is better for smaller amounts of collected samples, gaining 6% of normalized reconstruction error at 20% of collected samples. In particular, the DCS algorithm has problems in the reconstruction of the light signal, which shows less temporal correlations and has higher variance. Indoor light patterns often present sharp changes due to artificial illumination, which is not well managed by the adopted compressed sensing approach. In this case we achieve more than 10% of gain in the NRMSE, when collecting 20% of the samples.

5.5 Energy savings
Using the energy characterizations of the nodes from Tables 3 and 4, we can estimate the average amount of energy savings associated with different sampling policies. Figure 5 shows energy savings for various combinations of sampling periods and duty cycling rates in the two case studies. Different sampling policies are compared to the case when all the data is sampled. Regardless of the type of hardware used in the two scenarios, we determine a similar result, showing that our algorithm can get large energy savings in a wide range of applications. Aggressively duty cycling on a data-set that samples the environment every minute yields significantly higher energy savings (76%) than the one which only samples every hour (20%). This is expected because with lower sampling frequencies, the sensor nodes spend most of the time in a low power sleep state. Thus, the energy consumption within the sleep state dominates and the overall power consumption is less influenced by the sampling policy.

6. CONCLUSION
This work presents a latent variable model for energy efficient data collection in wireless sensor networks. This approach takes advantage of the multivariate nature of the data collected by a heterogeneous sensor networks, and uses tensor factorization methods combined with correlation.
Our model’s performance allows us to employ aggressive duty cycling strategies on the sensor nodes for better energy conservation. Each node is able to save energy by sampling just a portion of the desired data. The available data is then used to learn a latent variable model and reconstruct the missing samples. The proposed technique does not require any kind of data processing on the sensor node, which only adopts the desired duty cycling policy and sends the collected samples.

Experiments with real-world sensor networks show that our algorithm can maintain a low mean reconstruction error, below 9%, with up to 80% of missing samples, even under harsh transmission channel conditions. This permits energy savings ranging from 20% to 80% depending on the sampling rates in the two cases dataset. When compared to an existing state of the art approach, such as the distributed compressed sensing, our approach shows 2 times better reconstruction performance with 80% of missing samples. Our approach takes advantage of inter-node correlations across the network, so has advantage when signals have low correlations.

Acknowledgments
This work has been funded by the NSF project Citisense (grant CNS-0932403), NSF OCI Award 1219504 and the European project 3ENCULT (Grant agreement 260162). Additionally it was supported in part by TerraSwarm, one of six centers of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA. We also thank the Qualcomm Research center for their support.

7. REFERENCES