Latent Variables Based Data Estimation for Sensing Applications

Nakul Verma, Piero Zappi, Tajana Rosing

Computer Science and Engineering, University of California San Diego
9500 Gilman Dr., CA 92093, USA
{naverma,pzappi,tajana}@ucsd.edu

***Abstract*—Recovering missing sensor data is a critical problem for sensor networks, especially when nodes duty cycle their activity or may experience periodic downtimes due to limited energy. Fortunately, sensor readings are often correlated across different nodes and sensor types. Among state-of-the-art statistical data estimation techniques, latent variable based factor models have emerged as a powerful framework for recovering missing data. In this paper we propose the use of latent variable models to estimate missing data in heterogeneous sensor networks. Our model not only correlates data across different sensor locations and types, but also takes advantage of the temporal structure that is often present in sensor readings. We analyze how this model can effectively reconstruct missing sensor data when the individual sensor nodes have to duty-cycle their activity in order to extend network lifetime. We evaluate our model on a real life sensor network consisting of 122 environmental monitoring stations that periodically collect data from 13 different sensors. Results show that our proposed model can effectively reconstruct over 50% of missing data with less than 10% error.**

I. Introduction

Wireless Sensor Networks (WSNs) is an enabling technology for Ambient Intelligence (AmI) and ubiquitous computing applications [1]. The availability of a large number of low-cost, self organizing, unobtrusive sensor nodes that are able to collect and process large amounts of data allows the development of countless applications in several fields of human activity [2], [3]. In a general setting, sensor nodes sense the phenomenon of interest through a set of heterogeneous sensors. This information is typically processed locally to filter out the noise and extract the relevant features, and is forwarded to a back end infrastructure for further analysis and final presentation to the end user [4], [5].

Whenever deploying a WSN, one has to balance between two conflicting goals: (1) collect data frequently enough with reasonable accuracy in order to meet the application expectations, and (2) minimize energy consumption in order to maximize network lifetime. The amount of data that a node collects and processes directly affects its power consumption and the overall network lifetime, making power management a key issue in WSNs. For example, both for high performance wireless embedded systems that generate and process high volume of data (like the SHiMmer node for structural health monitoring [6]), and low power, low cost motes that sample few bytes of data per minute (like the Telos mote [7]) we see a clear relationship between the node energy consumption and the volume of data generated (see Table I).

Extending a sensor node's uptime—especially when the nodes are deployed in difficult to access remote locations—is of fundamental interest in power management. Common approaches directly try to enhance battery life by adding solar panels and implementing low-power hardware architectures [8], [9], [10], or using improved wireless protocols and using distributed computation for data processing [11], [12]. More recently, researchers are exploring an alternate set of approaches that optimize the battery life indirectly by systematically reducing the overall amount of sensed data [13], [14]. Here, the data is *selectively* sampled according to a predetermined protocol. This reduces the total amount of samples collected by the individual sensor nodes, thus minimizing the energy consumption. To maintain an acceptable amount of total measurements, the unsampled data is *inferred* according to some statistical model that captures 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 work, we propose a data driven statistical model to estimate the missing sensor data. Our model is based on latent variable factor model framework that is popular in the statistics and machine learning communities. We extend this framework to explicitly incorporate temporal relationships in data to adapt to situations where extreme amounts of data are missing. Our model correlates data across the heterogeneous sensors through a set of *latent* variables. These variables help provide a compact representation of the sampled data, and are effective in recovering any missing data. To evaluate the effectiveness of our proposed model, we use it to predict missing entries on a real-world dataset from environment monitoring domain under two important scenarios: (1) nodes duty-cycle their sensor sampling in order to extend their lifetime, and (2) a node experiencing extended period of inactivity due to a fault or lack of energy.

We observe that the performance of our model degrades gracefully with extreme amounts of missing data. Our experiments show that we can achieve less than 10% of average reconstruction error when more than 50% of data is missing in duty-cycling experiments. Furthermore, we are able to reconstruct a continuous block of over 5 days of missing with less than 10% average reconstruction error.

The rest of the paper is organized as follows. Section II presents the related work on data estimation techniques.

We introduce latent variable modelling in Section III, with our extension to the basic model that includes the temporal correlation between latent variables in Section IV. In Section V, we evaluate the efficacy of the proposed solution on an environment monitoring dataset. We conclude the paper with a brief discussion in Section VI.

TABLE I
POWER CONSUMPTION OF TWO SENSOR NODES (SHIMMER [6] AND TELOS [7]) AS A FUNCTION OF THE AMOUNT OF DATA GENERATED

Node	Sampling Rate	Average Power (mW)
SHiMmer	4 sample/hour	226.9
	0.5 sample/hour	28.6
Telos	60 sample/hour	3.47
	6 sample/hour	0.4

II. RELATED WORK

Several works address the problem of estimating missing values in data streams. CARM [13] and FARM [14], for instance, are techniques that build association rules for frequently occurring patterns in data. They use these rules to find similar patterns in missing regions of the data and impute the missing values. Although these techniques are fast and effective in correlating sensor values from a given sensor node, due to their rule based nature, they can only deal with discrete valued data (e.g., temperature is Low/Medium/High) rather than continuous valued data (e.g., temperature is $28.5°$C).

Garcia-Laencina et al. [15] recently proposed a k-nearest neighbor based imputation procedure that uses a weighted average of 'close' neighbors to estimate the missing values. They use a well-grounded notion of closeness based on the information theoretic concept of mutual information. However, due to the simple nature of weighted averaging, it is difficult for such techniques to take into account complex cyclical temporal trends and perform any future data prediction or interpolate chunks of missing data due to, say, node failure.

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 for reducing transmission bandwidth and improve the quality of sensing [16], [17]. In contrast to our work, [16] and [17] assume that all data is available from all sensors during transmission. The compression comes from the choice of the random basis (or dictionary) used to transmit the data. Thus, nodes must still sense periodically the environment and energy saving is achieved only by reducing the amount of data sent through the wireless channel. Ling and Tian [17] also explore recovering binary valued variables when the sensors follow a random sleeping schedule, but their technique can not be extended to real values.

Perhaps the most closely related technique to our work is the SPIRIT system [18]. This system is also a latent variable based technique that is used for data interpolation by summarizing a large number of data streams into a small, manageable number of latent variables. For example, they are able to summarize the streams from 8 temperature sensors using only 2 hidden variables with reconstruction error below $0.1°$C (about 0.4% of the absolute temperature value). While their method highlights several benefits of latent variable models which they demonstrate by performing select experiments on three model datasets, they do not focus on several practical issues present in real-world WSN data. They, for instance, do not explicitly model or exploit cyclic temporal trends in data nor do they evaluate the model's effectiveness on recovering large chunks of missing data when, say, a sensor node is down due to power/network failure and *none* of the sensors on the node can collect any data. For example, they evaluate the ability of SPIRIT to reconstruct a block of missing data of 5 hours using all the previous history. Results are provided through a qualitative graph that shows large errors during this time frame. In contrast, our work provides a quantitative analysis of the reconstruction of several days of inactivity. Our approach show an RMSE smaller than 10% even for blocks of missing data of 5 days or more. Finally, SPIRIT considers only homogeneous sensor networks, while here we address the more general case of heterogeneous sensor networks.

III. LATENT VARIABLES BASED FACTORIZATION

Latent variable based factorization is a simple yet powerful framework for modeling data, and has been successfully applied in several application domains [19], [20]. The main idea behind this framework is to model the large number of observed variables (the observed data) in terms of much smaller number of unobserved variables (the *latent variables*). These latent variables help model complex interactions between the observed variables through a simple interaction between the latent variables. In this work we extended the standard technique to incorporate the temporal relationship between successive data samples.

A. Latent Variable Model for Heterogeneous Sensor Networks

We model the data collected from heterogeneous sensor networks by associating a low-dimensional latent variable $u_{x,t}$ with each sensor node x at each time instance t. These latent variables will help encode the complex interaction between the heterogeneous sensors on the same node. The sensor specific observed reading at node x and time t for a particular sensor type j is obtained by combining the variable $u_{x,t}$ with sensor-type specific latent variable v_j.

More formally, given a $m \times n$ matrix R of sensor readings from n different sensor types collected at m different nodes/times with possible missing entries, we model R as follows. We assume that each reading $R_{(x,t),j}$ (reading at node x/time t for sensor type j) is a noisy realization of the underlying true reading that is obtained by combining the associated node/time specific latent variable $u_{x,t}$ with sensor-type specific latent variable v_j. That is,

$$R_{(x,t),j} = u_{x,t} \cdot v_j + \epsilon. \quad (1)$$

Here, each latent variable $u_{x,t}$ and v_j is assumed in $\mathbb{R}^k$ (typically, $k < \min\{m, n\}$), and ϵ is modeled as independent zero-mean Gaussian noise ($\epsilon \sim \mathcal{N}(0, \sigma^2)$).

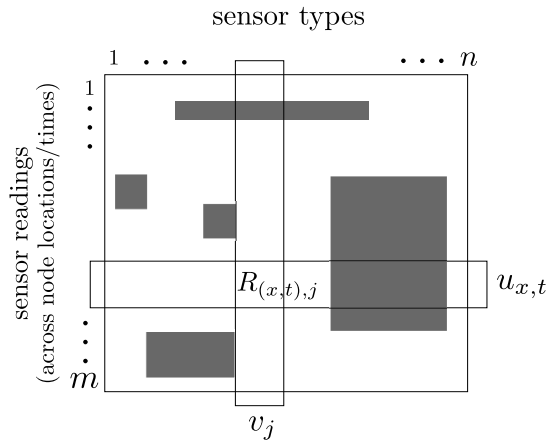

Fig. 1. The $m \times n$ data matrix R of sensor readings from n different sensor types collected at m different node/times with missing entries (shaded regions). Each column represents readings from a different sensor type, and each row represents readings from a sensor node at a particular time. Shaded regions represent missing entries in the data matrix.

The goal is to learn the underlying latent factors $u_{x,t}$ and v_j for all nodes/times x, t and sensors j, even when several entries in the matrix R are missing. The compact k-dimensional representation of the latent variables enables us to recover the expected reading at a missing node/time x', t' for sensor j' just by observing a few entries. See Figure 1.

B. Properties of a Factor Based Representation

Note that we have modeled our $m \times n$ data matrix R as:

$$R = UV + \text{noise},$$

where U is a $m \times k$ matrix (the collection of m node/time specific $u_{x,t}$'s) and V is a $k \times n$ matrix (the collection of n sensor specific v_j's). Observe that such a representation models the entire data of size mn by $mk + kn = k(m+n)$ modeling parameters. The choice of the free parameter k provides a key trade-off: a large k ($\approx O(m)$ or $\approx O(n)$) can help model the observed data exactly, but lacks capability on predicting unobserved/missing data due to overfitting. A small k ($\ll \min\{m, n\}$), 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.

A key limitation of this model in its current form is that it cannot recover a row if the data if the *entire* row is missing. This is primarily because one cannot learn the associated latent variable $u_{x,t}$ since there is no data (in the row) to constrain the choice for picking a good $u_{x,t}$. Since sensor nodes can periodically go offline due to duty-cycling or low power (preventing all sensors on a node from collecting any data for a period of time), we need to extend our basic model to deal with data missing from *all* sensors at multiple rows. We do so by explicitly incorporating information from neighboring rows (readings taken from the same node at close by time intervals) by enforcing 'temporal smoothness' across our $u_{x,t}$'s.

IV. Temporal correlation modeling

To successfully interpolate the sensor interactions to missing blocks of data, we need to explicitly model temporal trends between different $u_{x,t}$'s associated with the same location x. Such temporal smoothness ensures that the latent variables $u_{x,t}$ and $u_{x,t'}$ take similar values when the times t and t' are "similar", giving the ability to systematically extrapolate the latent variable $u_{x,t}$ to the missing rows and predict the associated sensor readings. We model this interaction by the following constraint. Since $u_{x,t}$ is a k-dimensional variable, let $u^d_{x,t}$ denote its d^{th} coordinate. Let $u^d_{x,:}$ represent the collection of $u^d_{x,t}$'s for different t's, then we model $u^d_{x,:}$ (independently for each coordinate d) as

$$\begin{aligned} u^d_{x,:} &= \mu^d_x + \alpha^d_:, \\ \alpha^d_: &\sim \mathcal{N}(0, \Sigma). \end{aligned} \tag{2}$$

Here μ^d_x is the base 'temporal' mean value of the latent variables $u^d_{x,t}$ (for the d^{th} coordinate) at a fixed node location x. The distributional constraint over $\alpha^d_:$ (as $\mathcal{N}(0, \Sigma)$), enforces the temporal similarity via the covariance matrix Σ. Here we set the individual entries of Σ via a temporal similarity function K, that is, $\Sigma_{t,t'} := K(t, t')$ (see also [21]). The exact notion of similarity, i.e. $K(\cdot, \cdot)$, is typically application dependent; for sensor applications where data has cyclical patterns, we shall use $K(t, t') := \exp\left\{-\left(\frac{\sin(|t-t'|\pi/P)}{h_1}\right)^2 - \left(\frac{|t-t'|}{h_2}\right)^2\right\}$, where P is the period length (e.g. 24 for hourly measurements) and h_1, h_2 are the bandwidth parameters that control the neighborhood size that can influence a particular reading (typically chosen via cross validation).

Note that when times t and t' are close, the similarity $K(t, t')$ takes on a high value. High covariance forces the corresponding $u_{x,t}$ and $u_{x,t'}$ to be similar, giving us the necessary tools for a systematic interpolation across time. The temporal smoothness constraint on $u_{x,t}$'s is also known as a prior over the $u_{x,t}$'s.

A. Learning the Latent Variables

We can learn the underlying latent variables $u_{x,t}$ and v_j, by *maximum a posteriori* (MAP) estimate. In particular, let θ denote all the model parameters (i.e. $\theta := \{\{u_{x,t}\}, \{v_j\}, \sigma\}$). Then, the optimum choice of parameters θ_{MAP} given the data R is obtained by

$$\begin{aligned} \theta_{\text{MAP}}(R) &:= \operatorname*{argmax}_{\theta} \ \underbrace{p(R|\theta)}_{\text{likelihood}} \ \underbrace{p(\theta)}_{\text{prior}} \\ &= \operatorname*{argmax}_{\{\{u_{x,t}\},\{v_j\},\sigma\}} \sum_{x,t,j \in \text{observed}} \log p(R_{(x,t),j} | u_{x,t}, v_j, \sigma) \\ &\qquad + \sum_{x \in \text{nodes}} \sum_{d=1}^{k} \log p(u^d_{x,:}). \end{aligned}$$

The first term (the likelihood) takes the form given by Eq. (1), and the second term (the prior) takes the form given by Eq. (2). Here we take uniform priors over $\{v_j\}$ and σ (independent of $\{u_{x,t}\}$), so they don't explicitly show up in the prior term.

This optimization does not have a closed form solution. Standard gradient based techniques can be used to get a locally optimal solution.

V. Experimental Results

A. System description

We evaluate the proposed model on an environment monitoring sensor network called the CIMIS network [22]. The California Irrigation Management Information System (CIMIS) is a program in the Office of Water Use Efficiency at the California Department of Water Resources that manages a network of 232 automated weather stations in the state of California. Each station provides hourly reading of 13 different measurements from its embedded sensors (see Table II).

For our experiments, we used one year worth of data (June 2010 - May 2011) and only considered the subset of sensor nodes (122 of 232) that were live continuously during this period.

TABLE II
Sensors list

Sensors	Unit
Precipitation	mm
Solar Radiation	W/m^2
Net Radiation	W/m^2
Air Temperature	$^\circ C$
Soil Temperature	$^\circ C$
Relative Humidity	$\%$
Wind Speed	m/s
Res Wind	m/s
Wind Direction	$0-360$
Wind Direction STD	$0-360$
Reference Evapotranspiration	mm
Vapor Pressure	kPa
Dew Point Temperature	$^\circ C$

We assume that each node periodically samples its sensors and wirelessly sends this data (with possible missing values) to a back end server where the data reconstruction is done. Since limited amount of data should be sent to the back end server, the nodes rely on state-of-the-art low power wireless communication modules (like the transceiver used in [7]).

Several factors affect a node's power consumption. Even if the actual power consumption is a function of the hardware used, for a wide range of systems the amount of data generated, processed and routed plays an important role, as shown in Table I. As a consequence, by reducing the number of samples from the sensor we are able to extend a node's lifetime. Since this comes at the expense of lower accuracy in monitoring, we mitigate this loss by our modeling technique.

B. Data Preprocessing

Our technique treats data from each heterogeneous sensor equally. Since raw data from different sensor types are at widely different scales (the temperature ranges from about 10 to 40°C, while vapor pressure ranges from about 0.5 to 3 kPa.), we preprocess the data to have zero mean and unit variance. This normalization brings all measurements to the same scale and helps in an effective prediction. The normalized prediction can easily be translated back into the original scale by rescaling and adding back the mean value.

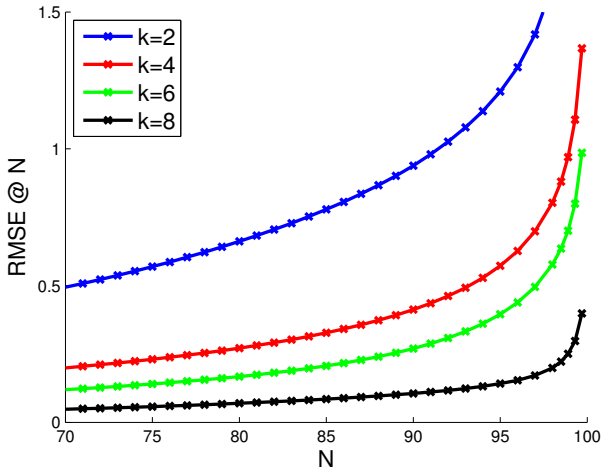

Fig. 2. Average reconstruction error for CIMIS dataset as one varies the model parameter k. (x, y) pair on a curve is read as: x percentage of the data matrix can be reconstructed with error at most y (normalized) units.

Even though the normalized scale is good for predicting heterogeneous data, the scale is difficult to interpret to assess the quality of prediction. If we have an average prediction/reconstruction error of δ (normalized) units, it is unclear how to translate this to percentage error. To ease interpretability, we assume that the data follows a Gaussian distribution. Now, since the measurements are normalized to have unit standard deviation, we know that about 95% of the data falls between ± 2 standard deviations. So, noting that most of the data spreads across four units, δ unit of normalized reconstruction error gives us $\frac{\text{error}}{\text{spread}} \times 100 = \delta/4 \times 100$ percentage error. For preciseness, we always report the normalized error, with references to the corresponding percentage errors where appropriate.

For all the experiments we shall use the following parameter settings: $h_1 = 0.6$, $h_2 = 100$, and $P = 24$ (see Section IV for details).

C. Compressibility Experiments

Our first set of experiments explore how well our latent variable technique models the given dataset. For these experiments we use the entire dataset without any missing entries and try to fit the model for different values of the free parameter k (see our discussion in Section III-B). The goal is to understand how well can the entire dataset be summarized by a few latent variables, thus exploring its compressibility. The experiments serve as an upper limit to the prediction accuracy that we expect for various choices of k.

Figure 2 shows the data estimation accuracy as one varies k. Each curve shows the reconstruction error (in normalized units) as a function of percentage of the data matrix for a

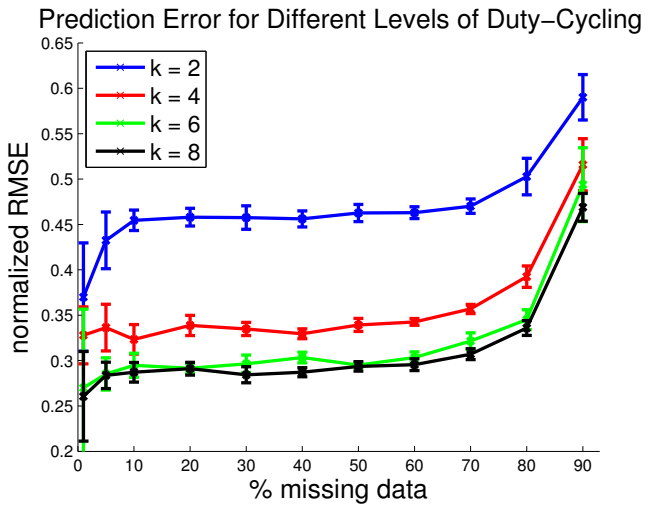

Fig. 3. Reconstruction error (in normalized units) of missing data for various levels of duty-cycling. Different curves represent the reconstruction errors associated with different values of the model parameter k.

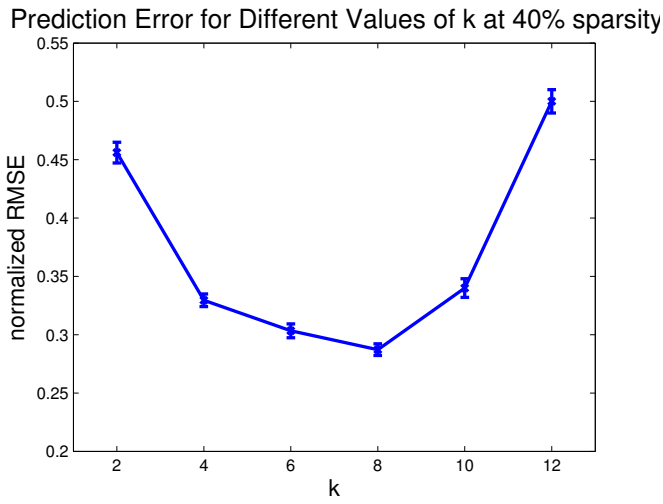

Fig. 4. Reconstruction error (in normalized units) for different values of the model parameter k for 40% sparsity (duty-cycling) level.

different choice of k. That is, (x, y) pair on the curve is read as: x percentage of the data matrix can be reconstructed with error at most y (normalized) units.

Observe that for $k = 4$ (which correspond to a compression ratio of 0.308), the curve shows that 90% of the data can be constructed with error less than ~ 0.4 units. According to our interpretation heuristic, this amounts to about 10% of reconstruction error. Indeed, if we rescale the normalized predictions of 'Air Temperature', we get an average prediction error of $2.8°$C.

Note that our model can reconstruct the data well for moderate size k (≈ 6). This suggests that our model should be able to recover missing data from observed values.

D. Experiments on Estimating Missing Data

Sensor nodes in our network duty-cycle their activity at a particular rate in order to increase the battery life. In these experiments, we want to study how well we recover the missing entries for different levels of duty-cycling for different choices of the model parameter k. For data collected from each sensor node, we randomly select p percent of the rows (i.e. data collected at the node for p fraction of random time instances), and remove *all* the sensor data associated at the selected rows for that node. We use the remaining data for training the model and the removed data for testing the data recovery capabilities of our model.

Recall that a basic latent variable factor model cannot recover missing entries from such data (see Section III-B). Here we study how the temporal prior (discussed in Section IV) helps in recovering the data.

Figure 3 shows the reconstruction error in estimating the missing data for various levels of data sparsity (sensor node duty-cycling). Each experiment is conducted 10 times. We report the average reconstruction error. For each fixed k (the model parameter), there is only a small degradation in prediction accuracy for different levels of matrix sparsity (duty-cycling). For $k = 4$, we have about 0.33 units (or about 8 percent) of average reconstruction error for nearly 20 to 60 percent of missing data. This shows that there is only a little degradation in prediction quality with extreme amounts of missing data.

Another interesting aspect of this model is that increasing k has a 'diminishing returns' effect: constantly increasing k gives lower improvement in error. This is primarily due the fact that even though increasing the model size can better fit the observed (training) data, due to overfitting, it cannot give better and better results on the missing (testing) data. Figure 4 shows this trade off for a fixed level of matrix sparsity. We get similar results for other levels of sparsity as well.

E. Predicting Blocks of Missing Data

Another interesting aspect is how well can we estimate the data when the node goes offline for a long block of time. This analysis is particularly useful when nodes experience long periods of inactivity due to, say, a mechanical failure or running out of battery. Consider, for instance, a scenario when a node placed in a remote location runs out of energy. It may take a few hours to a few days to reach it and replace the battery. Notice that in such a setting *consecutive* sensor readings from all the sensors on the node are missing. We analyze how well our proposed model performs in such a situation.

Given the CIMIS data, we pick a node and fix a block size of t time units. We remove all the data from the block and study how well we can reconstruct it, given different amounts of data surrounding the missing block.

Figure 5 (left) shows the quality of reconstruction for a fixed block size as we vary the amount of surrounding data, and Figure 5 (right) shows the quality of reconstruction for different block sizes for a fixed amount of surrounding data.

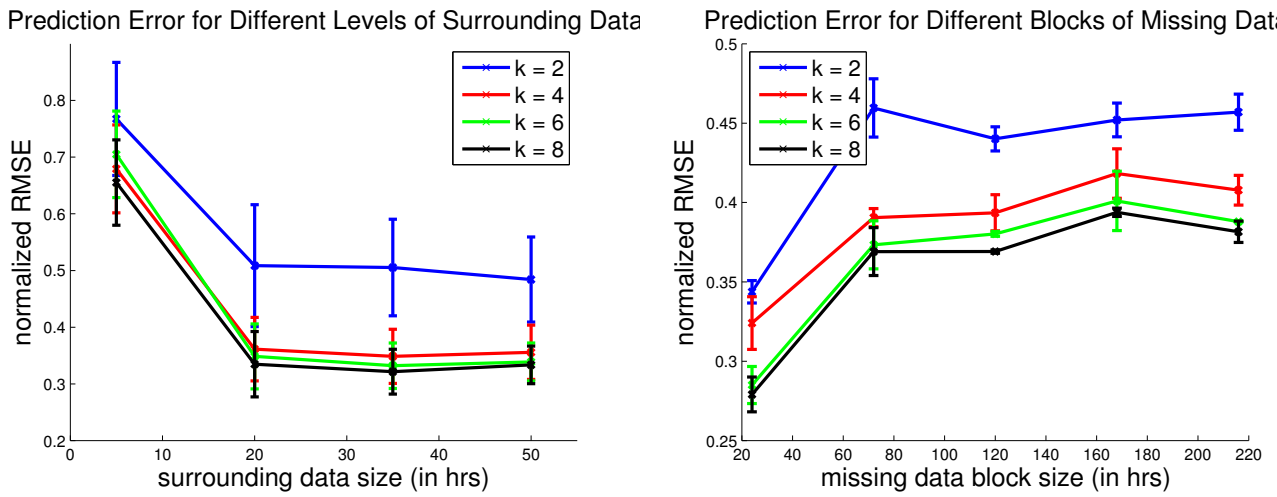

Fig. 5. Left: Reconstruction error (in normalized units) for different amounts of available surrounding data for a fixed 48 hour (2 day) block of missing data. Right: Reconstruction error for different blocks of missing data simulating long periods of node inactivity. The amount of available measured data surrounding the missing block is fixed to 120 hours (5 days).

It is interesting to note that even for relatively small amounts of surrounding measured data, we can recover the missing block of data reasonably well. This helps validate the effectiveness of our model for real-life sensor networks where node failure occurs fairly regularly.

VI. Conclusion

In this work, we have proposed a data prediction technique based on the latent variable factor model framework that is specially geared towards data collected from sensor networks. Using a real-life environmental monitoring dataset, we observe that our data estimation technique can provide less than 10% RMSE even when more than 5 days of consecutive data is missing. In duty cycling experiments, it is able to achieve less than 10% RMSE even when nodes are duty cycling at rates more than 50%, making it a very practical alternative for sensor applications.

Acknowledgment

This research was supported by NSF Grant CNS-0932403, Center for Networked Systems (http://cns.ucsd.edu) grants CNS08TR and CNS10TR, and Qualcomm.

References

[1] L. Benini, E. Farella, and C. Guiducci, "Wireless sensor networks: Enabling technology for ambient intelligence," *Microelectronics Journal*, vol. 37, pp. 1639–1649, 2006.

[2] D. Puccinelli and M. Haenggi, "Wireless sensor networks: applications and challenges of ubiquitous sensing," *Circuits and Systems Magazine, IEEE*, vol. 5, no. 3, pp. 19–31, 2005.

[3] I. F. Akyildiz, T. Melodia, and K. R. Chowdhury, "A survey on wireless multimedia sensor networks," *Computer Networks*, vol. 51, no. 4, pp. 921–960, 2007.

[4] S. Lan, M. Qilong, and J. Du, "Architecture of wireless sensor networks for environmental monitoring," in *Education Technology and Training and International Workshop on Geoscience and Remote Sensing. ETT and GRS 2008*, vol. 1, Dec. 2008.

[5] S. Drude, "Requirements and application scenarios for body area networks," in *Mobile and Wireless Communications Summit*, Jul. 2007.

[6] D. Musiani, K. Lin, and T. S. Rosing, "Active sensing platform for wireless structural health monitoring," in *6th int. conf. on Information processing in sensor networks (IPSN)*, 2007, pp. 390–399.

[7] J. Polastre, R. Szewczyk, and D. Culler, "Telos: enabling ultra-low power wireless research," in *Information Processing in Sensor Networks (IPSN)*, April 2005, pp. 364–369.

[8] V. T. Dau, O. Tomonori, T. X. Dinh, D. V. Dao, and S. Sugiyama, "A multi axis fluidic inertial sensor," in *IEEE Sensors*, Oct. 2008.

[9] B. Calhoun, D. Daly, N. Verma, D. Finchelstein, D. Wentzloff, A. Wang, S.-H. Cho, and A. Chandrakasan, "Design considerations for ultra-low energy wireless microsensor nodes," *Computers, IEEE Transactions on*, vol. 54, no. 6, pp. 727–740, jun 2005.

[10] D. Brunelli, L. Benini, C. Moser, and L. Thiele, "An efficient solar energy harvester for wireless sensor nodes," in *Design, Automation and Test in Europe (DATE)*, March 2008, pp. 104–109.

[11] Zigbee Alliance, "Zigbee," http://www.zigbee.org/.

[12] P. Zappi, C. Lombriser, T. Stiefmeier, E. Farella, D. Roggen, L. Benini, and G. Trster, "Activity recognition from on-body sensors: Accuracy-power trade-off by dynamic sensor selection." in *European Workshop on Sensor Networks (EWSN)*, 2008, pp. 17–33.

[13] N. Jiang and L. Gruenwald, "Estimating missing data in data streams," in *12th International Conference on Database Systems for Advanced Applications*, 2007.

[14] L. Gruenwald, H. Chok, and M. Aboukhamis, "Using data mining to estimate missing sensor data," in *IEEE International Conference on Data Mining (ICDM)*, Oct. 2007, pp. 207–212.

[15] P. Garcia-Laencia, J. Sancho-Gomes, A. Figueiras-Vidal, and M. Verleysen, "K nearest neighbours with mutual information for simultaneous classification and missing data imputation," *Neurocomputing*, vol. 72, pp. 1483–1493, 2009.

[16] W. Bajwa, J. Haupt, A. Sayeed, and R. Nowak, "Compressive wireless sensing," in *International Conference on Information Processing in Sensor Networks (IPSN)*, 2006, pp. 134–142.

[17] Q. Ling and Z. Tian, "Decentralized sparse signal recovery for compressive sleeping wireless sensor networks," *IEEE Transactions on Signal Processing*, vol. 58, no. 7, pp. 3816–3827, july 2010.

[18] S. Papadimitriou, J. Sun, and C. Faloutsos, "Dimensionality reduction and forecasting on streams," in *Data Streams*, ser. Advances in Database Systems, C. C. Aggarwal, Ed. Springer US, 2007, vol. 31, pp. 261–288.

[19] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *IEEE Computer*, vol. 42, no. 8, 2009.

[20] Y. Koren and R. Bell, *Recommender Systems Handbook*. Springer, 2011.

[21] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[22] Office of Water Use Efficiency, California Department of Water Resources, "CIMIS dataset," http://www.cimis.water.ca.gov.