iST-MRF: Interactive Spatio-Temporal Probabilistic Models for Sensor Networks

Nico Piatkowski

Department of Computer Science, LS 8
TU Dortmund University
D-44227 Dortmund, Germany
nico.piatkowski@tu-dortmund.de,
http://www-ai.cs.tu-dortmund.de

Abstract. Streams of sensor measurements arise from twitter, mobile phone networks, internet traffic, road traffic, home automation systems, seismic motion and sea level - to mention just a few. The interactive exploration and modelling of such measurements from multiple sensors induces the need for algorithms that are capable of processing the data as it becomes available and that can quickly provide partial results based on the data seen so far. Beside these requirements, the algorithm should capture the inherent spatio-temporal dependency structure within sensor data and allow a predictive analysis on arbitrary subsets of sensors. Spatio-Temporal Markov Random Fields (ST-MRF) are known to meet the requirements of modelling the dynamics of sensor networks. ST-MRF tracks the empirical distribution of each sensor and concurrently updates a Maximum Likelihood estimate of the underlying distribution. In the first part of this paper, we show how to train such models in an online fashion in order to perform near-instant updates to the model and provide them to the user. In the second part, we present iST-MRF, a free open source software for interactive modelling and analyzing data from sensor networks, which implements and visualizes ST-MRF. It guarantees high performance computations for offline models and concurrent learning and prediction for online models. We present two exemplary applications of iST-MRF to sensor network data, namely modelling a network of temperature sensors and a location prediction task.

1 Introduction

Data collected from multiple sources at consecutive points in time are commonly called multivariate time series or spatio-temporal data. In a sensor network, each series is measured by one sensor. Such networks are deployed to sense traffic densities on multiple streets and lanes, temperatures in multiple buildings and rooms, seismic motion on various areas on earth or the sea level at several buoys. The visualization of a corresponding data set may already become difficult. When it comes to a predictive analysis in large networks, even the specification of a prediction task may become unnecessary complex. The model should offer the opportunity to state queries like

– *"Given states at some locations and time points, what are the most likely states at some other locations and time points?"*

and mine probabilistic answers for them. Our goal is to provide an interactive tool that enables the user to explore and visualize a probabilistic model of a sensor network, while it is concurrently trained further from incoming sensor data. Methods that deal with such data sets have to be capable of processing the data as it becomes available and to quickly provide partial results based on the data seen so far. In general, we distinguish two cases of mining spatio-temporal data from sensor networks:

Offline In the *offline* setting, historical sensor readings are analyzed and used to build a model of the corresponding sensor network that generated the data set.

Online In case of *online* learning, each sensor is assumed to have compute capabilities which allow to implement the learning algorithm directly into each sensor in order to build a distributed model.

Based on Spatio-Temporal Markov Random Fields [1], we present iST-MRF which is designed to solve the following tasks:

– Learn ST-MRF model parameters in online and offline settings. In both settings, the user should be able to influence the training process by changing optimization parameters or the structure of the network.
– Visualize the computed probability densities of user selected sets of sensors.
– Detect if individual sensors show abnormal behavior, e.g. if a sensor generates a lot of measurements with low probability.
– Allow an interactive specification of prediction task, e.g. the user may set the values of certain sensors at certain points in time and the model predicts all the other values.

These task appear frequently in analysis of real world sensor networks. Note that the last task includes answering probabilistic queries, like the one stated above, as a special case. The required degree of user interaction is achieved by allowing the user to change nearly any parameter of the training process. This includes the manual setting of the stepsize, line-search parameters and insertion or deletion of nodes and edges to the underlying dependency graph. Additionally, the user gets statistics about the current training progress and visualizations (in terms of plots) of the empirical, estimated and conditional probability distributions for each sensor. The user interface of iST-MRF is shown in Figure 1. There, the graph in the middle corresponds to a sensor network. Because of the crowded areas which can be observed in the figure, it is possible to zoom in and out. Details about the current structure are shown in the panel on the right hand side, i.e. the current network contains 186 sensors with 237 dependencies between sensors. The spatio-temporal model contains 26k nodes which is simply the number of sensors multiplied with the length of the time series. Each sensor is capable of measuring four different values, e.g. the sensed

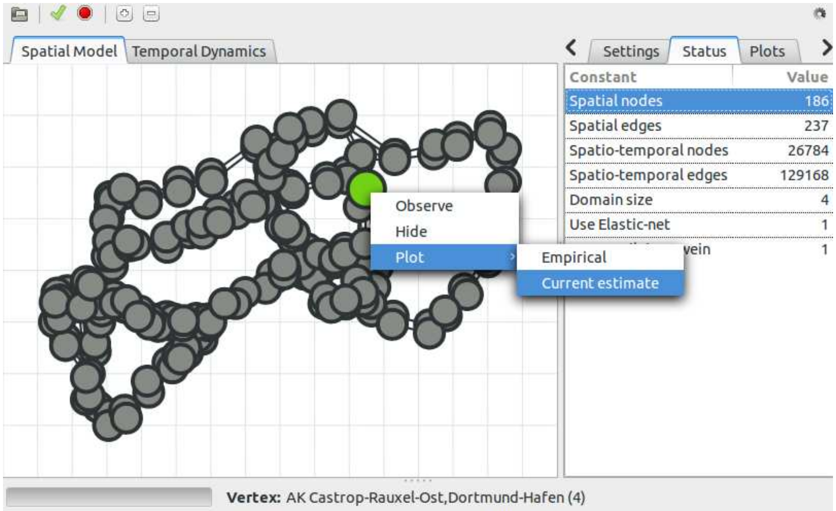

Fig. 1. Screenshot of iST-MRF. The graph in the middle corresponds to a sensor network. Details about the current structure are shown in the panel on the right hand side.

value is in $\{0, 1, 2, 3\}$. In contrast to all the grey vertices, the green colored vertex is selected and represents the sensor whose name is shown in the status bar at the bottom. The box at center is the context menu of the selected vertex. The user may set (*observe*) the vertex with a specified value at a specified point in time, reset (*hide*) any observation made on this vertex so far and show plots of its current *empirical* and *estimated* probability densities. That is, the probabilities of the four states over time.

In this paper, we show how the ST-MRF framework can be used for interactive exploration and modelling of measurements from multiple sensors. The base model is illustrated in Section 2. A description of the algorithms can be found in Section 3. The usage of iST-MRF is explained in Section 4. Exemplary applications that are illustrating what kinds of analysis can be done with iST-MRF on spatio-temporal data are shown in Section 5. In Section 6, we conclude and indicate opportunities for further work.

1.1 Related Work

A view of data mining towards *distributed sensor measurements* is presented in the book on ubiquitous knowledge discovery [2]. There are several approaches to distributed stream mining based on work like, e.g., [3] or [4]. The goal in these approaches is a general model (or function) which is built on the basis of local models while restricting communication costs. Most often, the global model allows to answer threshold queries, but also clustering of nodes is sometimes handled. Although the function is more complex, the model is global and not tailored for the prediction of measurements at a particular location. In contrast, our model can predict some sensor's state at some point in time given relevant previous and current measurements of itself and other sensors.

Since his influential book, David Luckham has promoted *complex event processing* successfully [5]. According to the slogan *Monitor, Mine, Manage* [6], series of data from heterogeneous sources are to be put to good use in diverse applications. Detecting events in streams of data has been modeled, e.g. in the context of monitoring hygiene in a hospital [7]. However, in our case, the monitoring does not imply certain events. We do not aim at finding patterns that define an event, although they may show up as a side effect. We rather want to predict a certain state at a particular sensor or set of sensors taking into account the context of other locations and points in time. Although related, the tasks differ.

Spatial relations are naturally expressed by *graphical models*. For instance, in the course of analyzing video or image data, graphical models such as Markov Random Fields (MRF) have been used [8], [9]. The standard algorithm for inference in MRF is Belief Propagation (BP). It converts local parameters into local probabilities which fit nicely together, that is, making them globally consistent with respect to the topology. BP is also used for the distributed computation of the gradient when learning the model directly from sensor streams.

Although ST-MRF is highly related to *Dynamic Bayesian Networks* (DBN), DBNs use directed acyclic graphs (DAG) to represent conditional dependencies. In contrast, MRFs are undirected models which allow cyclic probabilistic dependencies among sensor data. In some cases, it might be hard to impose an ordering of the sensors, which is needed to build the conditional dependency structure of DBNs, e.g. in wireless sensor networks. An undirected model fits naturally to the physical deployment of sensors and no explicit ordering of the sensors has to be chosen.

To the best of our knowledge, there is currently no implementation of ST-MRF (or even generic MRF) that is capable of performing the tasks listed in the introduction.

2 Spatio-Temporal Models

In order to model pairwise interactions of nearby sensors with respect to space and time, we consider Spatio-Temporal Markov Random Fields [1]. This model do enhance the general framework of MRF by considering a sequence of graphs $G_1, G_2, \ldots, G_T$ where each graph $G_t = (V_t, E_t)$ is called *layer*. Each layer serves as an undirected spatial dependency structure of all sensors at time t and each vertex from the set V_t corresponds to a sensor measurement at time t. Each edge $\{v, u\} \in E_t \subset V_t \times V$ with $V := \bigcup_{t=1}^{T} V_t$ encodes conditional independence assumptions among sensor measurements, i.e. a sensors value at time t is fully determined by its spatio-temporal neighborhood, which may contain values from several different $V_{t'}$. In the basic setting of ST-MRF, the spatial structure is stationary over time, that is $(v_t, u_{t'}) \in E_t \Leftrightarrow (v_{t+1}, u_{t'+1}) \in E_{t+1}$ for all

$1 \leq t \leq T$ with $v_{T+t} := v_t$. For this kind of model, the projection from real time, e.g. 2:00am, to time index t usually depends on the period that should be modeled and the physical sampling rate of the sensors. As an example, consider a building with n rooms, each equipped with a temperature sensor that measures the temperature every minute. This results in 1440 sensor readings per day per sensor, which means $T = 1440$. Is it also possible (and common) to build models with a temporal resolution that is lower than the physical sampling rate of the sensors. If we choose $T = 144$ in the above, measurements within consecutive intervals of 10 minutes have to be aggregated (min, max, mode, average, ..) or simply ignored. The particular projection is data and task dependent.

The spatial dependency structure is often given by the physical deployment of sensors but the temporal dependency structure is usually unknown. However, we show that our online update does allow insertion and deletion of edges and nodes. The proposed approach starts with an initially pre-specified temporal structure which can be changed interactively by the user while the model is continuously adapted to the data stream.

In the remainder of this paper, we use $|S|$ to denote the cardinality of a finite set S and symbols describing a vector entity are set in bold.

2.1 Learning from spatio-temporal sensor data

Although we want to learn a model from a stream of sensor measurements, lets assume for now that the data is given in a set $\mathcal{T} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(N)}\}$, where each $\boldsymbol{x}^{(i)}$ is the i-th joint reading of all sensors in the network over a period of length T. We denote the measurement of a single sensor s at time t as $\boldsymbol{x}_{s_t}$. Without loss of generality, we assume that sensor measurements are discretized, i.e. each sensor measures a discrete state from finite set $\mathcal{X}$. The model parameters $\boldsymbol{\theta} \in \mathbb{R}^d$ consist of one weight vector $\boldsymbol{\theta}_{s_t} \in \mathbb{R}^{|\mathcal{X}|}$ per spatio-temporal vertex and one weight matrix $\boldsymbol{\theta}_{\{s_t, v_{t'}\}} \in \mathbb{R}^{|\mathcal{X}| \times |\mathcal{X}|}$ per spatio-temporal edge. This results in a total of $d = |V| \cdot |\mathcal{X}| + |E| \cdot |\mathcal{X}|^2$ parameters, with $E := \bigcup_{t=1}^{T} E_t$. Instead of using discretized measurements, we could assume that measurements have a Gaussian distribution. In this case, $\boldsymbol{\theta}_{s_t}$ would store the mean and $\boldsymbol{\theta}_{\{s_t, v_{t'}\}}$ the partial correlation coefficients. In either case, the parameters can be obtained by Maximum Likelihood Estimation, whereby the Likelihood of a particular $\boldsymbol{\theta}$ given some data set $\mathcal{T}$ is defined as

$$\mathcal{L}(\boldsymbol{\theta}; \mathcal{T}) := \prod_{i=1}^{N} p_{\boldsymbol{\theta}}(\boldsymbol{X} = \boldsymbol{x}^{(i)}). \tag{1}$$

Here, $p_{\boldsymbol{\theta}}(\boldsymbol{X} = \boldsymbol{x}) = \exp(\langle \boldsymbol{\theta}, \boldsymbol{\phi}(\boldsymbol{x}) \rangle - A(\boldsymbol{\theta}))$ is the density of an exponential family member [10] which obeys the conditional independence structure given by G, $\boldsymbol{\phi}(\boldsymbol{x})$ is a function (sufficient statistic) that maps $\boldsymbol{x}$ into a d-dimensional binary vector space and $A(\boldsymbol{\theta})$ normalizes the density. The components of vector $\boldsymbol{\phi}$ represents a joint measurement of all sensors and all edges in the network as binary values. For sensors $v, w \in V$ and states $a, b \in \mathcal{X}$ the entries are as follows:

$$\begin{aligned} \boldsymbol{\phi}_{v,a}(\boldsymbol{X}) &:= \begin{cases} 1 & \text{if } \boldsymbol{X}_v = a \\ 0 & \text{otherwise,} \end{cases} \\ \boldsymbol{\phi}_{vu,ab}(\boldsymbol{X}) &:= \begin{cases} 1 & \text{if } \boldsymbol{X}_v = a \text{ and } \boldsymbol{X}_u = b \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{2}$$

If the model is implemented directly into a sensor network for online processing, we do not want it to store its complete history of measurements. Therefore, we take the logarithm of the Likelihood (1) and rearrange, such that it only depends on the average value or the *empirical expectation* $\tilde{\mathbb{E}}\left[\boldsymbol{\phi}(\boldsymbol{x})\right]$ of our sufficient statistic. Note that the extra $\frac{1}{N}$-factor does not change the optimal solution.

$$\ell(\boldsymbol{\theta}; \mathcal{T}) := \frac{1}{N} \sum_{i=1}^{N} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) = \left\langle \boldsymbol{\theta}, \tilde{\mathbb{E}}\left[\boldsymbol{\phi}(\boldsymbol{x})\right] \right\rangle - A(\boldsymbol{\theta}) \tag{3}$$

This implies that each sensor has to store an aggregate of historical readings instead of a full history. This fits to a streaming scenario, since we only have to count how often a certain combination appears in the stream. Taking derivatives of (3) it follows that

$$\frac{\partial \ell(\boldsymbol{\theta}; \mathcal{T})}{\partial \boldsymbol{\theta}_{v,a}} = \tilde{\mathbb{E}}\left[\boldsymbol{\phi}_{v,a}(\boldsymbol{x})\right] - \hat{\mathbb{E}}\left[\boldsymbol{\phi}_{v,a}(\boldsymbol{x})\right],$$

whereby $\boldsymbol{\theta}_{v,a}$ denotes the weight of sensor v and state $a \in \mathcal{X}$ and the *estimated expectation* $\hat{\mathbb{E}}\left[\boldsymbol{\phi}_{v,a}(\boldsymbol{x})\right]$ is computed by Belief Propagation (BP) [11]. The BP algorithm is a distributed message passing algorithm and it is used for inference in both online and offline models. It is also known as Sum-Product algorithm in the context of factor graphs [12]. Note that graphs of ST-MRF models contain many loops per definition and that BP is an approximate method in this case. The objective $\max_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}; \mathcal{T})$ can then be solved by any first-order optimization method.

The prediction is also executed by the BP algorithm. For this purpose, we fix the measurements of some sensors, say u at time t' and w at time t'', and run BP until convergence. The computed beliefs for any sensor v at time t will approximate $p_{\boldsymbol{\theta}}(\boldsymbol{X}_{v_t} \mid \boldsymbol{x}_{u_{t'}}, \boldsymbol{x}_{w_{t''}})$. Thus, the predicted value of each sensor v at time t is $\boldsymbol{x}^*_{v_t} = \arg\max_{x \in \mathcal{X}} p_{\boldsymbol{\theta}}(\boldsymbol{X}_{v_t} = x \mid \boldsymbol{x}_{u_{t'}}, \boldsymbol{x}_{w_{t''}})$. In general, we may use any two disjoint subsets $A, B \subset V$ with corresponding assignments $\boldsymbol{x}_A, \boldsymbol{x}_B$ to compute $p_{\boldsymbol{\theta}}(\boldsymbol{X}_A = \boldsymbol{x}_A \mid \boldsymbol{x}_B)$ with BP. We use this kind of prediction in the exemplary predictive analysis of sensor networks in Section 5.

In case of continuous $t \in \mathbb{R}$, a prediction can still be made by interpolating the probability between the two surrounding layers $\lfloor t \rfloor$ and $\lceil t \rceil$ and choosing its maximum point. A first-order interpolation is given by

$$\bar{p}_{\boldsymbol{\theta}}(\boldsymbol{X}_{v_t}) = p_{\boldsymbol{\theta}}(\boldsymbol{X}_{v_{\lfloor t \rfloor}}) + (p_{\boldsymbol{\theta}}(\boldsymbol{X}_{v_{\lceil t \rceil}}) - p_{\boldsymbol{\theta}}(\boldsymbol{X}_{v_{\lfloor t \rfloor}}))(t - \lfloor t \rfloor).$$

2.2 Changing the graphical structure

The general framework of probabilistic graphical models considers a fixed[1] graph G. This results from the fact, that adding vertices or edges to G would also change $\boldsymbol{\theta}$ and thus the associated optimization problem. A simple consequence is that we cannot continue the optimization if the graph has changed, the only way is to restart the training. Notice, that the new training can be started from any initial solution $\boldsymbol{\theta}$ that we choose. Consider a model which gets a new edge $\{u,v\}$ added during training: We add the new weight matrix $\boldsymbol{\theta}_{\{u,v\}}$ to $\boldsymbol{\theta}$, initialize its values to 0 and leave all the other parameters untouched. In case of dynamic or annealing stepsize, we also have to reset the stepsize to its initial value. The next iteration will behave like after a restart of the optimization procedure. The training may now continue approaching the Maximum Likelihood estimate.

3 iST-MRF

Now, the basic algorithms for offline and online training of ST-MRF are presented and discussed. For notational convenience, let $T+1 := 1$.

3.1 Offline Training

The offline training of iST-MRF has to execute the following algorithm:

1. Load G and create $\boldsymbol{\theta}$
2. Initialize $\boldsymbol{\theta} := \mathbf{0}$
3. Load $\mathcal{T}$
4. Compute $\tilde{\mathbb{E}}\left[\boldsymbol{\phi}(\boldsymbol{x})\right]$ from $\mathcal{T}$
5. While not converged:
   (a) Compute $\hat{\mathbb{E}}\left[\boldsymbol{\phi}(\boldsymbol{x})\right]$ with BP($\boldsymbol{\theta}$)
   (b) Update $\boldsymbol{\theta}^{\text{new}} := \boldsymbol{\theta}^{\text{old}} + \eta \cdot (\tilde{\mathbb{E}}\left[\boldsymbol{\phi}(\boldsymbol{x})\right] - \hat{\mathbb{E}}\left[\boldsymbol{\phi}(\boldsymbol{x})\right])$
   (c) Update η
   (d) If sensor v or edge $\{v,u\}$ is added to G:
      i. For each $1 \leq t \leq T$: Create $\boldsymbol{\theta}_{v_t}$ or $\boldsymbol{\theta}_{\{v_t,u_t\}}$,$\boldsymbol{\theta}_{\{v_{t+1},u_t\}}$ and $\boldsymbol{\theta}_{\{v_t,u_{t+1}\}}$
      ii. Initialize the new parameters with $\mathbf{0}$
      iii. Reset η
      iv. Goto 3.

This resembles usual Maximum Likelihood estimation but with the opportunity to change the graphical structure and restart the optimization. In order to have thread safe parallel computations in the offline version, we use the Graphlab framework [13] for implementing BP.

[1] In case of non-parametric graphical models, the graph may not be fixed. A distribution over several graphs is used instead.

3.2 Online Training

For online processing, a fully distributed version is needed, since data may arise asynchronously at several sources. The following algorithm (*online*) is executed by each sensor v in the network:

1. Create, for all $1 \leq t \leq T$: $\boldsymbol{\theta}_{v_t}$, $\boldsymbol{\theta}_{\{v_t,u_t\}}$,$\boldsymbol{\theta}_{\{v_{t+1},u_t\}}$ and $\boldsymbol{\theta}_{\{v_t,u_{t+1}\}}$ for all neighbors u and initialize the new parameters with $\mathbf{0}$
2. Initialize $\tilde{\mathbb{E}}\left[\boldsymbol{\phi}(\boldsymbol{x})\right] := \mathbf{0}$
3. Start BP($\boldsymbol{\theta}$) concurrently
4. While new data $\boldsymbol{x}_{v_t}$ is available:
   (a) Send $\boldsymbol{x}_{v_t}$ to neighbors
   (b) Add $\boldsymbol{x}_{v_t}$ to cache C
   (c) Receive measurements $\boldsymbol{x}_{u_t}$ from neighbors and add them to C
   (d) Update $\tilde{\mathbb{E}}\left[\boldsymbol{\phi}(\boldsymbol{x})\right]$ with C and clear C
   (e) Update $\boldsymbol{\theta}^{\text{new}} := \boldsymbol{\theta}^{\text{old}} + \eta \cdot (\tilde{\mathbb{E}}\left[\boldsymbol{\phi}(\boldsymbol{x})\right] - \hat{\mathbb{E}}\left[\boldsymbol{\phi}(\boldsymbol{x})\right])$
   (f) Update η
   (g) If adjacent edge $\{v,u\}$ is added to G:
      i. For each $1 \leq t \leq T$: Create $\boldsymbol{\theta}_{\{v_t,u_t\}}$,$\boldsymbol{\theta}_{\{v_{t+1},u_t\}}$ and $\boldsymbol{\theta}_{\{v_t,u_{t+1}\}}$
      ii. Initialize the new parameters with $\mathbf{0}$
      iii. Reset η
      iv. Goto 4.

In Section 2.1, we already mentioned that each sensor has to store an aggregate of historical readings instead of a full history. To be more precise, computing $\tilde{\mathbb{E}}\left[\boldsymbol{\phi}(\boldsymbol{x})\right]$ can be implemented in the turnstile[2] model of streaming [14], since we only have to count the number of occurrences of certain configurations in the stream. Those counts may also be approximated from the stream by the count-min sketch [15] or similar methods. We use an exact approach that caches the recent own and neighboring measurements and allows the computation of joint counts (and therefore empirical expectations) between a sensor and its neighbors. These are needed to compute the empirical expectation of $\boldsymbol{\phi}_{vu,ab}$ defined in (2). Each sensor that executes an instance of the online algorithm may read its measurements from an actual measurement device (e.g. a temperature sensor), from a file or database. Thus, the same application may be used to simulate the sensor network while reading from file or using actually measured values and implementing the model directly into an existing network. The empirical, estimated and conditional densities are collected on demand if requested by the user. It is clear from line 4, that Belief Propagation is running in the background. The computation and propagation of new BP messages is started, whenever the difference of the current empirical expectation and the one which was used to compute the current parameters $\boldsymbol{\theta}$ falls below a user specified parameter ρ. Briefly, the algorithm reacts on changes in $\tilde{\mathbb{E}}\left[\boldsymbol{\phi}(\boldsymbol{x})\right]$ with an updated $\hat{\mathbb{E}}\left[\boldsymbol{\phi}(\boldsymbol{x})\right]$.

[2] In the turnstile model, new data may appear in an arbitrary order and undergo any number of additive updates.

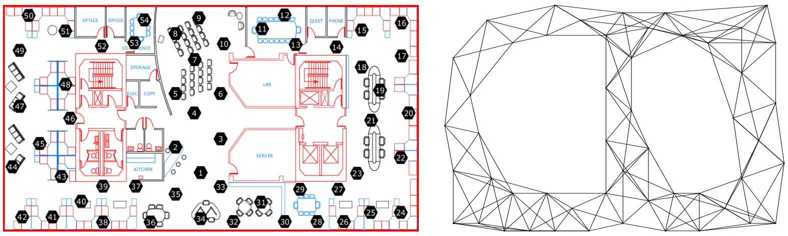

Fig. 2. Spatial structure used in the second example task. Left: Diagram that show the arrangement of sensors in the lab. Right: Nearest neighbor graph after removing faulty sensors and edges that were blocked by a wall or similar obstacle.

4 iST-MRF usage

The following describes the usage of iST-MRF. Three basic steps are involved to create an initial model:

Provide data. Data are loaded from one or multiple `.csv`-file(s). This generates as many nodes as columns are available in the data, whereas the first row is interpreted as sensor names. The initial graph is fully disconnected and visualized by a simple grid layout without any edges.

Provide a graphical structure. Vertices may either be connected by drawing lines between them or by loading a graphical structure from a `.graphml`-file [16]. The node names in those files have to be the same as the column names in the data. Nevertheless, it is possible to rename the nodes.

Setup ST-MRF. Basically, only the total number of layers has to be defined. It is also possible to set an initial stepsize η_0, elastic-net control parameter λ_1 and λ_2 and an BP termination bound, but all of these variables have reasonable default values.

Next, the iterative optimization procedure can be started for a user supplied number of iterations. Nevertheless, the user can always

- pause the learning process,
- adjust the above-mentioned parameters,
- add or remove edges,
- watch how the empirical and estimated densities change with new measurements or running optimization, respectively,
- store/load the model to/from a file,
- compute predictions for user selected sensors and user specified fixed values for a disjoint set of sensors.

If parameters or the model itself is changed during training, the changes are applied in the next iteration.

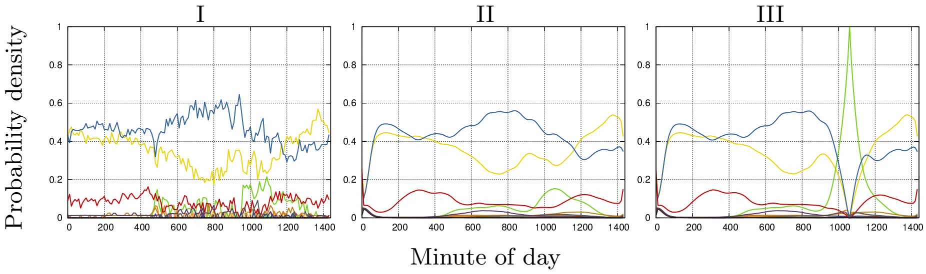

Fig. 3. Temporal dynamics of empirical (I), estimated (II) and conditional (III) marginal probability densities for a fixed user (A) and all cells that the user has visited. Each color represents the probability for one cell over time. The densities in the right plot are conditioned on the single event that the green cell is observed at 5:40pm.

5 Exemplary Applications

The examples here proposed, should demonstrate that iST-MRF is capable of performing at least three of the four tasks listed in the introduction. We present the results of our first experiments that where executed by iST-MRF. For training, we ran gradient descent optimization with an elastic-net regularization [17] until convergence to find proper model parameters. Although these experiments give only a rough impression of which kinds of predictive analysis can be done with iST-MRF, they serve as simple examples on how spatial-temporal data can be analyzed.

5.1 Network cell prediction

For the first task, we applied iST-MRF to a network cell prediction task [18]. The data consists of trajectories in terms of mobile network cell identifiers for several users, whereas the provided trajectories contained many missing values for some users. In this case, a sensor is identified with a user and a measurement is identified with the network cell identifier. The task was to model a users behavior over 24h. Due to missing values, we set the sampling rate to 10 minutes, which results in $T = \frac{24h}{10min} = 144$. We choose a temporal first order Markov dependency without any spatial dependency, which results in a non-stationary Markov chain per user. Figure 3 shows $\tilde{\mathbb{E}}[\boldsymbol{\phi}(\boldsymbol{x})]$, $\hat{\mathbb{E}}[\boldsymbol{\phi}(\boldsymbol{x})]$ and $\hat{\mathbb{E}}\left[\boldsymbol{\phi}(\boldsymbol{x}_{V\setminus\{94\}})|\boldsymbol{x}_{94} = \text{green}\right]$ for a certain anonymous user. The plots are generated directly with iST-MRF.

5.2 Temperature Sensor Network

The second task is based on data collected in March 2004 from the sensors deployed in the Intel Berkeley Research lab (available at `http://db.csail.mit.edu/labdata/labdata.html`). The measurements consist of humidity, temperature, light, and voltage values captured at every 31 seconds. We used a half of two million sensor readings for training, and the rest for testing.

A nearest neighbor graph was constructed as basic spatial dependency structure, whereas an edge is considered disconnected whenever it is blocked by walls. We also excluded sensors reported faulty in [19]. Figure 2 shows the arrangement of sensors in the lab and the resulting structure. The final spatial graph contained 48 nodes and 150 edges. We use the temperature measurements at every 10 minutes (which resulted again in $T = 144$), since it was the finest resolution without too many missing values, and discretized them into 19 equal sized bins.

In order to evaluate the prediction accuracy, we compute for all days which are included in the test set, the most probable (discretized) temperature for all the sensors from 12:00 to 23:50pm, given all values for all sensors from 00:00am to 11:50am. This results in a normalized absolute error of 0.141643. For comparison, the same experiment was repeated but without any spatial edge between sensors. The resulting error is 0.187426.

6 Conclusions and Future Work

We presented iST-MRF, an interactive exploration and modelling tool for measurements from multiple sensors. We described the underlying theory and showed how ST-MRF can be used for online learning from multiple sensors. Our experiments could show, that iST-MRF is capable of computing empirical, estimated and conditional probability densities and visualize them on demand to the user. The second example showed that iST-MRF can perform predictions for arbitrary subsets of sensors at nearly arbitrary points in time. Furthermore, the method can handle the insertion (or deletion) of sensors and edges, which heavily increases the interactivity of the learning process.

Our future work is focused on completing the development of online iST-MRF and evaluating its performance compared to the offline version. Also for the online case, approximate counting methods like count min sketch will be investigated whether they can preserve the accuracy and qualitative features of ST-MRF while saving memory in the device.

7 Acknowledgments

Work on this paper has been supported by Deutsche Forschungsgemeinschaft (DFG) within the Collaborative Research Center SFB 876 "Providing Information by Resource-Constrained Analysis", project A1.

References

1. Piatkowski, N., Lee, S., Morik, K.: Spatio-temporal models for sustainability. In Marwah, M., Ramakrishnan, N., Berges, M., Kolter, Z., eds.: Proceedings of the SustKDD Workshop within ACM KDD 2012, ACM (August 2012)
2. May, M., Saitta, L., eds.: Ubiquitous Knowledge Discovery. In May, M., Saitta, L., eds.: Ubiquitous Knowledge Discovery. Volume 6202 of Lecture Notes in Artificial Intelligence., Springer (2010)

3. Wolff, R., Badhuri, K., Kargupta, H.: A generic local algorithm for mining data streams in large distributed systems. IEEE Transactions on Knowledge and Data Engineering **21**(4) (2009) 465–478
4. Sagy, G., Keren, D., Sharfman, I., Schuster, A.: Distributed threshold querying of general functions by a difference of monotonic representation. In: Procs. of the VLDB Endowment. Volume 4. (2011)
5. Luckham, D.: The Power of Events - An Introduction to Complex Event Processing in Distributed Enterprise Systems. Addison Wesley (2002)
6. Campbell, D.: Is it still Big Data if it fits in my pocket? In: Procs. VLDB 2011. (2011)
7. Wang, D., Rundensteiner, E.A., Ellison, R.T.: Active complex event processing of event streams. In: Procs. of the VLDB Endowment. Volume 4. (2011)
8. Yin, Z., Collins, R.: Belief Propagation in a 3D Spatio-temporal MRF for Moving Object Detection. IEEE Computer Vision and Pattern Recognition (CVPR) (June 2007)
9. Huang, R., Pavlovic, V., Metaxas, D.: A New Spatio-Temporal MRF Framework for Video-based Object Segmentation. In: The 1st International Workshop on Machine Learning for Vision-based Motion Analysis - MLVMA'08, Marseille, France (2008)
10. Wainwright, M.J., Jordan, M.I.: Graphical Models, Exponential Families, and Variational Inference. Foundations and Trends in Machine Learning **1**(1–2) (2007) 1–305
11. Pearl, J.: Probabilistic reasoning in intelligent systems: networks of plausible inference. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1988)
12. Kschischang, F.R., Frey, B.J., Loeliger, H.A.: Factor graphs and the sum-product algorithm. IEEE Trans. on Infor. Theory **47**(2) (2001) 498–519
13. Low, Y., Gonzalez, J., Kyrola, A., Bickson, D., Guestrin, C., Hellerstein, J.M.: Graphlab: A new parallel framework for machine learning. In: Conference on Uncertainty in Artificial Intelligence (UAI), Catalina Island, California (July 2010)
14. Muthukrishnan, S.: Data streams: algorithms and applications. Found. Trends Theor. Comput. Sci. **1**(2) (August 2005) 117–236
15. Cormode, G., Muthukrishnan, S.: Approximating Data with the Count-Min Data Structure. IEEE Software (2012)
16. Brandes, U., Eiglsperger, M., Herman, I., Himsolt, M., Marshall, M.S.: Graphml progress report structural layer proposal. Centrum (2001) 501–512
17. Zou, H., Hastie, T.: Regularization and variable selection via the elastic net. Journal of the Royal Statistical Society B **67** (2005) 301–320
18. Michaelis, S., Piatkowski, N., Morik, K.: Predicting next network cell IDs for moving users with Discriminative and Generative Models. In: Mobile Data Challenge by Nokia Workshop in conjunction with Int. Conf. on Pervasive Computing. (June 2012)
19. Apiletti, D., Baralis, E., Cerquitelli, T.: Energy-saving models for wireless sensor networks. Knowledge and Information Systems **28** (2011) 615–644 10.1007/s10115-010-0328-6.