Predicting next network cell IDs for moving users with Discriminative and Generative Models

Stefan Michaelis
Artificial Intelligence Group
TU Dortmund University
Dortmund, Germany
stefan.michaelis@tu-dortmund.de

Nico Piatkowski
Artificial Intelligence Group
TU Dortmund University
Dortmund, Germany
nico.piatkowski@tu-dortmund.de

Katharina Morik
Artificial Intelligence Group
TU Dortmund University
Dortmund, Germany
katharina.morik@tu-dortmund.de

ABSTRACT

In mobile networks, moving users generate trails of traversed cells. For the network, this leads to a constant shift in data traffic loads when users with active connections switch from cell to cell. For active load balancing, a prediction of the next cell a user will move into enables reservation of resources a long time before standard micro-mobility handoff mechanisms.

The generated paths of the users train machine learning algorithms and predict the next cells. The learned model should give an impression of the relationship between geographical topology (streets, railways etc.) and radio coverage. The experiments presented in this paper have been able to exceed 80% accuracy for predicting the next cell of certain users.

1. INTRODUCTION

Network traffic, especially data, is ever increasing in mobile networks, putting a huge burden on the infrastructure. In parallel, the demand for high-quality, high-bandwidth, low-latency connections is rising due to widespread availability of smartphones and tablets. These devices enable applications from online gaming up to video streaming and video calls. A-priori knowledge about the next cell of a mobile user could deliver an indicator about upcoming changes in network routing and load. The operator can automatically and pro-actively react on these changes, e.g. by reserving capacity in the predicted cell.

The prediction of next cells is performed by training a model with the historical sequences of mobile user's traversed base stations. In [1] one of the earliest approaches to this problem has been described, using a compression algorithm to generate a tree of cell handoffs per user. A good overview about some basic technologies used for location predictions can be found in [3]. The methodology there is divided into domain-dependent/independent and user-specific or global models. In comparison to the methods presented there, we concentrate on domain-independent algorithms, i.e. not specifically implementing mobile network-only functions. This enables applying the same algorithms for different mobile networks, from GSM to LTE, and easily adapting new features like positioning as they become available by new technology. Typically, most machine learning algorithms provide some robustness against outliers and noisy data, e.g. during movement along cell borders.

Most other approaches on next cell predictions, even newer publications than mentioned afore, concentrate on modelling movement behaviour per user. While this is reasonable, it may severely impact user privacy. Therefore, we investigate both methods, starting with generating models independently of the user.

Often, only simulated cellular networks and movements have been used to measure the achievable prediction accuracy of the investigated algorithm. Here, real world data gathered by the Nokia Data Collection Campaign in the Lake Geneva region [6] has been used to validate the predictability of movements with minimal features and sophisticated algorithms. In [7] we already demonstrated some of the network optimization capabilities assuming that the user's next cell can be predicted. All techniques mentioned there provide some fallback to guarantee successful handoffs even if the prediction fails.

2. FEATURE GENERATION FROM MOBILITY TRACES

In-depth data of phone usage, location and seen cells has been provided in the measurement data. Details about the data sets can be found in [6]. For the specific task of predicting a user's next cell, only a minimal subset of the data is used, which would be available in many different mobile networks and observable without alteration of existing radio protocols.

Optimally, the features are available on the network side without the need for further information from the user's mobile phone.

2.1 Features available on network level

The selection of available features as input to the pattern detection algorithm is an essential parameter for the quality of the prediction of the next cell.

Figure 1 presents some features ideally being available in the network: The ID of the cell, the user is currently associated with and for handoff events the time between the last two events, i.e. the residence time inside the cell. Addi-

This material was prepared for the Mobile Data Challenge 2012 (by Nokia) Workshop; June 18-19, 2012; Newcastle, UK.

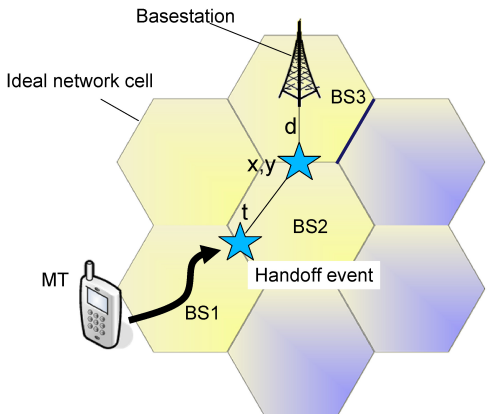

Figure 1: Network level features during user movements

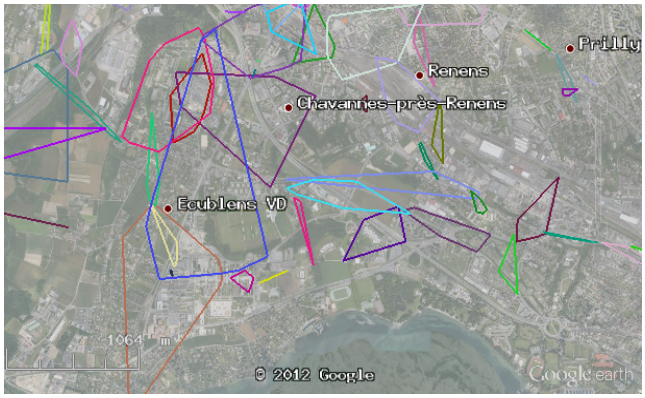

Figure 2: Cell coverage plot based on GPS measurements of one user

tionally, in some cases the user's position x, y at the time of the event (e.g. by triangulation or GPS) and the distance d from base station to user (e.g. estimated by timing advance parameters in GSM) may be available. Beside distance estimations, all of these features can be extracted from the provided measurement data.

Therefore, a sample sequence of the last three base stations and positions to be used in training may appear as *BS1, N51.3°, E7.6°, BS2, N51.4°, E7.6°, BS3, N51.5°, E7.7°*.

The maximum length k of the sequences is limited to prevent intense growth of training examples caused by exceptionally long usage of the mobile phone. Typically, training examples do not provide further useful patterns beyond a certain length. Selecting a good sequence length for obtaining a high prediction accuracy is also a part of the process optimizing feature selection and algorithm parametrization.

Each sequence provides a windowed view on the user's historical observations of seen cells. Each point i in a sequence (this may be a cell ID only or the cell ID in combination with a position) marks an event in the user's trace.

Only measurements leading to a change in the observed cell are inserted into the sequence, as these lead to a shift in network load. Intermediate measurements for the same cell as the cell before are not used in a cell sequence.

2.2 Limitations in used measurement data

The data used for the experiments provides an in-depth view on the mobile phone's usage and handling. Nevertheless, the way the data is gathered complicates the predictions in some points.

Figure 2 shows an estimated cell coverage plot of one single user near the Lausanne city center. To calculate the coverage, each cell measurement is associated by a GPS position if available and afterwards the minimal enclosing convex polygon is calculated and plotted. As the cell measurements are performed following approximately a 60s interval, moving users may pass by several other cells before the next record.

Additionally, in most cases more than one cell has to be in range, while the measurements only cover one cell. Our own measurements in [8] showed, that the position of handoffs between neighbouring cells strongly depends on the operator's implementation of the handoff algorithm, tending to stay longer in cells during active connections than without.

Finally, only 15% of the cell measurements could be associated with a GPS position, leading to positioning gaps. GPS data is tracked approximately each 10s and we associate a cell measurement with a location measurement, if the difference in the time stamp is not greater than 15s.

Nevertheless, predictions based on these measurements can provide the operators with an idea where a user will arrive next from a network's point of view.

3. PREDICTING CELL IDS WITH DISCRIMINATIVE MODELS

The applied pattern recognition algorithm generating a discriminative model is based on the Support Vector Machine (SVM [5]), which provides a powerful solution to non-linear classification. The flexibility in parametrization enables the model to adapt to nearly any type of function, with initially high computational requirements for finding the optimal set of parameters, but fast application to new data for generating a prediction.

The basic principle lies in separating the data points provided in the training with a hyper plane. For the classification task of future cells, the prediction process can be simplified to detection on which side of the plane the data can be found.

During the training phase, the SVM calculates the optimal separating hyper plane maximizing the distance between the plane and the training examples. In most cases, not all example points are needed to identify the plane, but just the nearest points in space. These so-called Support Vectors typically reduce the influence of outliers far away from the optimal plane. Two major restrictions are obvious: The hyper plane provides only a linear separation of the data and the prediction is reduced to two classes. The latter restriction can easily be solved by splitting the problem into a one-versus-all approach (e.g. is it cell A or not, is it cell B or not,...).

The second restriction of only linear separation of data can be prevented by transforming the training data into a different space before training the support vector machine. If the data transformation is non-linear, the prediction will also be able to predict non-linear separations, while keeping the SVM model generation on just calculating the plane.

So-called kernel functions transform the linear separation into a non-linear space. For finding the optimal separating hyper plane, several parameters like kernel function, kernel

degree and tolerance values need to be evaluated.

Experiments have been executed using the RapidMiner Data Mining framework [9], running an evolutionary algorithm for SVM parameter optimization and a ten-fold stratified cross-validation for calculating the prediction accuracy. As a reference algorithm OneR is used. This algorithms selects only a single attribute (typically the current cell) and predicts the next cell based on the value of this attribute. The lower the OneR accuracy, the higher the randomness of traversing from one cell to another.

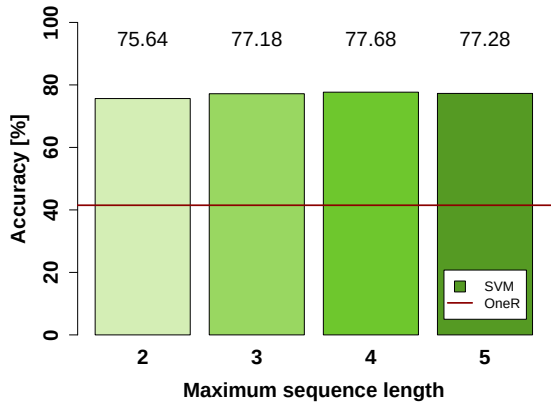

Figure 3: Prediction accuracy for all users based on sequence length

Figure 3 presents the prediction accuracy for different maximum sequence lengths k. The SVM has been trained generating a global model **independently** of a specific user, i.e. the user id has been removed from each sequence before training. This on the one hand improves privacy, as no individual user movement model is generated, and on the other hand reduces storage and computational power needed.

The OneR algorithm delivers a prediction accuracy of just 41%. Using the SVM and longer sequences improves the accuracy by more than 35% up to **nearly 78%**. Figure 3 additionally shows, that the effect of longer historical sequences is minimal beyond the last three seen cells, leading to lesser storage requirements. Standard deviation for all accuracies was below 2% each.

Beside using cell IDs only, similar experiments have been performed with cell ID in combination with residence time, GPS positions and finally without removing the user id for each sequence.

Prediction accuracies have been in a similar range compared to cell ID only, resulting in **77.67%** with residence time, **78.00%** with GPS and **78.72%** including user ids.

Especially for combining cell IDs and GPS positions we would have expected an increase in accuracy, but as only 15% of seen cells could be associated with a position, this feature represents not enough additional knowledge.

The results demonstrate the feasibility of predicting user's next seen cell, predicting the *where* a user will go, but lacking the *when* a user will be there. While it would be possible to train separate models for respecting time, the next section concentrates on generative models inherently building user profiles with time predictions.

4. INCORPORATING TIME WITH GENERATIVE MODELS

The provided GSM log data contains an inherent temporal structure which is not used by our SVM approach. In order to make use of this additional information, we applied probabilistic graphical models [12] to incorporate the time explicitly. Such an approach enables us to formulate probabilistic queries about a users movement behaviour, e.g. "Given that user A is in cell X at 5:50pm (current measurement), what is the most probable cell the user will visit at 6:10pm?". Since our model is generative and undirected, we may incorporate additional knowledge about the future which allows us to state queries like "Given that user A will be in cell X at 6:00pm (extracted from calendar data), what is the most probable cell the user will visit at 3:00pm?".

Note that the user A is explicitly mentioned in the above queries. That is because we build one seperate model per user and thus, treat all users as independent. A global model would require connectivity data between users to capture the structure of user interaction. Such connectivity data could in principle be extracted from social networks.

Another intuition behind user specific models is, that this process can be directly implemented into a cellphones software and queries like those above could be requested by the network provider on demand.

As mentioned above, we use undirected graphical models namely Markov Random Fields (MRF). The temporal structure is captured by a graph G with T vertices, whereby T is equal to a fixed number of sampling points per day. Each vertex represents a sampling point (measurement) and vertices of temporal consecutive sampling points are connected through edges to build the dependency graph. Additionally, the last and first vertices are connected to encode the repetitive temporal structure of user behavior. To be more precise, each vertex v_t is indentified with a discrete random variable $\boldsymbol{x}_t$ and the set $\mathcal{X}$ of possible realizations of $\boldsymbol{x}_t$ contains the cell IDs of all cells that the user has visited so far. If the value of a vertex is known or *observed*, e.g. $\boldsymbol{x}_{106} = y$ for a cell y, in principle all nodes in the graph are affected. Figure 4 shows how the probability densities of several cells change if a node's value is observed. The model is given by a multivariate exponential family with density

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \exp(\langle \boldsymbol{\theta}, \phi(\boldsymbol{x}) \rangle - A(\boldsymbol{\theta})). \tag{1}$$

Here, $\phi(\boldsymbol{x})$ is a sufficient statstic that transforms a random vector $\boldsymbol{x} \in \mathcal{X}^T$ into an d-dimensional binary vector space with $d := T \cdot |\mathcal{X}| + T \cdot |\mathcal{X}|^2$ and $A(\boldsymbol{\theta})$ normalizes the distribution. The parameters $\boldsymbol{\theta} \in \mathbb{R}^d$ are obtained by Maximum Likelihood Estimation, whereby the Likelihood of a particular $\boldsymbol{\theta}$ is given by

$$\mathcal{L}(\boldsymbol{\theta}|\mathcal{T}) := \prod_{x \in \mathcal{T}} p_{\boldsymbol{\theta}}(\boldsymbol{x}). \tag{2}$$

Here, $\mathcal{T}$ is the data set which contains the training instances. If our model is implemented into a cellphone, we do not want it to store the users complete history of cell visits. Therefore, we take the logarithm of the Likelihood (2) and rearrange, such that it only depends on the average value or the *empirical expectation* $\tilde{\mathbb{E}}[\phi(\boldsymbol{x})]$ of our sufficient statistic.

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

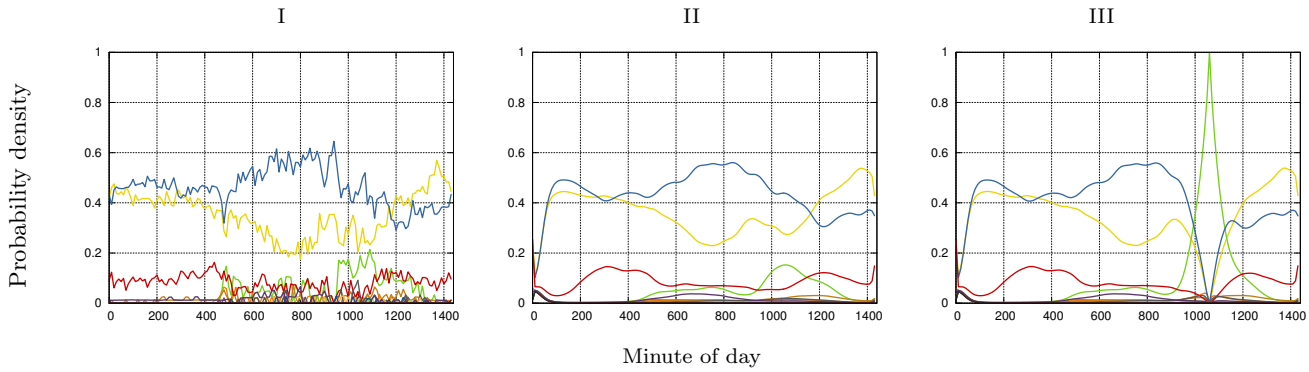

Figure 4: Temporal dynamics of empirical (I), estimated (II) and conditional (III) marginal probability densities for a fixed user (A) and all cells that the user has visited. Each color represents the probability for one cell over time. The densities in the right plot are conditioned on the single event that the green cell is observed at 5:40pm.

This basicly means that a phone has to store at most d integer values to represent the training data. Taking derivatives of (3) it follows that

$$\frac{\partial \ell(\boldsymbol{\theta}|\mathcal{T})}{\partial \boldsymbol{\theta}_\alpha} = \tilde{\mathbb{E}}\left[\phi_\alpha(\boldsymbol{x})\right] - \hat{\mathbb{E}}\left[\phi_\alpha(\boldsymbol{x})\right],$$

whereby α may be either a vertex or an edge assignment, e.g. $\alpha = (\boldsymbol{x}_t = y)$ for being in cell y at time t. The *estimated expectation* $\hat{\mathbb{E}}\left[\phi_{v_t}(\boldsymbol{x})\right]$ is equal to the marginal probability of the corresponding event, that is

$$\hat{\mathbb{E}}\left[\phi_\alpha(\boldsymbol{x})\right] = \hat{p}_{\boldsymbol{\theta}}(\alpha).$$

This value is computed by Belief Propagation [10] which reduces to a kind of forward-backward algorithm for this type of chain structured models. The objective $\max_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}|\mathcal{T})$ can then be solved by any first-order optimization method. We ran gradient descent optimization with an elastic net regularization [13] until convergence to find proper model parameters.

Our model uses T equidistant sampling points, hence the data had to be smoothend such that the time of each sampling point is projected onto the nearest multiple of ten minutes. Note that this simply moves the probability mass of a particular cell through time. In general, the model sampling rate depends on the data quality. If the probability between two sampling points is required, it is reasonable to apply an interpolation between sampling points. To generate continuous time density functions (Figure 4, I-III), the first-order interpolation

$$\bar{p}_{\boldsymbol{\theta}}(\boldsymbol{x}_t) = \hat{p}_{\boldsymbol{\theta}}(\boldsymbol{x}_{\lfloor t \rfloor}) + [\hat{p}_{\boldsymbol{\theta}}(\boldsymbol{x}_{\lceil t \rceil}) - \hat{p}_{\boldsymbol{\theta}}(\boldsymbol{x}_{\lfloor t \rfloor})](t - \lfloor t \rfloor), \qquad (4)$$

is applied whenever t is not integer. Because of the elastic net regularization, this interpolation results in smooth estimated density functions over time (II, III) when compared to the empirical density (I).

MRFs are generative models that use the data to compute the joint density of all sampling points. It is known that such models require a larger amount of training data compared to discriminative classifiers like SVMs. Accuracy strongly depends on the amount of traces covering the day, with a user A providing the most. Figure 5 shows the result in terms of accuracy and corresponding standard deviation of our discriminative SVM and our generative MRF models with $T = 144$. The time was ingnored for SVM model and predictions are performed like described in Section 3. Although this comparison ignores the benefit of formulating nearly arbitrary probabilistic queries for MRFs, it reveals the raw predictive power of both methods. The result was generated in a 10-fold cross validation on three users of the preprocessed MDC data (Section 2).

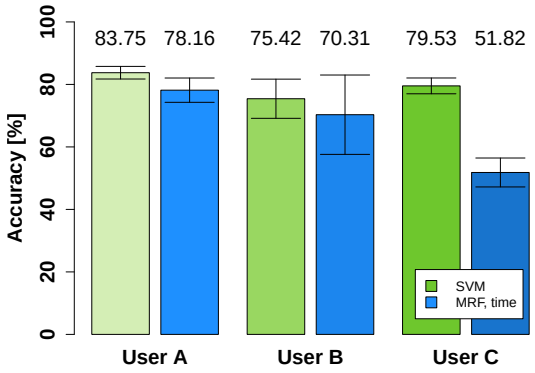

Figure 5: Prediction accuracy for specific users

Although the discriminative model shows a higher prediction performance, please notice the qualitative difference between both types of models: If we know that a user, say A, is in cell y at time t, we may ask the discriminative model for the most probable cell y' that follows y but not *when* this transition will occur. The generative model instead, will give us a probability density over all cells that the user could visit at time $t+1$, but also at $t+2$ and so on. In this way, we can compute the first time t' that satisfies $y \neq y'$ which is an estimate for the time of transition. If one tries to simulate this behavior with discriminative classifiers, she will need an infinite number of models. Depending on the actual task, one may also ask for the top-k most probable cells. While an answer to this question can be computed directly by a generative model, the number of discriminative classifiers required to answer this query is at least k per user.

The amount of data for user A was about 10 times higher than for users B and C. The difference in prediction performance of SVM and MRF, especially on users B and C, is due to the lack of data covering the days only partially.

Nevertheless, if our model is implemented directly into the device, the data can be continuosly collected to keep the empirical expectation and the model up-to-date.

There are some straightforward extensions of our generative approach like introducing higher order dependencys, e.g. connect vertices v_t and v_{t-2} $\forall t$, or incorporating future observations like calendar data. It is also possible to increase the prediction accuracy by building seperate models for weekdays, weekends and similar natural partitionings of time [4].

Furthermore, the generative model is able to compute the expected workload of each cell at each point in time as a byproduct. Let $\mathcal{U}_y$ be the set of all users which ever visited cell y and $\mathcal{U}_y(t)$ the set of all users which will visit cell y at time t. Hence, we have

$$\bar{\mathbb{E}}[|\mathcal{U}_y(t)|] = \sum_{u \in \mathcal{U}_y} 1 \cdot \bar{p}_{\boldsymbol{\theta}_u}(\boldsymbol{x}_t = y) + 0 \cdot \bar{p}_{\boldsymbol{\theta}_u}(\boldsymbol{x}_t \neq y), \quad (5)$$

where $\boldsymbol{\theta}_u$ are the model parameters which are learned for the user u and $\bar{\mathbb{E}}[|\mathcal{U}_y(t)|]$ is the expected number of users for cell y at time t. To compute this quantity in a collaborative manner, all users from the set $\mathcal{U}_y$ have to transmit their probability of being in cell y at time t to a central unit which performs the summation. Note that we use interpolated probabilities (4) in (5) to be able to compute an expectation for any time t. The expectation may also be conditioned on knowledge about the current locations of users or additional context information.

5. CONCLUSIONS AND OUTLOOK

Knowledge about the user's next cell provides a powerful tool for mobile network operators and manufacturers to balance load inside the network. Only minimal features, easily derivable in current mobile networks, are sufficient to train a machine learner and predict the location.

We demonstrated, that training a global model as well as for specific users both provides reasonable accuracy, with the possibility to exceed 80% for user based models. Time aspects could be integrated into the predictions via generative models, impacting the accuracy by only a few percent. The generative approach may be used to incorporate knowledge about future events and to estimate other network related quantities like the expected workload of each cell or the top-k most probable cells of each user.

Using available hardware in modern smartphones, it is possible to move nearly all required computations of both presented methods to many-core Graphics Processing Units [11, 2] to keep a smartphone's CPU free for regular phone related tasks. This enables to calculate and store models locally. Each smartphone could send, similar to link quality measurements, hints about the next expected cell to the network's base station. This approach still enables access to user specific models, but respects privacy without the need to store user profiles centrally, distributes the computation to each individual phone and saves storage space for the operator. Of course, local calculations performed on the phone will impact battery consumption. This effect on and optimization of energy will be investigated in future research.

6. ACKNOWLEDGMENTS

We thank the Nokia MDC team for the collection and preparation of the data set. Part of the work on this paper has been supported by Deutsche Forschungsgemeinschaft (DFG) within the Collaborative Research Center SFB 876 *Providing Information by Resource-Constrained Analysis*, A1: `http://sfb876.tu-dortmund.de`.

7. REFERENCES

[1] A. Bhattacharya and S. K. Das. Lezi-update: an information-theoretic framework for personal mobility tracking in pcs networks. *Wirel. Netw.*, 8(2/3):121–135, March 2002.

[2] A. Carpenter. cuSVM: a CUDA implementation of support vector classification and regression. 2009.

[3] C. Cheng, R. Jain, and E. van den Berg. Wireless internet handbook. chapter Location prediction algorithms for mobile wireless systems, pages 245–263. CRC Press, Inc., Boca Raton, FL, USA, 2003.

[4] H. Hohwald, E. Frías-Martínez, and N. Oliver. User modeling for telecommunication applications: experiences and practical implications. In *Proceedings of the 18th international conference on User Modeling, Adaptation, and Personalization*, UMAP'10, pages 327–338, Berlin, Heidelberg, 2010. Springer-Verlag.

[5] T. Joachims. Making large-scale svm learning practical. LS8-Report 24, Universitat Dortmund, LS VIII-Report, 1998.

[6] J. K. Laurila, D. Gatica-Perez, I. Aad, T.-M.-T. D. Jan Blom, Olivier Bornet, O. Dousse, J. Eberle, and M. Miettinen. The mobile data challenge: Big data for mobile computing research. In *Mobile Data Challenge by Nokia Workshop, in conjunction with Int. Conf.. on Pervasive Computing*, 2012.

[7] S. Michaelis. Balancing high-load scenarios with next cell predictions and mobility pattern recognition. In *International Conference on Mobile Services, Resources, and Users (MOBILITY 2011)*, Barcelona, Spain, October 2011.

[8] S. Michaelis, A. Lewandowski, K. Daniel, and C. Wietfeld. Macromobility prognosis for high-priority resource reservation in wireless networks. In *5th IEEE International Symposium on Wireless Communication Systems (ISWCS)*, pages 184–188, Reykjavik, 2008.

[9] I. Mierswa, M. Wurst, R. Klinkenberg, M. Scholz, and T. Euler. YALE: Rapid Prototyping for Complex Data Mining Tasks. In T. Eliassi-Rad, L. H. Ungar, M. Craven, and D. Gunopulos, editors, *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2006)*, pages 935–940, New York, USA, 2006.

[10] J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.

[11] N. Piatkowski. Parallel algorithms for gpu accelerated probabilistic inference. *NIPS Workshop on Big Learning 2011*, 2011.

[12] M. J. Wainwright and M. I. Jordan. Graphical Models, Exponential Families, and Variational Inference. *Foundations and Trends in Machine Learning*, 1(1–2):1–305, 2007.

[13] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society B*, 67:301–320, 2005.