Unsupervised, Dynamic Identification of Physiological and Activity Context in Wearable Computing

Andreas Krause
Munich University of Technology
krausea@cs.tum.edu

Daniel P. Siewiorek
Carnegie Mellon University
dps@cs.cmu.edu

Asim Smailagic
Carnegie Mellon University
asim@cs.cmu.edu

Jonny Farringdon
BodyMedia Inc.
jonny@bodymedia.com

Abstract

Context-aware computing describes the situation where a wearable / mobile computer is aware of its user's state and surroundings and modifies its behavior based on this information. We designed, implemented and evaluated a wearable system which can determine typical user context and context transition probabilities online and without external supervision. The system relies on techniques from machine learning, statistical analysis and graph algorithms. It can be used for online classification and prediction. Our results indicate the power of our method to determine a meaningful user context model while only requiring data from a comfortable physiological sensor device.

1. Introduction

Our goal is to make the wearable computer sense the user and his current state, exploiting context information to significantly reduce demands on human attention [1]. By estimating physiological properties like stress level, activity level, movement patterns and ambient context information, a human's current state can be predicted, e.g. if one is currently busy or not. We do not try to identify detailed scenes like shopping at a supermarket, or entering or leaving a building. An approach towards scene identification can be found in [2].

Microphone data has been used in [3] to identify whether the user of a cell phone can be interrupted or not. In [4], accelerometer data has been used to identify different kinds of activities like walking, running, climbing stairs. All these approaches rely on offline data analysis to learn "typical" contexts.

A method based on Kohonen Self Organizing Maps (KSOMs) and *k*-Means clustering is proposed in [5], which is able to identify typical motion profiles. This approach relies on active training, i.e. pushing buttons while performing certain activities. These sparse labels are used to construct a supervised context transition profile based on a first order Markov process. This approach also suffers from the KSOM's inflexible memorizing process [14]. To make the KSOM training procedure converge, the neighborhood radius of the learning neurons must decrease over time [9]. This results in an inability to keep up the learning process running over a long time period.

In [6], an online algorithm is given for the combination of KSOMs and *k*-Means clustering. This approach also requires active training by specification of user labels.

For context-aware systems to eventually become end-user products, both wearability (i.e. non-intrusiveness) and usability (minimal active training) are the key issues. Our objective is to make advances in both directions. For the wearability issue, we make use of an off-the-shelf, minimally intrusive armband. This armband can be placed on the back of the upper arm, and can communicate wirelessly with the other computers.

Our motivation is derived from the observation that context does not require a descriptive label to be used for adaptivity and contextually sensitive response. This makes an approach towards completely unsupervised learning feasible. By unsupervised learning we mean the identification of context without requiring exterior supervision, i.e. by manually annotating current user states. Most unsupervised machine learning techniques can be subsumed by the term clustering, i.e. density estimation in the underlying probability space. In general for clustering, it is difficult to tell whether two close (in terms of the feature space distance metrics) cluster

centers are actually part of the same cluster. This is the challenge of context abstraction.

In Section 2, we describe the wearable platform and Section 3 explains our method. In Section 4, we describe our experimental setup and evaluate the results of our studies. In Section 5, we identify two applications that can use our method, and Section 6 presents our conclusions.

2. Wearable platform

Figure 1. Wearing position and form factor of the SenseWear armband.

To provide sensor input, our system relies on physiological and movement data. We use the SenseWear armband from BodyMedia as shown in Figure 1. Information about wearable sensor badges can be found in [7]. The small armband combines five distinct sensors: two accelerometers, galvanic skin response, skin temperature and near-body ambient temperature. The armband is worn on the back of the upper arm. The sensors are described regarding this position, with the wearer standing up, arms at their side.

A 2-axis micro-electro-mechanical sensor device measures motion. These accelerometers are oriented in the transverse (through the chest parallel to the ground) and longitudinal (vertical to the ground along the arm, head-to-toe) planes. The motion can be mapped to forces exerted by the body and hence contribute to calculations of energy expenditure. By taking into account gravity they give orientation information as well as motion information, which are used to predict the wearer's context.

Heat flux is a measure of the amount of heat being dissipated by the body. The sensor uses very low thermally resistant materials and extremely sensitive thermocouple arrays. It is placed in a thermally conductive path between the skin and the side of the armband exposed to the environment.

Galvanic skin response (GSR) represents electrical conductivity between two points on the wearer's arm. The GSR sensor includes two hypo-allergenic stainless steel electrodes integrated into the underside of the armband. Skin conductivity is affected by the sweat from physical activity and by emotional stimuli. GSR can be used as an indicator of evaporative heat loss by identifying the onset, peak, and recovery of maximal sweat rates.

Skin temperature is measured using a highly accurate thermistor-based sensor located on the backside of the armband near its edges and in contact with the skin. Continuously measured skin temperature reflects the body's core temperature activity.

The near-body ambient temperature sensor measures the air temperature immediately around the wearer's armband. This sensor also uses a highly accurate thermistor-based sensor and directly reflects the change of environmental conditions around the armband, e.g. walking out of an air conditioned building on a hot day.

In addition to the integrated sensors, the SenseWear has a Polar® wireless receiver for heart beat detection from compatible chest straps. The receiver board includes a free running 8 kHz timer derived from the crystal controlled microprocessor clock that is accurate to as low as 50 beats per minute. Heart rate and energy expenditure exhibit a linear relationship, particularly between a heart rate of 110 and 150 beats per minute. Heart rate can also be used as an aid in distinguishing frequency, intensity, and duration of activity.

The SenseWear is worn on the back of the upper right arm for numerous reasons. Comfort is key, as the armband's normal use is 24 hours a day 7 days a week. It can be worn during the day and at night, only being taken off to wash. The position at the back of the upper arm, in combination with the armband shape makes it unobtrusive and quickly "unnoticeable". [8] Additionally, the a-symmetric heat flux sensor gives the most accurate readings when worn in this position.

The user sets a recording frequency. At one minute data recording intervals the armband can record data for over 5 days, 24 hours a day without recharging. Each of the integrated sensors is sampled 32 times a second regardless of the sample interval setting. Once a sampling period two measures of the sensors – the mean value, and the sum of the absolute differences (SADs) – are calculated. The heart-beat detection is a special case, each heart beat received is recorded in the armband and time stamped to the nearest 1000^{th} of a second. Heart rate is continuously calculated.

Sensor readings recorded in the armband are direct values from the internal analogue-digital (AD) converter, in the range [0, 4095]. Each armband contains calibration information for each sensor unique to the armband, determined during its manufacture.

The modus operandi of the SenseWear armband is continuous wearing and data collection, with a data download once a day or week. The armbands also contain a transceiver for data transmission using the Scientific and Medical Instruments (ISM) license free band. The recharging and docking cradle for the PC contains a corresponding transceiver. Within range of the cradle the armband can transmit live data readings for processing.

The armband transceiver operates on a "no unrequested broadcasts" principle. Under some circumstances wireless broadcasting may be inappropriate, for example in a hospital, or on an airplane. The armband is always listening for wireless communications. Each armband is identified by an eight digit serial number, and wireless data requests ask specific armbands to communicate.

Two wireless applications are routinely used in context and real-time physiological research. The first is request-based and the second is "continuous". When request based, a specific armband is asked to return a sample from specified sensors. Data sampling rates of up to four samples per second are possible with this method. The continuous method instructs an armband to start broadcasting specified sensor readings as fast as possible. The armband stops broadcasting when it is taken off-body, a state it automatically detects as being no longer in contact with skin. Data rates up to sixteen samples a second can be achieved with this method.

A button on the armband allows collecting time-stamps which can be used for annotating recorded data.

For data processing, an Acer C100 Tablet PC with a 900 MHz Pentium III and 256 MB of main memory was preferred to a Personal Digital Assistant (PDA) sized device, since the tablet platform has a larger screen for data visualization but can still be carried around easily. Although the Tablet PC can communicate wirelessly with the sensor, currently only recorded and downloaded data has been studied, which is input to the data processing software in a way to simulate real-time sensor queries. Figure 2 shows the devices included in the wearable platform.

3. Method

In this Section we describe our unsupervised machine learning approach. We first present an offline algorithm which uses a training sample time series as input and constructs a classifier from this data. Successively, we describe how we turn our offline algorithm into an online algorithm, which can use real-time sampling data.

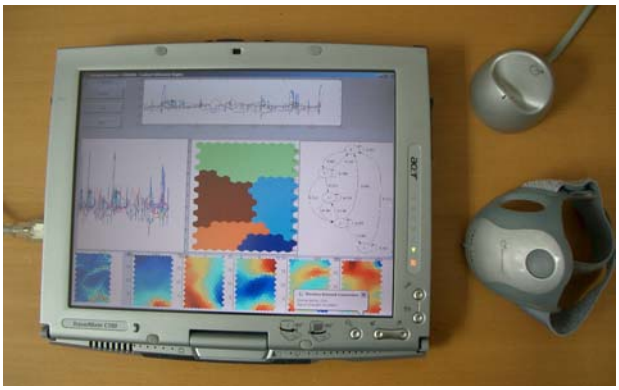

Figure 2. Wearable platform. The Acer C100 Tablet PC is used for data analysis and can communicate wirelessly and in real-time with the sensor armband on the lower right via the transceiver cradle on the upper right.

Our offline approach can be summarized as follows: After certain data preprocessing, a Kohonen Self Organizing Map is trained with the sensor values. A *k*-Means clustering is performed on the maps codebook vectors, where *k* is chosen as guided by the Davies Bouldin index; a description and analysis of this clustering quality measure can be found in [17]. After the clustering, a first order Markov model is trained with respect to the cluster transition probabilities. A graph reduction strategy is applied to the resulting transition graph. This graph reduction eliminates transient states. Codebook vectors for eliminated states are reassigned to non-transient states by further iterations of the *k*-Means algorithm. The resulting clustering is used as a classifier for test samples; the first order Markov model on the final clustering can serve for prediction. Figure 3 gives a graphical illustration of our proposed system.

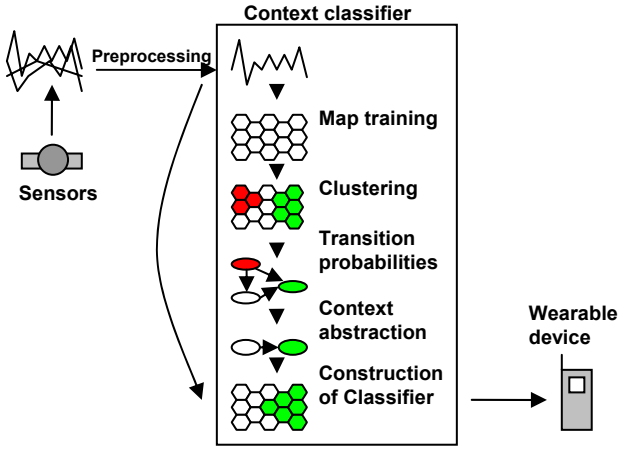

Figure 3. System diagram. Data is gathered by a wearable sensor. The data is used both to construct the classifier and to classify current sensor readings. The resulting classification is sent to a context aware wearable device.

We now give more details on the construction of the classifier described above. In the remaining part of this Section, we describe how we changed our offline algorithm into an online algorithm. In Section 4, we analyze the performance of our method with respect to sensor data from several studies.

3.1 Preprocessing

Since we experimented with varying sampling frequencies, different preprocessing techniques were chosen. For low data rate polling (1 and 10 samples per minute), the onboard averages and absolute differences proved to be useful. For high data rate polling (8 samples per second) we calculated the Fast Fourier Transform (FFT) oscillatory spectra of the recorded accelerometer values. It turned out that a 64 point FFT (i.e. spectra of 8 second time-frames, separate spectra for both axes) is a good resolution to discern various different fine-grain movement patterns. The resulting spectra have been log-transformed. In the low data rate studies, all 8 data channels of the BodyMedia device have been recorded, forming an 8 dimensional feature space. For the high data rate samples, the spectra of the two accelerometers were recorded, making it a 128 dimensional feature space.

To give each dimension equal importance in terms of the self organization process, we normalized the data to have unit coordinate-wise variance as proposed in [9].

Since learning in high dimensional feature spaces is adversely affected by the so called curse of dimensionality [10], techniques for dimension reduction were evaluated.

Commonly used candidates for this purpose are Principle Component Analysis (PCA) and linear Independent Component Analysis (ICA). ICA is computationally expensive and therefore currently not feasible for online algorithms on wearable computers. Therefore, PCA, which is a projection of the data vectors to the most important eigenspaces (in terms of eigenvalue size) of the covariance matrix, was evaluated. This approach shows good results on our training data. Figure 4 shows the amount of variance explained by the top eigenspaces for both the low data rate and high data rate approach. Figure 5 shows two recorded time-series, normalized to unit variance.

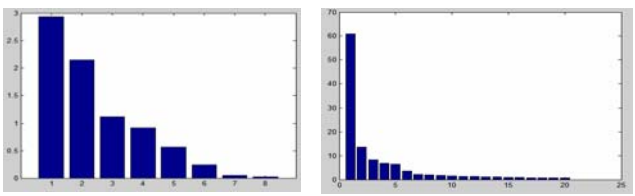

Figure 4. PCA. The bar charts display eigenvalue sizes of the low rate (left) and high rate (right) samples.

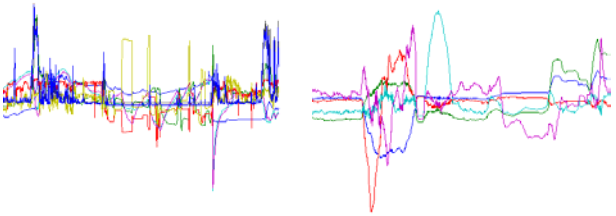

Figure 5. Time series. Plot of time series resulting from 20 hours of data at 1 sample per minute (left) and from 3 minutes of 8 samples per second data.

3.2 Further dimension reduction by Kohonen Self Organizing Maps

In both the high and the low data rate cases, the top five eigenspaces appear to explain most of the sensor variance, so the input data was projected onto the first five principle components. Since the topology of the Kohonen maps is typically two-dimensional (it can be seen as a two dimensional manifold in the higher dimensional feature space), it is expected that clusters in a lower dimensional feature space can be found more easily than in a higher dimensional feature space. We used a 20x23 cell hexagonal topology for the map, a size suggested by [11] for our data size. Training is performed by the batch training algorithm described in [9].

3.3 Clustering

Although the training of the self organizing map is already a clustering process, codebook vectors must be grouped together further to reach representative cluster sizes. The U-Matrix is a means of detecting clusters and emerging features [16]. It contains information about the feature space distance of codebook vectors adjacent with respect to the map topology. Figure 12 shows an example of a U-Matrix. Clustering based solely on the segmentation of the U-Matrix is difficult since emerging clusters rarely have sharp borders. As suggested in [5], a *k*-Means clustering is performed on the codebook vectors. Our approach does not require user labels for determining the initial cluster centroid starting values; the clustering is completely driven by the unlabeled data. We initially perform a *k*-Means clustering with quite large *k* (growing as the square root of the number of codebook values is a common heuristics). Clustering is performed for the *k* values lying in a certain interval. The Davies-Bouldin index is used as a further guidance.

3.4 Context abstraction

By this procedure, about 14 clusters are found for a typical low data rate sample for 20 hours of training data.

Since the chosen data preprocessing is expected to smooth out abrupt context changes, clusters representing these context changes are expected to be found (i.e. switching from running to walking and back). Although these transition states are interesting in terms of context change prediction, they adversely affect learning adaptive system behavior, since the system must rely on robust context state estimates as the basis of learning usage patterns. To identify transient states, transition probabilities are taken into account. Based on the clustering described above, a first order Markov model is trained based on the cluster transition probabilities. Transient states can be characterized by having a low "loop" probability, namely remaining in the same state for a long time is improbable. A graph reduction algorithm is performed on the Markov model, eliminating states with a "loop" probability (the diagonal values in the transition matrix) below a certain threshold. The corresponding transitive corrections including the update of the transition probabilities are implemented as shown in Figure 6. Each state v with a loop probability below a certain threshold α is removed from the transition graph; 0.3 turned out to be a good value although results do not show dramatic changes for variations of this value between 0.2 and 0.4. The main step in the algorithm is the following transitivity update,

`P(r,c) ← P(r,c) + P(r,v) * P(v,c) / S`

which redistributes the probabilities of out-adjacent edges of v to corresponding transitive out-adjacent edges of v's predecessors. P denotes the state transition probability matrix, i.e. $P(i,j) = \text{Prob}(X_t = j \mid X_{t-1} = i)$, where X_t is the cluster index assigned to the sensor value vector at time t, $\text{Prob}(X|Y)$ is the conditional probability of X for the prior Y.

```
For v = 1 to #states
  If P(v,v) < α then
    P(v,v) ← 0;
    S ← Σi P(v,i);
    For r = 1 to #states, r ≠ v
      For c = 1 to #states, c ≠ v
        P(r,c) ← P(r,c)+P(r,v)*P(v,c)/S;
      End
      P(r,c) ← 0;
    End
  End
End
```

Figure 6. Pseudo code for elimination of transient vertices.

After the elimination of the transient clusters, a few more iterations of the k-Means algorithm are necessary to reassign now unclustered codebook vectors to new clusters. Successively, a new transition probability Markov model is trained on the reduced set of clusters; this model can then be used to predict possible context changes. Figures 7 and 8 show the elimination of transient states. For visualization of the transition graphs, the GraphViz package from [12] was used.

3.5 Online algorithm

The algorithm described above is offline, i.e. the entire sensor values time-series must be available at training time. Our online algorithm is buffer based, and it proceeds as follows: sensor values are recorded until a certain time interval (i.e. some hours or days) has passed.

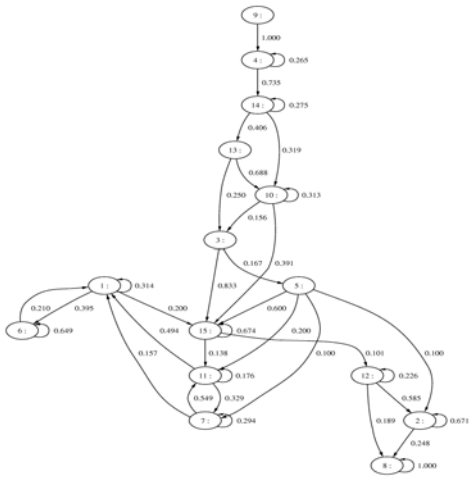

Figure 7. Transition probability graph before elimination of transient states.

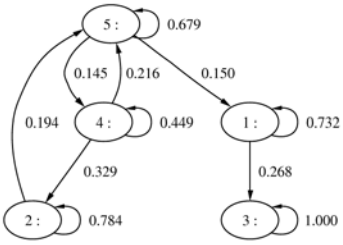

Figure 8. Transition probability graph after elimination of transient states.

This data is taken to construct a classifier and context transition probability model as described above. To refine this model, a training sample is constructed, consisting of the compressed prior training sample and the newly buffered data. The prior training sample is compressed by selecting a certain percentage of its data vectors at equally spaced intervals. Using these training samples, another map is trained, and the new codebook vectors are again clustered by a k-Means algorithm with the existing cluster centers as initial values plus a certain number of randomly initialized cluster centers. Successively the steps described above (building a Markov model, do graph reduction, build another Markov model) are

applied again. By adding these randomly initialized cluster centers, potentially new context can be identified. Figure 9 illustrates the construction of this online classifier.

Since the empirical covariance matrix changes as new sensor data becomes available, PCA can only be performed for fixed sets of data. We consider the principle component directions as properties of the sensors and the sensor placement. Therefore we expect that data from a limited time interval can determine an initial estimate of the principle component direction. The linear transformation determined by these directions can be used to project new data in real-time to the previously determined principle components. Recalibration can be triggered, as soon as the amount of variance explained by the principle components drops beneath a certain threshold.

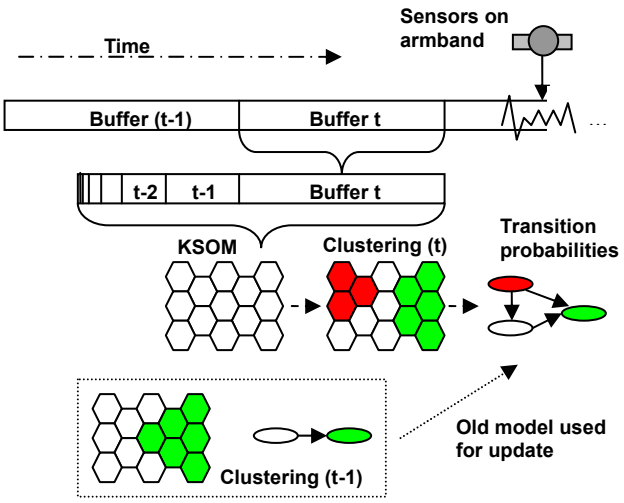

Figure 9. Online algorithm: The top row represents the buffering of the input data. In the second row, the geometric memory decay is sketched. The third row shows the training of the KSOM, the clustering, the Markov model and the integration of the prior clustering.

By buffering a certain fraction of the past sensor data we can still keep track of important past clusters, while maintaining the KSOM's ability to learn new contexts. Since the fraction is fixed, the size of the learning data converges. This memory decay strategy is geometric in nature, which is the discrete version of exponential decay.

The motivation of this buffered learning strategy comes from the human ability to process daytime experiences over night. Another advantage of the buffer based update is that the computationally expensive training and clustering procedure can be implemented in parallel to the data acquisition, classification and prediction algorithms.

A prototype of the online algorithm described above was built in Matlab 6.5, using the SOM Toolbox 2.0 [13]. Figure 11 shows the graphical user interface of the system. At each time step, the signal values are projected onto the map topology and visualized in the clustering diagram, which determines the output of our classifier.

4. Results and evaluation

To evaluate our method, two studies have been made. In the first study, two subjects wore the BodyMedia armband over the duration of several days, annotating their determined context – i.e. working in the office, being in a meeting, commuting, relaxing in the sun – by use of the armband button and additional note-taking.

Subject A is one of the authors, a graduate student in his early twenties leading a moderately active life style. Subject B is a graduate student in her mid twenties, leading an active life style. The subjects were required to mark time-stamps close to their subjectively felt context change and verbally classify the corresponding context. In summary, over 240 hour long studies were done by recording one sample per minute (low data rate).

The second study investigates other properties of our classifier. We look at higher data rate samples, performance of our classifier to determine fine-grain motion, stability and computational properties. To determine how our method performs with higher sampling frequency data we collected data with different sampling settings: One data set was sampled at 10 samples per minute, another one at 8 samples per second. All data samples were recorded at specified sampling rates on the armband's onboard storage. The collected data was downloaded to the Tablet PC via the SenseWear docking cradle at varying time intervals.

4.1. Results of Study 1

Subject A collected data for 4 days for an overall recording time of 74 hours, subject B gathered data during daytime for 22 days with a total recording time of 168 hours. The sampling rate was set to one sample per minute. Data was analyzed in blocks of varying lengths, including a variety of different contexts. Per block, an initial clustering with on average 15 clusters was done. After the elimination of transient states, an average of five clusters were identified for subject A, who continuously wore the device for four days, and an average of four clusters were identified for subject B, who had varying recording time ranges. All recordings were done during weekdays.

The data was processed by our classifier and the resulting clustering was manually compared with the data annotation. It was examined whether equally annotated contexts correspond to single or separate clusters. Figure 13 displays the performance of our method on a 20 hour

data sample. Our method could compete with the subjective context classification well up to a certain extent. While subject A walked home every day – a context which could clearly be identified, subject B commuted, which was rarely identified; in most cases it was typically misclassified as the "working in office" context. However, we did not use any temporal context in our studies, and especially activities like commuting are expected to correlate well with temporal information. The study shows that both subjects only identified a small number of subjectively typical contexts; the number of contexts annotated is only slightly higher than the number of clusters found by our classifier. The frequent use of the armband's timestamp button by subject B indicates the existence of a number of minor contexts which could not be identified by our current algorithm. The number of state transitions our classifier determines matches the number of timestamps taken by both subjects well, so the non-identified clusters might only have been eliminated by a too strict context abstraction threshold.

Table 1 shows the results of Study 1. Col. 1 is the sample number, Col. 2 denotes recording duration and involved days, Col. 3 lists abbreviations for major contexts identified by wearer, Col. 4 shows the number of clusters found, Col. 5 gives the number of time stamps, Col. 6 the number of stable (only one transition per five minutes counted) transitions. Col. 7 gives the offline map training quantization error. The following abbreviations are used: B – Busy, C – Commute, D – Driving, E – Eating, F – Fitness, H – at Home, L – Social, M – Meeting, N – Not Busy, O – Office, R – Relaxing, S – Sleeping.

Consistent with [8], both subjects reported the unobtrusiveness of the wearable sensor badge, especially for long term usage. The data from Study 1 was both input to the offline and the online algorithm which showed comparable classification performance.

4.2. Results of Study 2

To evaluate the sensor data quality and the performance of our methods we performed analyses from different perspectives: Firstly, we looked at higher data rate samples (4.2.1., 4.2.2.), secondly we assessed sensor quality by a cross-validation (Section 4.2.3.), and thirdly we measured computational performance and memory requirements (Section 4.2.4.).

Two higher data rate data sets were collected. A movement study at eight samples per second was used to test the quality of our method to discern fine grain movement patterns (Section 4.2.1.). A 24 hour data set at 10 samples a minute was collected to estimate stability of our online algorithm and to determine the impact of higher resolution time series data (Section 4.2.2.).

4.2.1. Stability study. Subject A wore the armband for another two days, with the sampling rate set to 10 samples per minute. The comparison of both low data rate sensor recordings indicate that the availability of more data samples at the higher sampling rate of 10 samples a minute makes the clustering more stable. A reason is that less time samples are associated with transient states as are for the one sample per minute sampling rate. A more detailed clustering is found, resulting in an average of eight stable clusters for a 24 hour period.

To evaluate the geometric decay memory of our online algorithm we input a typical 24 hour sample (11002 data vectors) to the online algorithm, repeated three times. The buffer was sized to capture the most recent four hours of data and four hours of compressed prior information; the algorithm showed stability of the clustering for this experiment.

The overall quantization error with the dynamically updated map was 0.42 after the one day sample was repeated three times. The overall quantization error for the offline map was 0.37 after the one day sample. This indicates that, although the buffer size was only 12 % of the complete sample size, the quantization error of the map constructed online was only 10% larger than the quantization error of the offline map, for which the complete sample was used while training. Figure 10 shows the influence of the buffer size on the quantization error.

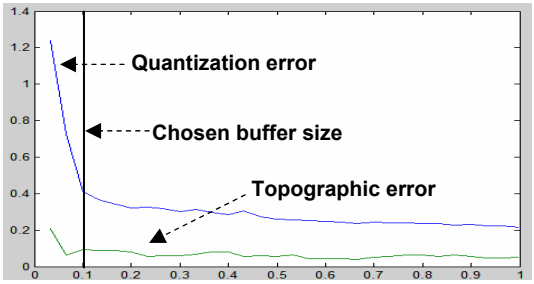

Figure 10. Buffer size vs. quantization and topographic error

4.2.2. Movement study. Subject A performed several distinct kinds of movements over a period of about three minutes (1632 samples in total), in the following order: Walking (1), running (2), walking (1), sitting (3), knee-bends (4), walking (1), waving his arms (5), walking (1), climbing up and down stairs, one stair a step (6). Six clusters were identified, clearly corresponding to the trained movement patterns. Projection of adjacent values of the sensor time series onto the map exhibited close locality while the subject performed movements corresponding to a certain activity.

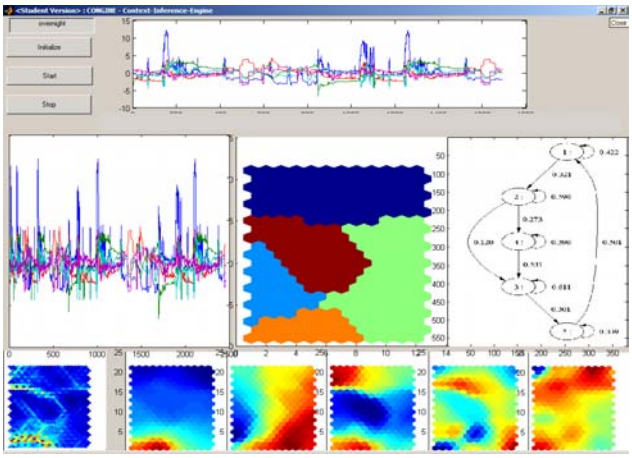

Figure 11. Graphical user interface of the online classifier. The top row shows the sensor data and is updated in real time. The second row displays the geometric memory decay buffer (left), the reduced Markov model graph (right) and the corresponding clustering (center). In the third row, the U-Matrix is displayed on the left, along with values of the codebook vector coordinates for the top five principle components.

The map constructed by the offline training algorithm was evaluated by testing its performance as a classifier on test data, which consisted on recorded movement patterns of four minutes, 2204 samples in total. Movements were recorded in the following order: walking slowly (1,3), knee-bends (4), running (2), climbing up and down stairs, two stairs a step (6), walking (1) and sitting (3). All movements were correctly classified as indicated by the numbers in brackets. The algorithm was very successful even in discerning the transition from running to climbing stairs; the classification of walking slowly was wrong only in the first few seconds, when it was misclassified as corresponding to the "sitting" cluster. Since the FFT is calculated from an eight second time frame, the transition lag varied between 0 and 4 seconds, 1 second being the average classification lag. Figure 12 shows the resulting clustering of the high data rate sample data.

4.2.3. Cross-Validation of the sensors. To assess the importance of all eight sensor data channels for stable clustering, a cross-validation was performed. Using a 20 hour data sample, a map was trained for all data channels enabled, and for disabling either the GSR (i), the heat flux (ii), ambient temperature (iii), skin temperature (iv), both accelerometer SAD channels (v), both accelerometer mean values (vi) and SAD and mean of the longitudinal (vii) and the transverse accelerometer (viii). The most detailed clustering was found in (v). The accelerometer SAD values tended to produce many transient clusters. In all observations with SAD values enabled, only four or five clusters were found; in case (v), six clusters were found. Case (v) had problems with identifying the sleep state, which can be characterized by an extremely low level of accelerometer change. The sleeping state was characterized best in case (iv), where accelerometer SAD values were available and the skin temperature was disabled. Figure 14 shows the influence of the fluctuating skin temperature and the accelerometer SAD values on the detection of the sleeping state.

The topographic error of the map training process was smallest for the exclusion of accelerometer means or SAD values, as shown in Figure 15. This indicates that the accelerometer values have great impact on cluster variety. A definition and discussion of the topographical error can be found in [9]. The overall result is that each sensor value contributes towards a meaningful clustering, each sensor characterizing certain contexts better than others, but possibly also inhibit the characterization of other contexts. Incorporating time series information could help to determine irregular periodicities in the data which could be used to reveal more detailed context information.

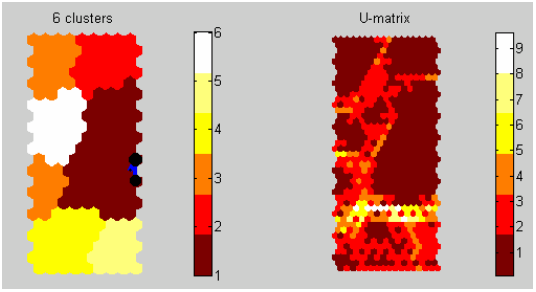

Figure 12. Results of our offline analysis of the high sample rate movement data. The left image shows the clustering, based on the U-Matrix (right).

4.2.4. Computational performance results. The buffer update procedure of training and re-clustering took on average 68 seconds on the current implementation for the 4800 data point buffer with 50 percent geometric memory; classification and sensor buffering works in real-time for the low-data rate samples. Since the current system has been implemented on Matlab which is an interpreted language, the update procedure is expected to perform well if optimized for current wearable computer processors as the StrongArm with 400 MHz.

Since only the map and the buffered data must be stored in memory, the memory requirements are rather small; for a 24 hour buffer at the 10 samples per minute data rate, 1.2 Megabytes are enough for storing the buffer.

Table 1. Results of Study 1.

Sample	*Size*	*Annotated Contexts*	*Clusters*	*Timestamps*	*Transitions*	*Quant. Error*
A-1	20h / 2d	C,E,H,O,S	6	9	11	0.5
A-2	25h / 2d	C, E,F,H,O,R,S	6	9	14	0.3
A-3	29h / 2d	C,E,H,O,S	5	8	17	0.4
B-1	57h / 6d	B,C,H,M,O,S	4	26	35	0.6
B-2	17h / 3d	M,O	2	20	25	0.5
B-3	26h / 4d	C,M,N,O,S	4	18	24	0.6
B-4	22h / 3d	C,D,E,L,O,M	4	25	27	0.8
B-5	46h / 5d	C,E,L,M,O	3	37	35	0.9

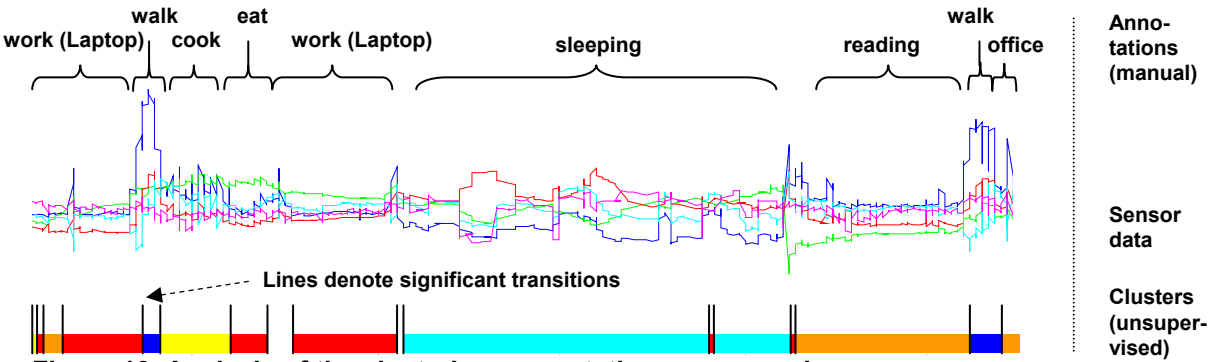

Figure 13. Analysis of the clustering – annotation correspondence

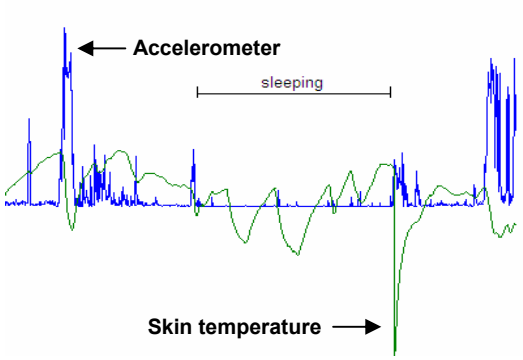

Figure 14. Unit variance normalized raw sensor data of the skin temperature (green) sensor, the longitudinal accelerometers SAD value and their characterization of the sleeping context.

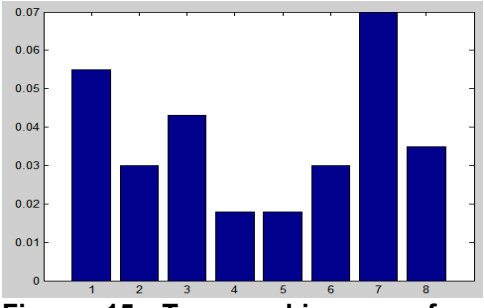

Figure 15. Topographic error for the cross-validation scenarios.

5. Intended applications and future work

To assess the practical suitability of our context inference method, we intend to employ it in two applications. The first one will be SenSay, a context-aware cell phone, developed at Carnegie Mellon University. Our method will be used to identify typical user context like attending a meeting, having a conversation, being idle or being highly active. These states can be the basis for learning adaptive behavior, like proactive configuration of the ringer volume and vibration alert. Our system concept is illustrated in Figure 16. The other intended application is in contextual car driver interface. We will evaluate how our system will perform in predicting the driver's current cognitive load, e.g. to decline incoming phone calls in a busy situation.

Our next steps will be twofold: firstly we will investigate the use of other sensor qualities, i.e. light sensor, ambient and voice microphones to allow more detailed description of ambient context. Secondly, our effort will address the incorporation of temporal context in the clustering procedure. This includes both using time as sensor values (day, week rhythm) and incorporating time series information in the clustering process. Extensions to the KSOM algorithms like Recurring SOMs [15] will be investigated.

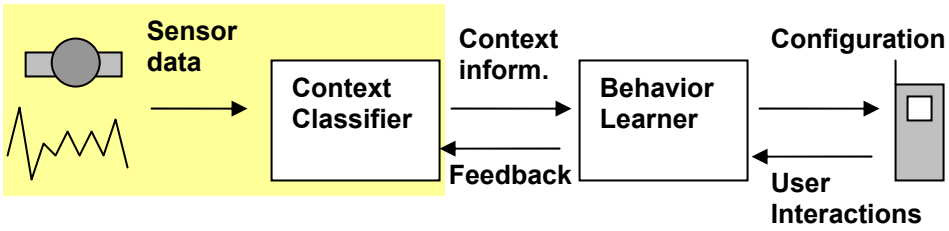

Figure 16. Intended system design for the context aware cell phone application. The highlighted part is described in this paper. A behavior learning algorithm will learn from user interactions with the phone and identify usage patterns from that data, which are used to configure the cell phone. The behavior learner can pass feedback to the classifier, e.g. to control sensitivity of context abstraction.

Since context can be looked at from different abstraction levels – i.e. running as a fine grain movement pattern and high activity as high level context description – a hierarchical context model is appropriate. We will examine a hierarchical system of SOMs and clusterings, which will be constructed by training with data from various sampling rates. We hope that this approach will help us identify a suitable model for context abstraction, in addition to the graph reduction method already employed.

6. Conclusions

We have presented a method for unsupervised and dynamic identification of physiological and activity context in wearable computing. Studies indicate the usefulness of physiological properties for context identification. We developed, implemented and evaluated a system which can identify a set of context states without requiring supervision, based on data from a comfortable wearable sensor armband. Our improvements in wearability and usability for context-awareness are advances towards making context-aware wearable computers feasible for end-user products.

7. References

[1] A. Smailagic, D.P. Siewiorek "Application Design for Wearable and Context-Aware Computers", *IEEE Pervasive Computing* Vol. 1 No. 4 Dec. 2002 Pp. 20-29.

[2] B. Clarkson, "Life Patterns: structure from wearable sensors" *PhD Thesis, 2002, MIT Media Lab*

[3] Hudson, S.E, Fogarty, J., Atkeson, C.G., Avrahami, D., Forlizzi, J., Kiesler, S., Lee, J.C., and Yang, J. (2003). "Predicting Human Interruptibility with Sensors: A Wizard of Oz Feasibility Study." To Appear, in proc. *SIGCHI Conference on Human Factors in Computing Systems* (CHI 2003).

[4] R.W. DeVaul, S. Dunn "Real-Time Motion Classification for Wearable Computing Applications", 2001, Project paper, *http://www.media.mit.edu/wearables/mithril/realtime.pdf*

[5] K. VanLaerhoven, Ozan Cakmakci "What Shall We Teach Our Pants", In proc. *The Fourth International Symposium on Wearable Computers*. IEEE. Atlanta, GA, USA, October 18-21 2000

[6] K. Van Laerhoven. "Combining the Kohonen Self-Organizing Map and K-Means for On-line Classification of Sensordata". *Artificial Neural Networks - ICANN 2001*, G. Dorffner, H. Bischof & K. Hornik (Eds.), Vienna, Austria; Lecture Notes in Artificial Intelligence; Vol. 2130, ISBN 3-540-42486-5; Springer Verlag, 2001, pp.464-470

[7] J. Farringdon, A.J. Moore, N. Tilbury, J. Church, P.D. Biemond. "Wearable Sensor Badge & Sensor Jacket for Context Awareness." In proc. *The Third International Symposium on Wearable Computers*. IEEE. San Francisco, CA, USA. October 18-19 1999, pp 77-83

[8] F. Gemperle, I. Stivoric, C. Kasaback et al. "Design for Wearability" In proc. *The Second International Symposium on Wearable Computers*. IEEE. Pittsburgh, PA, USA. Pp. 116-122

[9] Kohonen, T., "Self-Organizing Map", 2nd ed., Springer-Verlag, Berlin, 1995, pp. 127-128.

[10] R. Bellman, Adaptive Control Processes, Princeton University Press (1961)

[11] Kohonen, T., "Things you haven't heard about the Self-Organizing Map", in proc. *International Conference on Neural Networks (ICNN)*, San Francisco, 1993, pp. 1147-1156.

[12] E.R. Gansner, S.C. North "An open graph visualization system and its applications to software engineering", *Software – Practice and Experience* 00 (S1), 1-5 (1999)

[13] J. Vesanto, E. Alhoniemi, et al, "Self-Organizing Map for Data Mining in MATLAB: the KSOM Toolbox", *Simulation News Europe*, ARGE Simulation News 25 March (1999), Pg. 54

[14] R.M. French, "Catastrophic Forgetting in Connectionist Networks" *Trends in Cognitive Sciences, 3*(4), 1999, pp. 128-135.

[15] T. Koskela, M. Varsta, J. Heikkonen, and K. Kaski. "Temporal sequence processing using recurrent KSOM" In proc. *KES'98, 2nd International Conference on Knowledge-Based Intelligent Engineering Systems,* volume 1, pages 290-297, Adelaide, Australia, April 1998.

[16] A. Ultsch. "Data mining and knowledge discovery with emergent Self-Organizing Feature Maps for multivariate time series", Kohonen maps, Elsevier, pp. 33-45

[17] D.L. Davies, D.W. Bouldin, "A cluster separation measure" *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI*-1(2) (1979), 224--227.