COMPARISON OF CLASSIFIERS IN AUDIO AND ACCELERATION BASED CONTEXT CLASSIFICATION IN MOBILE PHONES

Okko Räsänen[1], Jussi Leppänen[2], Unto K. Laine[1] and Jukka P. Saarinen[2]

[1]Department of Signal Processing and Acoustics,
School of Electrical Engineering, Aalto University
PO Box 13000, FI-00076 Aalto, FINLAND
phone: +358-09-470 22499, fax: +358-9-460 224,
email: okko.rasanen@tkk.fi

[2]Nokia Research Center,
P.O.Box 100, FIN-33721, Tampere, Finland
email: jussi.ar.leppanen@nokia.com

ABSTRACT

This work studies combination of audio and acceleration sensory streams for automatic classification of user context. Instead of performing sensory fusion at a feature level, we study the combination of classifier output distributions using a number of different classifiers. Performance of the algorithms is evaluated using a data set collected with casually worn mobile phones from a variety of real world environments and user activities. Results from the experiments show that combination of audio and acceleration data enhances classification accuracy of physical activities with all classifiers, whereas environment classification does not benefit notably from acceleration features.

1. INTRODUCTION

Automatic recognition of user contexts has received a large amount of interest in the field of ubiquitous computing. For mobile devices such as mobile phones and MP3-players, one central goal is to make the devices aware of their surroundings so that they are able to change their behaviour according to context specific usage patterns. The assumption is that if the systems can be made to behave intelligently, human-computer interaction becomes more natural and thereby technical solutions easier and more efficient to use.

Before intelligent adaptation can take place, the system needs to be able to separate different contexts from each other. In order to do this in real-time, the device has to collect sensory data from one or more of its various sensors and either discover and store patterns from the data for further usage and/or classify the incoming data using previously learned models. In this work, we adopt the paradigm where the mobile device is trained with a number of a priori context classes of interest using large amount of training data from real world recordings. More specifically, we are interested in automatic recognition of the (auditory) environment of the user, and the physical activity of the user.

Although auditory context recognition (e.g., [1-2]) and physical activity recognition (e.g., [3-4]) have been widely studied in context recognition, combination of audio and acceleration streams in mobile context recognition has received much less attention. To the authors' knowledge, the only approaches utilizing both audio and acceleration sensors for physical activity classification are reported in [5] and [6]. In the work of Ward et al. [5], audio and acceleration are combined for enhanced performance in recognition of assembly tasks using body external microphone. In [6], Doukas and Maglogiannis apply audio and acceleration features for fall detection of patients and elderly people. They concatenate audio features obtained from externally placed microphone and acceleration data measured from feet into a single feature vector that is used as input to an SVM classifier. They report lower false alarm rate when audio features are included in the analysis.

Since nearly all modern smartphones come with built-in audio and acceleration sensors, we wanted to investigate whether the combination of these two sensory streams could also yield enhanced performance in context recognition of a mobile phone user. The use situation with mobile phones is somewhat different from body-worn data collection systems, since the placement of the device is not fixed for all contexts. For example, the device (including the microphone) is often located inside a bag or pocket, making the quality of the sound recordings very different from body-external microphones.

Unlike low-level sensory fusion (e.g., concatenation of feature vectors), which is susceptible to unreliable input in one or more sensory streams, we investigate the feasibility of combining information from the data streams after the classification stage of the classifier. Since outputs of most classifiers work in a probabilistic domain, the outputs should inherently carry information not only regarding the context, but also regarding the reliability of each sensory stream at each time instant. Therefore the combination of class probabilities after stream-specific classification stages should be both complementary in terms of available sensory information, but also robust against non-informative inputs (such as noisy input or input that is not explained by the training data; see also [7]). We evaluate the feasibility of this hypothesis using four different classifiers on a large-scale data set collected from various real-world mobile phone usage situations.

2. MATERIAL

The data set used in this paper consists of audio and accelerometer data recorded on a mobile phone. We wanted the data to be recorded in as unobtrusive manner as possible and such that the recording procedure would not alter the regular behaviour of the person doing the recording. For this reason, the data was recorded using a mobile phone application that was running on the user's own phone. The application would prompt the user after regular intervals to annotate the situation the user was in. The annotation consisted of selecting an environment ('office', 'car' etc.), activity ('sitting', 'running') and phone location ('pocket', 'hand') from provided default lists. The users could also add their own environments, activities or phone locations. After the annotation was set, the application would record a one minute clip of 16 bit audio at a 16 kHz sampling rate and 3-axis accelerometer data at a rate of around 34 Hz. A five second pause was held before initiating the recording to allow the user to place the phone where it was before being prompted for the annotation (i.e. putting the phone back to the jacket pocket).

The data set selected for the experiments presented in this paper consists of data from three users and was recorded on several different Symbian S60 mobile phones. The annotated activities included travelling by car/bus, cycling, being still, walking, and running. The environment classes included office, street, car, train, meeting, restaurant, outdoors, home, and shop. The above do not cover all of the different environments and activities that were recorded. The ones used here were selected based on the number of recorded clips. Moreover, the classes are often a combination of user annotations. For example, the restaurant class corresponds to clips annotated as restaurant, pub, and café and the car class corresponds to clips annotated as car, taxi and bus.

The total duration of the data was approximately 22 hours. For the experiments, the signals were divided into three second segments, yielding a total of 26732 signals. The data set was divided into ten separate data sets of approximately equal size for N-fold cross validation purposes. The ten separate sets contain data recorded at different times, but two or more sets may have data recorded in the same physical place. For example, a user's office or car might appear in two or more sets. During experiments, the overall performance was measured with the N-fold procedure, always training the classifiers with nine data sets and testing with the remaining one. Then the data sets were rotated so that each of the data sets was once used for evaluation.

3. METHODS

Four classifiers were chosen for the study based on their computational complexity, since the overall goal is to perform on-line context classification in mobile devices with limited computational resources. The studied classifiers can be divided into two main categories: those that do not take temporal dependencies of subsequent feature values into account and those that do so. The first category was represented by Minimum-distance classifier (MDC) and k-Nearest Neighbours (kNN) algorithm. Classifiers belonging to the latter category included discrete Hidden-Markov Models (dHMM; [8]) and Concept Matrices (CM; [9]).

The data from the two different sensory streams, audio and acceleration, is combined at the decision level. Since the same type of classifier is always used for the two streams, we take a naïve but computationally straightforward approach and perform a linear combination of streams-specific class probability distributions from both classifiers in order to make the final classification decision (except for kNN where we combine neighbour-pools from the two streams before taking a majority vote). Since the aim of the current study is not to implement sensory fusion directly to a practical application, but to compare different classifiers, we simply search for the optimal weighting factor between the two data streams for each classifier.

In addition to studying the effects of multi-stream combination, the difference in performance between the two classifier categories was of interest. Since at least audio signals often contain temporal dependencies at the scale of hundreds of milliseconds, the assumption was that temporal models (CM and dHMM) would achieve better performance in classification of complex spectrotemporal audio patterns.

3.1 Data pre-processing

The audio features were created using a mel-frequency cepstral coefficient (MFCC) front-end that outputs 13 cepstral coefficients and their 1st and 2nd order derivatives. The MFCCs are calculated from 16 kHz audio data using a window of 30 ms and a frame shift of 40 ms.

As for the acceleration data, each sample was characterized by acceleration amplitude in 3 dimensions (x, y, and z). During preliminary tests, this representation was compared against a representation where the three first components of feature vector $\mathbf{a_f}$ defined the direction of the acceleration and a fourth component defined the overall magnitude A of the acceleration ($\mathbf{a_f} = [a_x, a_y , a_z, A]$). The direction was obtained by normalizing the original acceleration vector as a unit vector, and the magnitude was computed as the RMS of the original vector. In addition, the magnitude value of acceleration was divided by a manually chosen constant of 50 in order to scale it down to the approximately same value range as the direction information. Since the tests indicated that the normalized representation $\mathbf{a_f}$ led to slightly higher performance than the use of original acceleration values, it was therefore used in the actual experiments. In order to match the sampling rate of audio and acceleration features for classifier output combination, the acceleration data was downsampled to the windowing rate of audio by interpolation and decimation according to the signal lengths.

As a pre-processing step for CM and dHMM algorithms, the MFCC and acceleration features were transformed into discrete sequences from finite alphabets **A1** and **A2** using vector quantization (VQ) codebooks created with the standard *k-means* algorithm. Codebooks of size $N_{\mathrm{A1}} = 128$ for audio and $N_{\mathrm{A2}} = 32$ for acceleration were used. 10000 randomly chosen feature vectors from the training set were used to create each codebook. After pre-processing, each signal consisted of an approximately 65 elements (~3 seconds) long VQ sequence representing audio and a VQ sequence of equal length representing acceleration measurements.

3.2 MDC

MDC represents possibly the simplest (and fastest) classification algorithm available, making it an interesting baseline for environments with limited computational capacity. In MDC, each class c is represented by the vector $\mathbf{v}_c$, which corresponds to the mean of feature values $\mathbf{f}_c$ of the training data belonging to given class:

$$\mathbf{v}_c = \sum_{i=1}^{N} \mathbf{f}_c / N \tag{1}$$

where N is the total number of frames in the training data. The mean vector can be also referred to as the prototype model of the class. During classification, distance from each frame in the test signal is computed to all class prototypes and the prototype with the smallest average distance to the test signal is chosen as the class hypothesis.

In this study, separate prototypes were computed for audio and acceleration streams. The overall distance d between model c and signal S was defined as

$$d_{c,S} = \sum_{j=1}^{J} \alpha(\|\mathbf{v}_c - \mathbf{f}_j\|) + (1-\alpha)(\|\mathbf{w}_c - \mathbf{g}_j\|) \tag{2}$$

where $\mathbf{v}$ and $\mathbf{w}$ are the prototype vectors for audio and acceleration (respectively), $\mathbf{f}_j$ and $\mathbf{g}_j$ are the corresponding feature values in frame j, and J is the number of frames in the test signal. Weighting factor α is a free parameter that defines the weighting between data streams, $\alpha = 0$ corresponding to pure acceleration and $\alpha = 1$ corresponding to pure audio.

3.3 kNN

In contrast to the MDC classifier above, the kNN classifier represents an exemplar based approach to pattern recognition. Instead of computing a prototype vector for each class of interest, kNN stores all feature vectors that occur in the training data as exemplars of the given class. When a new input vector $\mathbf{f}$ arrives to the system, its distance to all exemplars in the training set is computed and k nearest exemplars are chosen. The class hypothesis of the input is defined by taking the majority vote of the classes of the k exemplars. In the given classification task, each test signal consists of approximately 65 feature vectors (3 seconds of data), and the k nearest candidates were chosen separately for each feature vector. Then the majority class was computed over all 65*k frames in the signal.

In order to combine the two data streams in the kNN, the 65*k best candidates were taken from both streams independently (k neighbours for each audio and k neighbours for each acceleration feature vector), and a majority vote was taken for a combined pool of neighbours of the both streams. Since the reasonable values of k were relatively low for our experiments, no continuous weighting between data streams was used, but kNN was only evaluated in three conditions: pure acceleration, pure audio, and equal combination of these two.

As a drawback, in its most basic form, the kNN classifier is computationally intractable for large training data sets (see also [10]). For example, one training set (set #1) in our context data set already contains 1 579 890 MFCC vectors with 39-dimensions, and each test signals contains 65 feature vectors of the same dimensionality. One can easily see that even the storage of all training examples is not practical for any reasonable implementation, not to mention that the distances from all feature vectors in each test signal have to be computed to all of the examples, leading to more than one hundred million distance computations for each three second signal. This implies that the number of exemplars has to be cut down dramatically. In this work, the issue was resolved by randomly selecting N feature vectors from the training set for each activity and environment class.

During preliminary experiments, the behaviour of the kNN was studied comprehensively with different values of k and different number of exemplars per class. Although the performance started to saturate already at 200 tokens per each class, 1000 exemplars per each class were selected for the final experiments since it provided marginally better performance (+1-3% accuracy for classification depending on the task) and was still computationally feasible on a powerful desktop machine. The number of nearest neighbours was set to $k = 5$ for the tests.

3.4 CM

The CM-algorithm is mainly designed for weakly-supervised pattern discovery from sequential data [9], but can be also utilized in purely supervised classification. In supervised mode, the training of CM begins by collecting frequencies $f_c(a_x, a_y \mid k)$ of all lagged bigrams consisting elements a_x and a_y in the input sequences $X = [a_1, a_2, a_3, \ldots, a_n]$ at lags $\boldsymbol{k} = \{k_1, k_2, \ldots, k_K\}$ for all $a_x, a_y \in [1, \ldots, N_\mathrm{A}]$ and all $\mathrm{X} \in \mathbf{X}_c$ corresponding to class c. Lagged bigrams $(a_x, a_y \mid k)$ mean that there are always k-1 non-determined elements between a_y and a_x. Then the transition frequencies are transformed into stochastic matrices $\mathbf{P}^\mathrm{S}$ by having:

$$\mathbf{P}_c^\mathrm{S}(a_y \mid a_x, k) = f_c(a_x, a_y \mid k) / \sum_{y=1}^{N_\mathrm{A}} f_c(a_x, a_y \mid k) \tag{3}$$

The Eq. (3) creates the probability distribution for the next state at distance k in the model c when a_x is given. Next, the probability that the given transition from a_x to a_y occurs in the case of c instead of any other models is taken into account by having:

$$\mathbf{P}_c(a_y \mid a_x, k) = \mathbf{P}_c^\mathrm{S}(a_y \mid a_x, k) / \sum_{m=1}^{N_c} \mathbf{P}_m^\mathrm{S}(a_y \mid a_x, k) \tag{4}$$

During recognition, activity $A(c,t)$ of model c at time t is achieved by simply summing the normalized transition probabilities P of the occurring transitions in X across all lags $\boldsymbol{k}$ ($|\boldsymbol{k}| = K$).

$$A(c,t) = \sum_{k=1}^{K} \mathbf{P}_c(X[t] \mid X[t-k], k) \tag{5}$$

This provides a temporally local estimate for the probability of the model. For test data with only one possible category per signal, the activation is computed for each moment of time t in the test sample and then the sum of activation over the entire test signal is computed for all models $\mathbf{c}$. Model with the highest activation is chosen as the hypothesis.

In the multi-stream CM, separate input streams are modelled in parallel and independently of each other. Models $\mathbf{P}_1$ are created for audio input similarly to single-stream condition using equations (3) and (4). Models $\mathbf{P}_2$ are created for acceleration in the similar way. The only difference to the single-stream condition is recognition phase, where the activation A(c) is computed as a linear combination of the probabilities in the two separate streams:

$$A(c) = \sum_{t=K+1}^{J} \left(\alpha \sum_{k=1}^{K} \mathbf{P}_{1,c}(X_1[t] \mid X[t-k], k) + (1-\alpha) \sum_{k=1}^{K} \mathbf{P}_{2,c}(X_2[t] \mid X[t-k], k) \right) \quad (6)$$

The weighting factor α determines the relative weight of each stream. In theory, the optimal weighting factor of CM is always $\alpha = 0.5$ if $N_{T,c} >> N_A^2$ is satisfied for all c, where $N_{T,c}$ is the total number of transitions in the training material for c and N_A is the size of the input alphabet. This property is due to the normalizations performed in equations (3) and (4) that cause equally probable transitions in different models c to have equal value despite their absolute frequencies or number of class specific training samples. In practice, the sensory VQ data is always more or less sparse, and therefore small biases in the optimal weighting factor may occur. Lags $\boldsymbol{k} = \{1,\ldots, 5\}$ were used in the experiments.

3.5 Discrete Hidden Markov Models

Discrete Hidden Markov Models were implemented using the HMM-Toolbox for MATLAB by Kevin Murphy (http://www.cs.ubc.ca/~murphyk/Software/HMM/). Due to lack of space, the details of the HMM algorithm are not discussed here, but the reader is suggested to see [8] for introduction to discrete and continuous density HMMs.

Similarly to the CM, the MFCC and acceleration data was vector quantized with codebooks of size N_{A1} = 128 for audio and N_{A2} = 32 for acceleration. The maximum-likelihood parameters of the dHMM were estimated using the standard Baum-Welch EM-algorithm. Before the actual experiments, the optimal topology of the classifier, the number of states per model, and the number of EM-algorithm iterations were determined. The best results in preliminary experiments were obtained with left-to-right dHMMs with 23 states for context classification and 25 states for activity classification, and therefore 25 state dHMM models where used in the actual experiments. Seven EM-algorithm iterations (with random initial parameters) were used since the additional iterations yielded only very slight increases in likelihood of the training data with an increasing risk of overtraining.

In the multi-stream experiments, a dHMM classifier was built separately for audio and acceleration for each class. During recognition, log-likelihoods of the both streams were combined with weights α and 1-α similarly to MDC and CM before making the final classification decision.

4. EXPERIMENTS

Fig.1 shows the mean results from user activity classification over all test signals in the ten test sets and Fig. 2 shows the corresponding results for user environment classification. Table 1 shows the combined stream performance separately for each activity type and table 2 shows the same for different environments (note that the mean results are now somewhat different from overall mean results in the figures since the number of signals per each class varies greatly).

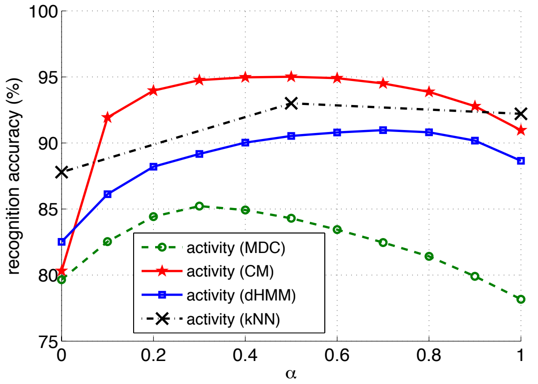

Figure 1 – Mean recognition accuracies over all test samples for user activities as a function of weighting between acceleration and audio data (α = 0 corresponds to pure acceleration whereas α = 1 stands for pure audio).

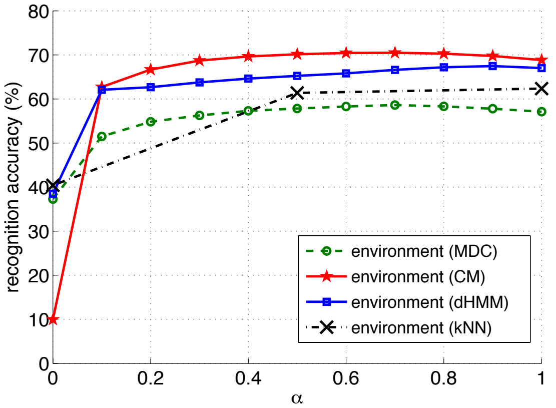

Figure 2 – Mean recognition accuracies over all test samples for user environments as a function of weighting between acceleration and audio data (α = 0 corresponds to pure acceleration whereas α = 1 stands for pure audio).

Table 1 – Activity recognition accuracies (%) for different classifiers in combined stream condition.

	CM	**dHMM**	**MDC**	**kNN**
car/bus	80.81	**84.13**	78.75	53.4
bicycling	**41.18**	38.82	34.71	28
idle still	**97.28**	91.54	87.75	96.15
running	**99.04**	93.89	85.85	98.07
walking	82.83	89.53	60.55	**89.63**
mean	**80.23**	79.58	69.52	73.05

Table 2 – Environment recognition accuracies (%) for different classifiers in combined stream condition.

	CM	**dHMM**	**MDC**	**kNN**
office	72.6	69.41	57.07	**77.25**
street	37.4	34.73	50.93	**71.77**
car	62.36	69.82	39.41	**72.52**
train	0	0	**41.22**	0.92
meeting	**78.78**	66.2	33.94	25.47
restaurant	**62.29**	55.81	53.29	39.89
outdoors	**49.94**	43.09	24	4.48
home	**78.61**	70.28	74.36	62.08
shop	26.9	**40.79**	28.51	20.96
mean	**52.1**	50.01	44.75	41.7

As can be seen from the figures, combination of audio and acceleration increases classification performance for user activities significantly for all classifiers. When the four methods are compared, it is evident that CM performs best when the two data streams are combined. However, dHMM comes close to the CM in performance when class specific accuracies are considered (tables 1 and 2). Interestingly, kNN outperforms others in activity classification when either acceleration or audio features are used in isolation. However, combination of the data streams gives only a slight improvement for the kNN. As discussed in section 3.4, optimal weighting factor for CM is $\alpha = 0.5$, i.e., no weighting of data streams is required for optimal performance. Optimal point of operation has to be optimized separately for other classifiers, or more intelligent stream combination methods are needed for actual real-time applications than simple decision level combination of posterior class probabilities.

As for the user environments, no large gain is observed in classification performance when acceleration is used in addition to audio. This is somewhat expected result, since the physical state of the user is not strongly coupled to the environment in our data. It is also evident that the user activity classification is in general more accurate than classification of the environment. This is partly explained by the fact that there are fewer activities than there are environments. Closer inspection also reveals that the contexts in which the phone may be located on a table (e.g., meeting, office and home) are detected more accurately than those in which the phone is often located inside pockets (e.g., train, outdoors, shop), causing audio signal quality to degrade notably.

Regarding modelling of temporal dependencies, CM and dHMM perform well on audio signals not completely characterized by an order-free collection of local spectral properties. By looking at table 2, one can observe that kNN performs very well on office, street and car environments that have relatively stationary acoustic content. On the other hand, CM and dHMM are significantly better in recognition of meeting, restaurant and outdoors environments where the auditory environment is not completely static but contains series of separable acoustic events such as operating of cash registers, people talking and passing by, footsteps, birds singing etc.

As for the computational complexity, the MDC and CM are computationally very fast (see [9]) and can be easily applied as a real-time algorithm in a mobile device, including real-time training of new signals. Classification with dHMM is also feasible, although it is significantly slower than the previous two due to more complex decoding ($O(N^2S)$ for the dHMM and $O(S)$ for the CM, where S is the sequence length and N the alphabet size). For kNN, the computational requirements depend on the number of exemplars that are used for each class (i.e. it has the complexity of MDC multiplied by the number of tokens per class). Tens or few hundreds of exemplars per class is feasible for modern phones, but then the question is how to properly select most efficient exemplars for each class. As for the memory, storing the parameters used in the experiments requires 1548B, 1548kB, 348kB, and 168kB for MDC, kNN (1000 exemplars), CM and dHMM, respectively. However, the CM models are very sparse and can be usually compressed to at least one fifth of the original size by storing transition probabilities into tables instead of reserving memory for full stochastic matrices.

5. CONCLUSIONS

Performance of four different classifiers was studied in the task of automatic recognition of mobile device user environments and physical activities when both audio and acceleration data were used as input features. The results show that the use of audio in addition to acceleration enhances classification performance for all classifiers in activity recognition. For environment recognition, no such gain was observed. As for the individual algorithms, CM algorithm outperforms kNN, dHMM and MDC classifiers in combined stream accuracy. It was also observed that the algorithms with capability to take temporal structure into account, i.e., CM and dHMM, perform well on signal types that have complex spectrotemporal dependencies that exceed feature window length. On other hand, kNN shows good performance on signals with relatively stationary properties.

REFERENCES

[1] L. Ma, B. Milner, and D. Smith, “Acoustic Environment Classification,” *ACM Transactions on Speech and Language Processing*, vol. 3, no. 2, pp. 1-22, 2006.

[2] T. Heittola, A. Mesaros, A. Eronen, and T. Virtanen, “Audio Context Recognition Using Audio Event Histograms,” in *Proc. EUSIPCO 2010*, Aalborg, Denmark, August 23-27, 2010, pp. 1272-1276.

[3] J. Parkka, M. Ermes, P. Korpipää, J. Mäntyjärvi, J. Peltola, and I. Korhonen, “Activity Classification Using Realistic Data From Wearable Sensors,” *IEEE Trans. Information Tech. Biomedicine*, vol. 10, No. 1, pp. 119-128, 2006.

[4] S. Dornbush, J. English, T. Oates, Z. Segall, and A. Joshi, “XPod: A Human Activity Aware Learning Mobile Music Player,“ in *Proc. Workshop on Ambient Intelligence*, IJCAI-2007, January 6-7, Hyderabad, India, 2007.

[5] J. Ward, P. Lukowicz, G. Tröster, and T. Starner, “Activity Recognition of Assembly Tasks Using Body-Worn Microphones and Accelerometers,” *IEEE Trans. Pattern Analysis and Machine Intel.*, vol. 28, pp. 1553-1567, 2006.

[6] C. Doukas and I. Maglogiannis, “Advanced Patient or Elder Fall Detection Based on Movement and Sound Data,” in *Proc. Pervasive Computing Technologies for Healthcare 2008,* Tampere, Finland, Jan. 30-Feb. 1, 2008, pp. 103-107.

[7] H. Wu: Sensor Fusion for Context-Aware Computing Using Dempster-Shafer Theory. Doctoral Thesis, Carnegie Mellon University, Philadelphia, USA, 2003.

[8] L. Rabiner, “A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition,” *Proceedings of the IEEE*, vol. 77, No. 2, pp. 257-286, 1989.

[9] O. Räsänen, and U.K. Laine, “A method for noise-robust context-aware pattern discovery and recognition from categorical sequences,” *Pattern Recognition*, in press.

[10] V. Könönen, J. Mäntyjärvi, H. Similä, J. Pärkkä, and M. Ermes, “Automatic feature selection for context recognition in mobile devices,” *Pervasive and Mobile Computing*, vol. 6, No. 2, pp. 181-197, 2009.