Odyssey 2012
The Speaker and Language Recognition Workshop
25-28 June 2012, Singapore

Audio Context Recognition in Variable Mobile Environments from Short Segments Using Speaker and Language Recognizers

Tomi Kinnunen[*], *Rahim Saeidi*[**], *Jussi Leppänen*[†], *Jukka P. Saarinen*[†]

[*]School of Computing, University of Eastern Finland (UEF), Joensuu, Finland
[**]Centre for Language and Speech Technology, Radboud University Nijmegen, the Netherlands
[†]Nokia Research Center (NRC), Tampere, Finland

Abstract

The problem of context recognition from mobile audio data is considered. We consider ten different audio contexts (such as *car*, *bus*, *office* and *outdoors*) prevalent in daily life situations. We choose mel-frequency cepstral coefficient (MFCC) parametrization and present an extensive comparison of six different classifiers: k-nearest neighbor (kNN), vector quantization (VQ), Gaussian mixture model trained with both maximum likelihood (GMM-ML) and maximum mutual information (GMM-MMI) criteria, GMM supervector support vector machine (GMM-SVM) and, finally, SVM with generalized linear discriminant sequence (GLDS-SVM). After all parameter optimizations, GMM-MMI and and VQ classifiers perform the best with 52.01 %, and 50.34 % context identification rates, respectively, using 3-second data records. Our analysis reveals further that none of the six classifiers is superior to each other when class-, user- or phone-specific accuracies are considered.

1. Introduction

There is no doubt that mobile phones have changed the world we live in. Modern smartphones are no longer just telephones but entire mobile computers with WiFi access and multiple sensors. Many of them are equipped with a high-resolution digital camera, global positioning system (GPS), infrared and accelerometer sensors. The detailed data streams provided by these specialized sensors can be used for inferring the user's current activity (*bicycling, walking*), physical location (*Länsikatu 15, Joensuu, Finland*) or perhaps even the user's social situation (*work meeting, attending a lecture, out in a pub with friends*). Being able to infer the user's activity patterns or physical location – commonly referred to as *context* – would be certainly useful for improving the relevance or quality of services from the customer's viewpoint. In this study, we focus on the core technology component, inference of the user's context using pattern recognition techniques.

While accelerometers, GPS and digital imaging have extensively been used for inferences of user's location and activity patterns, the most commonly available sensor found in *any* mobile phone – the microphone – has received less attention. There are, however, at least two good reasons to study auditory cues. Firstly, unlike WiFi or GPS signals, audio-based context recognition is not restricted to an existing network infrastructure (and its weaknesses, such as unavailable or unreliable GPS coordinates inside a building). Secondly, audio stream can be captured from any direction or even from inside a backpack or handbag without user interaction. There is also some indication that audio-based cues might be more accurate in recognizing both user's action and the acoustic environment in comparison with accelerometer sensors [1]. The focus of this study is to recognize the mobile user's *audio context* based solely on auditory cues.

The problem of audio context recognition has been studied e.g. in [8, 9, 10, 11]. Similar to speech and speaker recognition, the methods rely on short-term feature extraction followed by pattern classification. Regarding the first component, feature extraction, different parameterizations of the short-term spectrum, such as mel-spectrum, mel-frequency cepstral coefficients (MFCCs) and linear prediction (LP) coefficients have been popular. Even though a few studies [8, 12] have also attempted to use alternative time- and frequency-domain features, such as zero crossing rate, spectral flux or spectral centroids, or even sparse time-frequency patterns [11], there is experimental evidence that such features can merely complement the MFCCs or MFCC-like spectral shape descriptors; for detailed comparisons of features, see e.g. [8]. MFCC features are also used in the ETSI standard front-end for distributed speech recognition (DSR) and therefore, integrating context or environment detection for improved ASR model adaptation would not increase overhead at front-end. For these reasons, we utilize the standard MFCCs with their first and second order time derivatives, the delta and double delta coefficients, and focus on the pattern classification part.

The existing pattern modeling techniques for audio context recognition can be roughly divided into three major types, (1) *bag-of-frames* models, (2) *temporal* models and (3) *event detection* based models. The bag-of-frame approaches, similar to text-independent speaker recognition machinery, treat feature vectors as independent observations. They characterize the environment- or context-specific characteristics of audio excerpts using distribution models such as Gaussian mixture models (GMMs) or discriminatively trained models such as support vector machines (SVMs) [12]. The second class of methods, temporal models, rely on continuous [8] or discrete [1] observation hidden Markov model (HMM) variants or other sequence modeling techniques [1] to learn context-specific temporal profiles of acoustic features. Finally, event detection based methods [13, 14] use pre-trained event detectors (e.g. *laughter, cheering*) to characterize typical events in an audio stream. Similar to high-level feature modeling in speaker verification, the occurrences of events, modeled using histograms or other discrete models, constitute the context-specific back-end models.

In this study, we focus on the bag-of-frames paradigm for two reasons. The first is to keep the systems generally sim-

The work of T. Kinnunen was supported by Academy of Finland (proj. no. 132129) and the work of R. Saeidi was funded by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 238803.

Table 1: Six methods are compared for audio context recognition. GMM: Gaussian mixture model, ML: maximum likelihood, MMI: maximum mutual information, SVM: support vector machine, GLDS: generalized linear discriminant sequence.

Id	Classifier	Training	Main control parameters	Values considered
(1)	k-nearest neighbor (kNN) [2]	Generative	Codebook size (M)	$M \in \{4, 8, 16, \ldots, 8192\}$
			No. of neigbors (k)	$k \in \{1, 3, 5, 9\}$
(2)	Vector quantization (VQ) [3]	Generative	Codebook size (M)	$M \in \{4, 8, 16, \ldots, 8192\}$
(3)	GMM with ML training [4]	Generative	No. of Gaussians (K)	$K \in \{256, 512, 1024\}$
(4)	GMM with MMI training [5]	Discriminative	No. of Gaussians (K)	$K \in \{256, 512, 1024\}$
(5)	GMM-SVM [6]	Discriminative	No. of Gaussians (K)	$K \in \{8, 16, \ldots, 256\}$
			Relevance factor (r)	$r \in \{1, 16\}$
(6)	GLDS-SVM [7]	Discriminative	Max. monomial order (Q)	$Q \in \{1, 2, 3\}$

ple and computationally efficient; no Viterbi decoding or additional event detector training is required. The second reason – our primary motivation for the present study – is that we would like to utilize as much as possible the existing infrastructure for two well-studied speech classification problems, speaker and language recognition. To this end, we consider six classifiers shown in Table 1. The two classical distance-based methods (kNN and VQ) have very low computational complexity, whereas the four other classifiers are widely used in modern speech classification tasks.

Our main contributions can be summarized as follows. Firstly, we utilize a challenging mobile audio context database consisting of 100k+ 3-second audio segments collected with different Nokia phones in ten different audio context categories. Secondly, we provide detailed evaluation and analysis of context classification using the six classifiers in Table 1. Finally, detailed break-down of the recognition results is presented in terms of our ten audio contexts, our six users (four males, two females), our four mobile devices and even the two different cities (Tampere and Helsinki, Finland) the data was collected in.

2. Classifiers

All the six classifiers considered in this study (Table 1) are trained using labeled training set $\mathcal{T} = \{(\boldsymbol{x}_n, y_n)|n = 1, 2, \ldots, N_{\text{train}}\}$, where N_{train} indicates the number of d-dimensional acoustic vectors $\boldsymbol{x}_n = (x_{1,n}, x_{2,n}, \ldots, x_{d,n})^{\text{T}} \in \mathbb{R}^d$, and $y_n \in \{1, 2, \ldots, C\}$ denotes the label of one of our $C = 10$ target contexts (see Table 2). Further, let $\mathcal{X}_c \triangleq \{\boldsymbol{x}_n|y_n = c\}$ denote the pool of training vectors of class (context) c. We consider three generative and three discriminative methods. Generative models for each class are trained independently from each other, whereas discriminative training utilizes training examplars from the competing classes. In the system operation phase, each classifier attempts to predict the true class of an unseen audio segment, now presented as a sequence of vectors $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$. In our case, we classify 3-second segments using standard MFCC + Δ + Δ^2 front-end with $d = 39$ and $T = 65$. More details of feature extraction will be provided in Subsection 3.1.

2.1. k-Nearest Neighbor (kNN) and Vector Quantization (VQ)

A few simple dissimilarity- or template-based context predictors can be built up by computing pairwise distances between the test segment $\mathcal{X}$ and the training examplars of each class. Here we consider the well-known k-nearest neighbor (kNN) [2] and vector quantization (VQ) [3] classifiers. They both require search of nearest training set neighbors (in Euclidean metric sense) for each test vector which becomes impractical for sizeable training sets as are used here. In [1] this computational problem was tackled by using 1000 randomly chosen examplars to represent each class. In this study, we attack the problem by using *codebooks* to represent the training sets. That is, each training set $\mathcal{X}_c$ is replaced by its quantized version $\hat{\mathcal{X}}_c = \{\hat{\boldsymbol{x}}_1^{(c)}, \hat{\boldsymbol{x}}_2^{(c)}, \ldots, \hat{\boldsymbol{x}}_M^{(c)}\}$ consisting of M representative centroid vectors $\{\hat{\boldsymbol{x}}_m^{(c)}\}_{m=1}^M$. These centroid vectors are independently optimized for each class using K-means [15] with deterministic splitting initialization and 20 K-means iterations.

The match score for an unseen audio excerpt is computed using three alternative methods, kNN scoring, rank-based scoring and MSE scoring. To define these precisely, let $\pi(\boldsymbol{x}) = \{\pi(1), \pi(2), \ldots, \pi(k)\}$ index the k disjoint nearest neighbors of the query vector $\boldsymbol{x}$ across all the class-dependent codebooks so that $\|\boldsymbol{x} - \hat{\boldsymbol{x}}_{\pi(1)}\|^2 \leq \|\boldsymbol{x} - \hat{\boldsymbol{x}}_{\pi(2)}\|^2 \leq \cdots \leq \|\boldsymbol{x} - \hat{\boldsymbol{x}}_{\pi(k)}\|^2 \leq \|\boldsymbol{x} - \hat{\boldsymbol{x}}_i\|^2$ for all $\hat{\boldsymbol{x}}_i \in \hat{\mathcal{X}}_1 \cup \ \ldots \ \cup \hat{\mathcal{X}}_C$. In kNN scoring, we assign $\boldsymbol{x}$ to the class that collects the majority of class assignments in the k-neighborhood. To classify the entire sequence $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$, we simply count the majority of frame-level assignments. In the rank-based method, instead of assigning a hard label to each vector, the rank of each class is recorded so that the class corresponding to $\hat{\boldsymbol{x}}_{\pi(j)}$ gets score j; the full sequence is then assigned to the class whose ranks summed up over the full sequence attains minimum. Finally, the mean-square error (MSE) scoring or vector quantization (VQ) method assigns $\mathcal{X}$ to the class that minimizes $\text{MSE}(\mathcal{X}, \hat{\mathcal{X}}_c) = \frac{1}{T}\sum_{t=1}^T \min_m \|\boldsymbol{x}_t - \hat{\boldsymbol{x}}_m^{(c)}\|^2$.

2.2. GMM Systems: GMM-ML and GMM-MMI

The GMM density function (1) consists of finite mixture distribution of K multivariate Gaussians whose parameters are the prior probabilities P_k, mean vectors $\boldsymbol{\mu}_k$ and covariance matrices $\boldsymbol{\Sigma}_k$,

$$p(\boldsymbol{x}|\lambda) = \sum_{k=1}^{K} P_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \tag{1}$$

where the prior probabilities follow probabilistic constraints $P_k \geq 0$ and $\sum_{k=1}^K P_k = 1$. Here, $\lambda = \{P_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k|k = 1, \ldots, K\}$ is a shorthand for all the model parameters. In maximum likelihood (ML) training, we train class-dependent GMMs $\{\lambda_c|c = 1, 2, \ldots, C\}$ independently for all $C = 10$ pattern classes (contexts). This is carried out by maximizing the following objective function,

$$\mathcal{F}^{\text{ML}} = \sum_{c=1}^{C} \sum_{\boldsymbol{x} \in \mathcal{X}_c} \log p(\boldsymbol{x}|\lambda_c), \tag{2}$$

using the expectation-maximization (EM) [4, 16] algorithm. In the classification phase, we find the most likely class (c^*) for a test sequence $\mathcal{X}$ as $c^* = \arg\max_c \sum_{\boldsymbol{x}_t \in \mathcal{X}} \log p(\boldsymbol{x}_t|\lambda_c)$. In our implementation, we use diagonal covariance matrices and 7 EM iterations.

The ML training of GMMs does not take misclassification effect in modeling phase into account as each class is trained by using training examplars from that class only. To enhance classification accuracy, GMM training can also be realized discriminatively. One of the most successful approaches is to train GMMs with *maximum mutual information* (MMI) criterion [17]. The MMI objective function to be maximized is,

$$\mathcal{F}^{\mathrm{MMI}} = \sum_{c=1}^{C} \sum_{\mathcal{X} \in \mathcal{X}_c} \log \frac{p(\mathcal{X}|\lambda_c)P(c)}{\sum_{c'=1}^{Y} p(\mathcal{X}|\lambda_{c'})P(c')}. \quad (3)$$

In practise this is maximized using *extended Baum-Welch* (EBW) algorithm [5]. To prevent MMI training from learning the class-dependent prior, the statistics in the training phase are weighted inversely proportional to the number of training segments. We used the STK toolkit[1] for constructing our MMI-based recognition system. GMMs trained with ML criterion are used as initial models for MMI training using 20 EBW refinement iterations. Context classification is done according to,

$$c^* = \arg\max_c \log \left\{ \frac{p(\mathcal{X}|\lambda_c)^{1/T}}{\sum_{c'=1}^{C} p(\mathcal{X}|\lambda_{c'})^{1/T}} \right\}, \quad (4)$$

where T is the number of feature vectors in $\mathcal{X}$.

2.3. GMM Supervector SVM (GMM-SVM)

In the GMM supervector method [6], any speech utterance $\mathcal{X}$ – in our case, arbitrary audio excerpt – is represented as a supervector formed by stacking all the Gaussian means of a GMM with K components, $\boldsymbol{m} = (\boldsymbol{\mu}_1^{\mathrm{T}}, \boldsymbol{\mu}_2^{\mathrm{T}}, \ldots, \boldsymbol{\mu}_K^{\mathrm{T}})^{\mathrm{T}}$. The mean vectors are obtained by maximum *a posteriori* (MAP) adaptation of a universal background model (UBM); the details can be found in [18] but for completeness, we briefly review the method here.

Assume that a UBM – a single GMM – has been trained using large quantities of off-line data using the EM algorithm. In speaker verification, where massive off-line development corpuses from NIST and LDC are available, the UBM is usually trained using disjoint data from the target class modeling data. Due to lack of similar training set containing all the possible sounds of our auditory world, we simply pool all the training set vectors $\mathcal{T}$ for UBM training.

Let us denote the mean vectors of the UBM as $\boldsymbol{\mu}_k^{\mathrm{UBM}}$ for $k = 1, \ldots, K$. Given a training or test segment $\mathcal{X}$, its k^{th} adapted mean vector $\boldsymbol{\mu}_k$ is computed as $\boldsymbol{\mu}_k = \alpha_k \hat{\boldsymbol{x}}_k + (1 - \alpha_k)\boldsymbol{\mu}_k^{\mathrm{UBM}}$, where $\hat{\boldsymbol{x}}$ is the mean of acoustic vectors assigned (softly) to the k^{th} Gaussian of the UBM and α_k is adaptation factor given by $\alpha_k = n_k/(n_k + r)$. Here, n_k is the soft count of vectors assigned to the k^{th} Gaussian and r is a fixed *relevance factor*. For $r = 0$, the prior model (UBM) is effectively ignored while for large r, contribution from UBM is larger.

Assuming that $\boldsymbol{\mu}_k^a$ and $\boldsymbol{\mu}_k^b$ represent the adapted mean supervectors of two audio excerpts, their similarity is measured by the GMM supervector kernel [6] defined as,

$$K(\lambda_a, \lambda_b) = \sum_{k=1}^{K} \left(\sqrt{P_k} \boldsymbol{\Sigma}_k^{-\frac{1}{2}} \boldsymbol{\mu}_k^a \right)^{\mathrm{T}} \left(\sqrt{P_k} \boldsymbol{\Sigma}_k^{-\frac{1}{2}} \boldsymbol{\mu}_k^b \right), \quad (5)$$

[1] `http://www.fit.vutbr.cz/research/groups/speech/stk.html`

where P_k and $\boldsymbol{\Sigma}_k$ are the prior probability and diagonal covariance matrix of the k^{th} Gaussian of the UBM, respectively. This means that the GMM mean supervectors are pre-normalized using $\sqrt{P_k}\boldsymbol{\Sigma}_k^{-\frac{1}{2}}$, where P_ks and $\boldsymbol{\Sigma}_k$s are the priors and covariances taken from the UBM. The normalized supervectors are then used in training and scoring linear-kernel SVMs for each of the pattern classes; for this, we use the `LibLinear` package [19] with the default options and multiclass training.

2.4. SVM with GLDS Kernel (GLDS-SVM)

In the generalized linear discriminant sequence (GLDS) method [7], any training or test segment $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$ is mapped into a high-dimensional average expanded vector $\mathcal{X} \mapsto \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{b}(\boldsymbol{x}_t)$, where the nonlinear mapping $\boldsymbol{b}(\boldsymbol{x})$ expands acoustic vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_d)^{\mathrm{T}}$ using all the monomials $x_{i_1} x_{i_2} \ldots x_{i_Q}$, where $i_1 \leq i_2 \leq \cdots \leq i_Q$. The maximum degree of the monomials, Q, is a user-supplied parameter; in practise it is usually $Q = 2$ or $Q = 3$ since the dimensionality of the expanded space, $(d + Q)!/(d! \times Q!)$, becomes impractically high for typical cepstral feature extraction schemes.

In the GLDS kernel implementation (for details, refer to [7]), the polynomially expanded supervectors are further normalized with an inverse of estimated correlation matrix $\boldsymbol{R}$ of the expanded supervectors. Including this to the inner-product form of the generalized linear kernel implies that all the polynomially expanded supervectors are normalized using matrix $\boldsymbol{R}_{\mathrm{sqr}} \triangleq \boldsymbol{R}^{-\frac{1}{2}}$. In [7], $\boldsymbol{R}$ is estimated using the nontarget speaker data (background data). In our implementation, $\boldsymbol{R}$ is chosen to be diagonal as in [7]. We estimate it using all the polynomial supervectors of the training set, independent of the class. We also experimented with class-dependent normalization (e.g. using *non-office* data to create normalization matrix for all the *office* supervectors). This yielded slightly, but systematically, lower recognition accuracy and with generally somewhat cumbersome implementation. The polynomially expanded and normalized supervectors are used for with default multi-class linear-kernel SVM training and scoring settings with `LibLinear` [19].

3. Evaluation Data and Setup

3.1. Data Collection and Feature Extraction

The data set used for the experiments presented in this paper was collected using mobile phones. Specialized software that allowed for as unobtrusive as possible recording of everyday environmental sounds was installed on the phones of six users. The software would prompt the users at regular intervals to annotate their current environment. The annotation consisted of selecting the user's environment ("office", "shop", etc.), activity ("walking", "bicycling", etc.) and phone location ("pocket", "hand", etc.) from drop-down lists. Users could also add their own environments, activitites and phone locations if they were missing from the lists. After the annotation, the software would record a one-minute clip of 16-bit audio at 16 kHz. The users were instructed to continue whatever they were doing before being prompted for the duration of the recording so that the annotation would match the recording. A five second pause was held before the start of the recording to allow for the user to place the phone where it was before being prompted for the annotation. The default recording interval was set to 20 minutes, but this could be altered by the users.

The number of users participating in the data collection

Table 2: Composition of the context classification data.

User	Gender	Age grp	City	Phone models	Number of collected samples per user and class. Car	Home	Lect.	Office	Outd.	Restaur.	Mall	Street	Train	Other	Total
1	Male	30–40	Tampere, FIN	5800, 6210 Nav., N95, N85	1151	3358	888	2615	811	694	456	720	551	1242	12486
2	Male	30–40	Tampere, FIN	6210 Navig., N95	2605	8312	3327	9483	3532	1942	1099	2684	1873	4790	39647
3	Female	20–30	Tampere, FIN	6210 Navig.	1106	17708	800	22917	949	3313	1581	5769	695	3147	57985
4	Female	30–40	Tampere, FIN	6210 Navig.	38	1045	95	1063	95	114	0	0	0	228	2678
5	Male	40–50	Helsinki, FIN	6210 Navig.	283	38	190	38	169	114	0	247	112	285	1476
6	Male	<20	Tampere, FIN	6210 Navig.	0	310	19	37	0	0	0	0	0	19	385
				Total	5183	30771	5319	36153	5556	6177	3136	9420	3231	9711	114657

was six. The data was recorded on several different Symbian S60 phones. A subset of the collected data, as shown in Table 2, was selected for the experiments in this paper. Namely, we chose data whose annotation matched with the following nine classes: car/bus, home, meeting/lecture, office, outdoors, restaurant/pub/café, shop/mall, street/road and train/metro/tram. In addition to the above, some data not matching to the above classes was selected to from an 'unknown' class; i.e. a class that represents all data that is not defined by the nine "known" classes.

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. The choice of the somewhat longer than usual frame shift is partly due to computational power restrictions on mobile phones – a longer frame shift means less feature vectors. For every second of audio, the front-end produces 25 feature vectors with a total of 39 components. For our 3-sec test segments, we get a total of 65 feature vectors. 10 feature vectors are lost from the beginning due to delta and double delta initializations.

Since the focus of this study is on comparing classifiers, rather than implementing a context recognition prototype running in an actual mobile device [20], all the classification experiments are carried out using Matlab and other tools running on desktop PCs.

3.2. Evaluation Setup and Performance Measure

Evaluation is carried out using *leave-one-user-out* cross-validation. That is, each of the six users in Table 2 is considered as a held-out test user at a time and the classifiers are trained using data from all the five other users. Total accuracy can then be assessed by accumulating the errors from all held-out test sets. To define this precisely, let $y_n(u), \hat{y}_n(u) \in \{1, 2, \ldots, 10\}$ denote, respectively, the ground truth and the predicted class labels (given by a classifier) of the n^{th} test sample for user $u \in \{1, 2, \ldots, 6\}$. Further, let $N(u)$ denote the total number of test samples of user u.

A natural measure of performance is the correct context identication rate or weighted accuracy (WA),

$$\text{WA} = \frac{1}{N} \sum_{u=1}^{U} \sum_{n=1}^{N(u)} \mathcal{I}\{y_n(u) \equiv \hat{y}_n(u)\}, \qquad (6)$$

where $U = 6$ and $N = \sum_{u=1}^{U} N(u) = 114657$ is the total number of test samples and $\mathcal{I}\{\cdot\}$ is the 0/1 indicator function. Here, *weighted* stresses that (6) is dominated by classes with high number of test samples. Using this measure alone for optimizing classifier parameters may favor developing good detectors for the majority classes. To account for the class imbalance problem, we also report the average of the per-class identification rates, or *unweighted* accuracy (UWA):

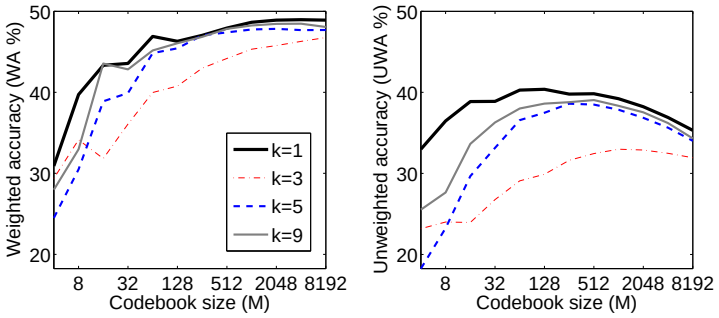

Figure 1: Effect of neighborhood size (k) to kNN accuracy.

$$\text{UWA} = \frac{1}{C} \sum_{c=1}^{C} \frac{1}{N(c)} \sum_{u=1}^{U} \mathcal{I}\{\hat{y}_n(u) \equiv y_n(u) \wedge y_n(u) \equiv c\}. \qquad (7)$$

Here $C = 10$ is the number of classes, $c \in \{1, 2, \ldots, 10\}$, and $N(c) = \sum_{u=1}^{U} \mathcal{I}\{y_n(u) \equiv c\}$ is the total number of test samples from class c. Similar unweighted and weighted accuracies have also been used for evaluating emotion recognition accuracy for imbalanced class sizes in [21]. UWA is our primary metric for parameter optimizations.

4. Results for the Individual Classifiers

4.1. Distance-Based Methods

The overall context identification accuracies for the simplest classifier, k-NN, are shown in Fig. 1. As expected, using larger codebook size (template) systematically improves accuracy. For codebook sizes larger than 512, UWA decreases due to overfitting. Regarding the number of neighbors, k, simple accumulation of majority votes over all the frames ($k = 1$) gives the highest accuracy. Fig. 2 furthermore contrasts this case against the two other scoring methods considered, MSE- and rank-based scoring. The standard VQ scoring (MSE) outperforms the voting type of k-NN and rank methods as might be expected. Regarding UWA, best codebook sizes are on the range $128 \leq M \leq 2048$; from these choices of M, WA attains maximum at $M = 2048$ and will be fixed for the rest of the paper.

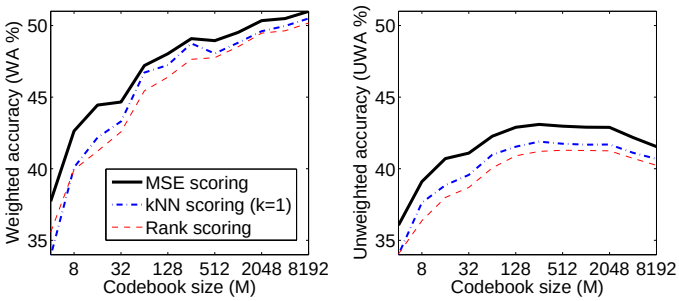

Figure 2: Overall performance of the distance-based classifiers.

Table 3: Accuracy of the GMM system with ML and MMI training. K: number of Gaussians.

K	Weighted accuracy (WA %) ML train.	MMI train.	Unw. accuracy (UWA %) ML train.	MMI train.
256	49.97	37.47	44.55	31.81
512	49.96	52.01	44.44	44.03
1024	50.05	51.24	43.88	42.92

4.2. GMM with ML and MMI training

The GMM-ML and GMM-MMI systems both require setting the number of Gaussians, which is varied as $K \in \{256, 512, 1024\}$. The number of MMI iterations is set to 20 as in [5]. The results for ML and MMI trained GMMs are presented in Table 3. The fact that the results are just slightly different for ML trained GMMs with different number of Gaussians might be an indication of limited within-class acoustic variability which is already captured using 256 Gaussians. The improvements in MMI training over ML training for the WA metric are expected because of the capability of discriminative training to incorporate the class boundaries into the training process. The choice of right model order is critical in building an accurate boundary. Because of different amounts of training data for model training in the held-out training sets, it turns out that, on average, MMI-training with 512 Gaussians yields the highest weighted accuracy whereas the ML-trained GMM with 256 Gaussians provide the highest unweighted accuracy. MMI training attempts to maximize the weighted, rather than unweighted, accuracy which can be a reason why the ML-trained GMMs outperform MMI-trained GMMs in terms of UWA. In the rest of the paper, we fix $K = 512$ for both ML- and MMI-trained systems.

4.3. GMM-SVM

Results for the GMM-SVM classifier are shown in Table 4. Highest recognition rate, in terms of WA, is obtained using $K = 128$ Gaussians with relevance factor $r = 16$. Regarding UWA, the maximum is obtained using $K = 16$ and $r = 16$. Regarding the choice of the number of Gaussians or relevance factor, no obvious trends can be seen. For the two highest model orders, $K = 128$ and $K = 256$, using larger relevance factor helps as expected. For the rest of the paper, we fix $K = 128$ and $r = 16$; this gives highest WA and second highest UWA.

4.4. GLDS-SVM

Results for the GLDS-SVM method are shown in Table 5 for three different maximum monomial orders ($Q = 1, 2, 3$). Note

Table 4: Accuracy of the GMM-SVM system. K: number of Gaussians, r: relevance factor in MAP adaptation.

K	Weighted accuracy (WA %) $r = 1$	$r = 16$	Unw. accuracy (UWA %) $r = 1$	$r = 16$
8	13.57	9.54	10.69	10.57
16	27.20	28.19	12.25	**18.76**
32	17.68	11.43	**16.13**	7.96
64	**28.89**	20.46	15.19	13.27
128	9.36	**34.37**	10.80	17.08
256	16.32	28.60	10.26	16.76

Table 5: Results for the GLDS-SVM method with three maximum monomial orders (Q).

	Weighted acc. (WA %)	Unw. acc. (UWA %)
$Q = 1$	34.18	24.40
$Q = 2$	49.57	35.72
$Q = 3$	46.87	32.89

that the case $Q = 1$ merely corresponds to SVM trained using utterance-averaged MFCC features studied for low-cost context recognition in [12]. The supervector dimensionalities for $Q = 2$ and $Q = 3$, in turn, are 820 and 11480, respectively.

As expected, the polynomial kernels containing second ($Q = 2$) and third order ($Q = 3$) terms outperform simple utterance-level MFCC averaging ($Q = 1$) because they capture higher-order statistics of the MFCC features. Highest accuracies (both WA and UWA) are obtained using $Q = 2$ which seems to represent a compromise for preserving context-related characteristics without overfitting. Importantly, this is also computationally faster in comparison to $Q = 3$.

5. Analysis of the Results

Up to this point, we have presented the results in terms of total accuracy (either WA or UWA) for ease of interpretation. We now analyze the achieved recognition rates in detail. Summing up the above findings, we fix the classifier parameters as follows. For the distance-based methods (kNN and VQ scoring) we use $M = 2048$ codevectors per class and for kNN, we further set the neighborhood size as $k = 1$. For the GMM-ML and GMM-MMI systems we use $K = 512$ Gaussians, and for the GMM-SVM system, we set it to $K = 128$ with relevance factor $r = 16$. For the GLDS-SVM system, we use polynomials up to second order ($Q = 2$).

5.1. Classifier- and Class-Dependent Accuracies

The results broken down by class, along with the two integrated WA and UWA measures and number of test cases in each class, are given in Table 6. It is obvious that our classifiers behave differently across the classes. The results for kNN and VQ are very similar since they both use the same codebooks and differ only in the scoring process. Comparing the two GMM systems, none of the training principles (ML vs MMI) is superior over each other; both win in half of the cases.

Comparing the two SVM methods, GLDS-SVM generally outperforms the GMM-SVM kernel. The behavior of the GMM-SVM is relatively low; it classifies *all* the test samples incorrectly in four cases. An interesting observation, however, is that even though GMM-SVM gives the lowest overall accuracy, it has the most accurate *office* detector with 91.36 % accu-

Table 6: Audio context recognition results (correct identification rates, %) broken down by class. The classifier with highest accuracy per class is bolded.

Class	N_{test}	kNN	VQ	GMM-ML	GMM-MMI	GMM-SVM	GLDS-SVM	Average
Car/bus	5183	59.71	59.63	63.14	**63.12**	20.06	45.24	51.82
Home	30771	48.41	48.55	53.19	54.31	1.06	**57.98**	43.92
Meeting/lecture	5319	47.20	54.08	**62.77**	57.49	0.00	27.39	41.49
Office	36153	70.93	70.88	62.82	68.30	**91.36**	70.07	72.39
Outdoors	9711	14.11	15.40	**16.41**	13.91	12.90	10.49	13.84
Restaurant/pub/café	5556	47.64	50.10	62.50	**65.76**	0.00	43.33	44.89
Shop/mall	6177	45.82	44.92	**48.53**	39.79	0.00	21.74	33.47
Street/road	3136	40.41	41.68	36.79	39.49	39.34	**41.88**	39.93
Train/metro/tram	9420	**36.83**	36.08	29.12	24.79	0.00	28.07	25.81
Other	3231	5.80	7.51	9.30	**13.33**	6.14	10.98	8.85
Total (WA %)	114657	49.60	50.34	49.96	52.01	34.37	49.57	
Total (UWA %)	114657	41.69	42.89	44.44	44.03	17.08	35.72	

Table 7: Recognition rates (% correct) for the GMM-MMI classifier ($K = 512$) by user and class. The missing values (–) mean that there is no test data for that class/user pair. The mean values indicated by * have been computed over the nonmissing values only. TMP = Tampere, HEL = Helsinki.

		User ID (gender, city)					
		1 (♂, TMP)	**2** (♂, TMP)	**3** (♀, TMP)	**4** (♀, TMP)	**5** (♂, HEL)	**6** (♂, TMP)
Class	N_{test}	12486	39647	57985	2678	1476	385
Car/bus	5183	63.77	62.11	78.48	84.21	7.07	–
Home	30771	72.66	72.17	42.35	52.34	86.84	62.90
Meeting/lecture	5319	45.95	56.12	88.13	62.11	7.89	21.05
Office	36153	1.95	84.11	72.54	0.00	18.42	94.59
Outdoors	9711	17.14	9.77	27.71	27.37	0.00	–
Restaurant/pub/café	5556	68.88	64.93	67.61	72.81	0.00	–
Shop/mall	6177	40.79	49.04	33.08	–	–	–
Street/road	3136	53.75	61.55	29.14	–	0.00	–
Train/metro/tram	9420	30.49	28.78	13.53	–	0.00	–
Other	3231	15.14	11.92	16.27	6.14	3.51	0.00
	Average	41.05	50.05	46.88	43.56*	13.74*	44.63*

racy. While this extreme behaviour calls for a detailed analysis, we hypothesize that our UBM required for GMM supervector generation is likely to be biased towards the majority class (office). We did not attempt to do any data or Gaussian balancing in the UBM training process.

A possible explanation for the low performance of GMM-SVM in comparison with GMM-ML and GMM-MMI is that the former requires creating a GMM for the test utterance prior to SVM scoring whereas the latter two just compute a likelihood score; the extremely short data duration (3 seconds or 65 MFCC vectors) probably causes unreliable estimation of the GMM parameters.

Comparing the classes themselves, *office* is clearly the easiest to classify correctly, with 72.39 % average over our classifier pool. There are two possible reasons for this. The first one is statistical: the office environment has generally the largest amount of training data available which the statistical models are able to take advantage of. The second one could be due to homogeneity of the environment itself – most of the office recordings were carried out on the same floor of an office building with relatively homogenous soundscape.

The most difficult class, in turn, is *other* with only 8.85 % average recognition rate. This comes as no surprise since this represents an out-of-set class. The individual recordings inside this class are highly variable and our classifiers are unable to capture general behavior of *all the rest of our auditory world* with such a small training set. Clearly, more intelligent way of modeling the out-of-set class would be needed.

5.2. Analysis by User and Class

Different users have generally different lifestyle, daily routines, places of interest and behavioral patterns, which might be reflected also in the types of audio contexts collected. For the analysis of the user factor, we present the results for the GMM-MMI classifier ($K = 512$ Gaussians) only. The results for each class and user are given in Table 7. In the case of missing values, the mean values are computed only over the non-missing values and should be interpreted with care. Perhaps the most interesting accuracy is that of the user no. 5 whose data is partly collected in another city (Helsinki). His average recognition accuracy (13.74 %) is clearly lower than that of the other five users. This might be partially caused by the differences in the acoustic contexts across the two cities. None of the *street*, *train*, *restaurant* or *outdoors* samples of user no. 5 were correctly recognized when trained using Tampere training data.

5.3. Effect of the Mobile Phone

Our last analysis for the VQ and GMM-MMI classifiers in Table 8 considers the possible channel effect arising from differ-

ent collection devices. We have selected the two users (1 and 2) who have test clips from than just one mobile phone. Regarding WA for user 1, the lowest accuracies for both VQ and GMM-MMI are obtained for N85. This seems to suggest for a channel effect, since *there is no training data for this phone* (see Table 2). Conversely, 6210 Navigator yields the highest accuracy for the GMM-MMI system; this is expected since it has the largest amount of training data. Regarding WA for user 2, similar effect can be observed for the GMM-MMI system; low accuracy on N95 which has training data only from user 1, and clearly higher accuracy for 6210 Navigator, the model with most training data. For the VQ classifier, the result is however opposite. One limitation of the present analysis is that, in addition to the device and user factors, other factors such as the class-dependent amount of training data or selection of training excerpts may also affect the results. Some careful data balancing or, ideally, parallel audio recordings with multiple devices, would be needed for detailed isolation of the "device-only" factor.

Table 8: Accuracy of the VQ ($M = 2048$) and MMI-GMM ($G = 512$) classifiers for different phones from users 1 and 2.

			Weighted acc. (WA %)		Unw. acc. (UWA %)	
User	Phone	N_{test}	VQ	GMM-MMI	VQ	GMM-MMI
1	6210 Nav.	6070	24.11	30.57	21.71	14.51
	5800	4865	36.55	18.07	39.71	12.53
	N95	1172	8.44	9.30	6.71	15.84
	N85	380	3.15	5.00	12.07	25.00
2	6210 Nav.	34497	18.09	49.89	8.43	32.90
	N95	5150	32.69	8.15	38.61	7.38

6. Discussion and Conclusions

The problem of audio context recognition from short and highly variably mobile data records was considered. This problem setup is challenging without a doubt; in our comparison of six classifiers, even after all parameter optimizations, the highest identification rates in a 10-class identification task using 3-second segments was about 52 % correct rate (weighted accuracy) and average class-dependent accuracy about 44 %.

In our comparisons, the distance-based (kNN, VQ) and probabilistic classifiers (GMM-ML, GMM-MI) performed somewhat better than the two SVM-based systems. Note that the first four classifiers (kNN, VQ, GMM-ML, GMM-MI) do direct scoring of MFCC vectors against the acoustic context models whereas linear sequence kernel SVM scoring boils down to computing an inner product between two model supervectors corresponding to the training and the test signals. Finding reliable model parameters for the test signal is challenging due to the extremely short durations. In fact, such degradation of sequence kernel SVMs over conventional GMM scoring for short durations has also been reported in speaker verification [22]. Detailed study on the effect of the test signal duration would be a natural future goal to focus on.

In the overall comparisons, GMM with MMI training ranked the highest while GMM supervector SVM ranked the lowest. However, none of the classifiers can be said to be an universal winner for all classes (contexts). This observation agrees well with an independent comparison of four classifiers in [1]. It is also evident that the accuracy varies greatly across different users and the city where the data was collected in. Because of such diversity, classifier selection and fusion methods would be also an interesting research direction to study. There was also some indication of possible device dependency (similar to channel or environmental variability in speech classification tasks). Thus, in addition to addressing the short duration problem, compensating for the biases due to different users, devices and imbalanced training set sizes should be considered.

7. References

[1] O. Räsänen, J. Leppänen, U. K. Laine, and J. P. Saarinen, "Comparison of classifiers in audio and acceleration based context classification in mobile phones," in *Proc. 19th European Signal Proc. Conf. (EUSIPCO 2011)*, Barcelona, Spain, August–September 2011, pp. 946–950.

[2] R. Duda, P. Hart, and D. Stork, *Pattern Classification*, Wiley Interscience, New York, second edition, 2000.

[3] F.K. Soong, A.E. Rosenberg A.E., B.-H. Juang, and L.R. Rabiner, "A vector quantization approach to speaker recognition," *AT & T Technical Journal*, vol. 66, pp. 14–26, 1987.

[4] D.A. Reynolds and R.C. Rose, "Robust text-independent speaker identification using Gaussian mixture speaker models," *IEEE Trans. on Speech and Audio Processing*, vol. 3, pp. 72–83, January 1995.

[5] P. Matějka, L. Burget, P. Schwarz, and J.H. Černocký, "Brno University of Technology system for NIST 2005 language recognition evaluation," in *Proceedings of Odyssey 2006: The Speaker and Language Recognition Workshop*, San Juan, PR, 2006, pp. 57–64.

[6] W.M. Campbell, D.E. Sturim, and D.A. Reynolds, "Support vector machines using GMM supervectors for speaker verification," *IEEE Signal Processing Letters*, vol. 13, no. 5, pp. 308–311, May 2006.

[7] W.M. Campbell, J.P. Campbell, D.A. Reynolds, E. Singer, and P.A. Torres-Carrasquillo, "Support vector machines for speaker and language recognition," *Computer Speech and Language*, vol. 20, no. 2-3, pp. 210–229, April 2006.

[8] A. J. Eronen, V. T. Peltonen, J. T. Tuomi, A. P. Klapuri, S. Fagerlund, T. Sorsa, G. Lorho, and J. Huopaniemi, "Audio-based context recognition," *IEEE Trans. Audio, Speech and Language Processing*, vol. 14, no. 1, pp. 321–329, January 2006.

[9] L. Ma, B. Milner, and D. Smith, "Acoustic environment classification," *ACM Trans. on Speech and Language Processing*, vol. 3, no. 2, pp. 1–22, July 2006.

[10] W. Dargie, "Adaptive audio-based context recognition," *IEEE Trans. on Systems, Man, and Cybernetics – Part A: Systems and Humans*, vol. 39, no. 4, pp. 715–725, July 2009.

[11] S. Chu, S. Narayanar, and C.-C.J. Kuo, "Environmental sound recognition with time-frequency audio features," *IEEE Trans. Audio, Speech and Language Processing*, vol. 17, no. 6, pp. 1142–1158, August 2009.

[12] M. Perttunen, M.V. Kleek, O. Lassila, and J. Riekki, "Auditory context recognition using SVMs," in *The Second International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM)*, Valencia, Spain, 2008, pp. 102–108.

[13] A.-M. Mesaros, T. Heittola, A. Eronen, and T. Virtanen, "Acoustic event detection in real life recordings," in

Proc. of the 18th European Signal Processing Conference (EUSIPCO 2010), Aalborg, Denmark, August 2010, pp. 1267–1271.

[14] T. Heittola, A.-M. Mesaros, A. Eronen, and T. Virtanen, "Audio context recognition using audio event histograms," in *Proc. of the 18th European Signal Processing Conference (EUSIPCO 2010)*, Aalborg, Denmark, August 2010, pp. 1272–1276.

[15] Y. Linde, A. Buzo, and R.M. Gray, "An algorithm for vector quantizer design," *IEEE Transactions on Communications*, vol. 28, no. 1, pp. 84–95, January 1980.

[16] C.M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, New York, 1996.

[17] L. Burget, P. Matejka, and J. Cernocky, "Discriminative training techniques for acoustic language identification," in *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, may 2006, vol. 1, p. I.

[18] D.A. Reynolds, T.F. Quatieri, and R.B. Dunn, "Speaker verification using adapted gaussian mixture models," *Digital Signal Processing*, vol. 10, no. 1, pp. 19–41, January 2000.

[19] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," *Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008, `http://www.csie.ntu.edu.tw/~cjlin/liblinear/`.

[20] M. Perttunen, M.V. Kleek, O. Lassila, and J. Riekki, "An implementation of auditory context recognition for mobile devices," in *2009 Tenth Int. Conf. on Mobile Data Management: Systems, Services and Middleware (MDM '09)*, Taipei, Taiwan, 2009, pp. 424–429.

[21] M. Kockmann, L. Burget, and J. Černocký, "Application of speaker- and language identification state-of-the-art techniques for emotion recognition," *Speech Communication*, vol. 53, pp. 1172–1185, 2011.

[22] M. McLaren, R. Vogt, B. Baker, and S. Sridharan, "Experiments in SVM-based speaker verification using short utterances," in *Odyssey 2010: The Speaker and Language Recognition Workshop*, Brno, Czech Republic, June 2010, pp. 83–90.