Comparing and combining classifiers for self-taught vocal interfaces

Lize Broekx, Katrien Dreesen, Jort F. Gemmeke, Hugo Van hamme

ESAT, KU Leuven, Leuven, Belgium

{katrien.dreesen,lize.broekx1}@student.kuleuven.be

{jort.gemmeke,hugo.vanhamme}@esat.kuleuven.be

Abstract

An attractive approach to enable the use of vocal interfaces by impaired users with dysarthric speech is the use of a system which learns from the end-user. To enable such technology, it is imperative that the learning is fast to reduce the time spent training the interface. In this paper we investigate to what extend various machine learning techniques are able to learn from only a single or a few spoken training samples. Additionally, we explore whether these techniques can be combined through boosting to improve the performance. Our evaluations on a small, but highly realistic home automation database reveal that non-negative matrix factorization seems best suited for fast learning and that some of the boosting approaches can indeed improve performance, especially for small amounts of training data.

Index Terms: vocal user interface, self-taught learning, machine learning, boosting

1. Introduction

Vocal user interfaces (VUIs) allow us to control a wide range of appliances and devices such as computers, smart phones, car navigation and other domestic devices and environments. While for most the use of a VUI is just a luxury, for individuals with a physical disability using a VUI can greatly improve their independence and quality of living, because for them operating and controlling devices would often require exhausting physical effort [1].

Conventional speech recognition systems employed in VUIs are trained by the developer using vast amounts of speech material. While offering impressive performances for users whose word choice, grammar and speech conforms to the training material used, performance suffers in the presence of accented, dialectical and disordered speech. A possible solution, adaptation of existing acoustic models, may not suffice for severe speech pathologies [2, 3, 4, 5, 6].

The goal of this research is to explore methods which allow training speech commands by the end-user himself. This way, the acoustic models of the VUI are maximally adapted to the end-user's speech while at the same time bringing development costs down. The challenge is to employ a learning strategy that can learn from only one or a few examples, in order to minimize the time the end-user spends on training the system. In this work, we will offer a comparison between multiple popular machine learning strategies to evaluate their effectiveness in developing a fast learning self-taught VUI.

In previous work, we obtained encouraging results on fast vocabulary acquisition [7] using non-negative matrix factorization (NMF). Although in that work the learning speed of acquiring acoustic models was investigated, it focused on larger amounts of training data than targeted in this work. Moreover, it considered a *multi-label* task in which spoken utterances were associated with multiple labels at once, which penalized other machine learning methods less suited for multi-label learning. In contrast, in this work we will focus on *speech classification*, labelling a spoken utterance with a single label.

Our contribution is twofold. First, we compare the performance of five machine learning techniques: Dynamic Time Warping (DTW), Gaussian Mixture Models (GMMs), Hidden Markov Models (HMMs), Support Vector Machines (SVMs) and Non-negative Matrix Factorization (NMF). Each of these techniques have their strengths and weaknesses; for example, while a HMM improves upon a GMM by being able to model temporal structure, it does require more parameters to be trained. When training with only one or a few training samples, this may lead to overfitting. Second, we investigate to what extent combining the aforementioned classification techniques, 'boosting', can improve results. We do this by comparing a number of combination rules operating at the class label posterior level [8].

The remainder of the paper is organised as follows. In Section 2 we give an overview of the speech classification methods that are investigated. In Section 3 we describe the various boosting approaches that will be considered. In Sections 4 and 5 we describe the experimental setup for evaluation on a small, but highly realistic home automation database collected in the ALADIN project [9]. The results of these experiments are presented in Section 6 and discussed in Section 7, and we conclude with our summary and thoughts for future work in Section 8.

2. Classification methods

In a speech classification problem an unlabelled speech signal X is assigned to one of the m possible commands $\{\omega_1, \ldots, \omega_m\}$. The classification methods Dynamic Time Warping (DTW), Gaussian Mixture Model (GMM) and Hidden Markov Model (HMM) use a spectrographic feature vector representation $\mathbf{X} = [\mathbf{x_1}, ..., \mathbf{x_T}]$ of a speech signal X, with T the number of frames. The number of rows of $\mathbf{X}$ is the dimension of the feature vector $\mathbf{x_t}$ for one frame.

The classification methods Non-negative Matrix Factorization (NMF) and Support Vector Machine (SVM) use a utterance based feature vector $\mathbf{x}$ of a speech signal X. The utterance based feature vector $\mathbf{x}$ is a column vector whose dimension N depends on the kind of feature vector. More information about the feature vectors used in this work for each of the techniques is given in Section 4.2.

A classification method evaluates how well the unlabelled speech signal X resembles speech signals associated with a command class ω_k. This similarity is expressed by the positive number $\Pi(\omega_k, X)$, which is method dependent. The larger $\Pi(\omega_k, X)$, the larger the similarity between X and the speech signals belonging to command ω_k. The speech signal X is assigned to the command ω_j with the highest similarity:

$$\text{assign } X \rightarrow \omega_j = \underset{\omega_k}{\text{argmax}}\, \Pi(\omega_k, X). \quad (1)$$

2.1. Dynamic Time Warping

Dynamic Time Warping (DTW) [10, 11, 12], is a method in which an unlabelled speech signal is compared with a large collection of labelled speech signals (exemplars) extracted from the training data. Since such a comparison needs to take different signal lengths and speech rate variations into account, DTW first finds an optimal alignment between each pair of utterances through non-linear time warping. The unlabelled speech signal is labelled with the command which is associated with the most similar exemplar.

As a learning method, DTW has the advantage that it makes optimal use of all the available labelled data, because all training samples can be used as an exemplar. A drawback of DTW is that the computational complexity of classification increases linearly with the number of training samples.

Formally, the similarity between the frames of the unlabelled speech signal X and the frames of each exemplar X_i is represented by a $T \times T_i$ matrix $\mathbf{D}_{X,X_i}$ containing the cosine distance between the spectrographic based feature vectors representations $\mathbf{X}$ and $\mathbf{X_i}$:

$$\mathbf{D}_{X,X_i} = \frac{\mathbf{X}'\mathbf{X_i}}{\|\mathbf{X}\|\|\mathbf{X_i}\|}. \quad (2)$$

The similarity between two speech signals is expressed as the score $\tilde{D}_{X,X_i}$ along the optimal path through the distance matrix $\mathbf{D}_{X,X_i}$. The optimal path, from the left upper corner to the right lower corner (Figure 1(a)), minimises the cumulative acoustic differences and the total number of steps. The optimal path is determined using a dynamical programming approach with the Needleman-Wunsch algorithm [13].

The unlabelled speech signal X is assigned to the command of the exemplar X_i with the highest similarity. The similarity between X and ω_k is expressed by the positive number

$$\Pi(\omega_k, X) = \max_{X_i \text{ of } \omega_k} \tilde{D}_{X,X_i}. \quad (3)$$

2.2. Gaussian Mixture Model

In a Gaussian Mixture Model (GMM), the probability density function is used to determine the acoustic likelihood of a command given the feature vectors of the unlabelled speech signal [14].

Using a GMM is attractive, because it is a parametric model in which the classification of an unlabelled speech signal takes the same time independent of the amount of training data. Another advantage is that the use of a parametric model typically allows better generalisation to unseen data, provided enough training data is available to accurately estimate the parameters.

Each command ω_k is represented by a weighted sum of multivariate Gaussian distributions

$$f_k(\mathbf{x_t}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{i=1}^{M} w_i N(\mathbf{x_t}, \boldsymbol{\mu_i}, \boldsymbol{\Sigma_i}), \quad (4)$$

with $\boldsymbol{\mu_i}$ the mean spectrographic based feature vector, $\boldsymbol{\Sigma_i}$ the covariance matrix, w_i the weights for each multivariate Gaussian distribution in the GMM and M the number of multivariate Gaussian distributions in the GMM. The weighted sum of multivariate Gaussian distributions for each command ω_k is trained on the collection of speech signals $\{X^{(1)}, \ldots, X^{(N)}\}$ in the training data belonging to command ω_k. By applying the Expectation Maximization algorithm [15], the mean spectrographic based feature vector $\boldsymbol{\mu_i}$, the covariance matrix $\boldsymbol{\Sigma_i}$ and the weights w_i are obtained for each multivariate Gaussian distribution in the GMM. The similarity between X and ω_k is expressed by the positive number

$$\Pi(\omega_k, X) = \exp\left(\frac{1}{T}\sum_{t=1}^{T} \log\left(f_k(\mathbf{x_t}, \boldsymbol{\mu}, \boldsymbol{\Sigma})\right)\right). \quad (5)$$

2.3. Hidden Markov Model

A Hidden Markov Model (HMM), the de facto standard model in automatic speech recognition, augments the GMM by taking the temporal structure of the speech into account.While known to be a powerful model for speech given enough training data, it is not clear a priori whether it can perform better than a GMM if there is very little training data and when whole-word HMMs to model the spoken commands are used rather than sub=words HMMs.

In a HMM for speech classification, the commands ω_k are represented by a sequence Q of states q [16, 14]. For speech classification the order in the speech signal X is important, therefore a left-to-right (Bakis) HMM (Figure 1(b)) is used. A HMM $\lambda_k = (\boldsymbol{\Pi}_k, \mathbf{A}_k, \mathbf{B}_k)$ is characterized by the initial probabilities $\pi_i^{(k)}$, the transition probabilities $a_{ij}^{(k)}$ and the emissions $b_i^{(k)}(\mathbf{x_t})$, where i and j are the state indices. The emissions $\mathbf{B_k}$ are a GMM $f_k(\mathbf{x_t}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ for each state.

The HMM λ_k is trained on the collection of speech signals $\{X^{(1)}, \ldots, X^{(N)}\}$ in the training data belonging to command ω_k by the Baum-Welch algorithm.

For an unlabelled speech signal X the optimal state sequence $\tilde{Q}$ in each HMM λ_k is calculated as

$$\tilde{Q} = \arg\max_{Q} P(Q|\mathbf{X}, \lambda_k) = \arg\max_{Q} \frac{P(Q, \mathbf{X}|\lambda_k)}{P(\mathbf{X}|\lambda_k)}. \quad (6)$$

The similarity between X and ω_k, which is obtained by applying the Viterbi algorithm, is the positive number

$$\Pi(\omega_k, X) = P(\mathbf{X}, q, |\lambda_k). \quad (7)$$

2.4. Support Vector Machine

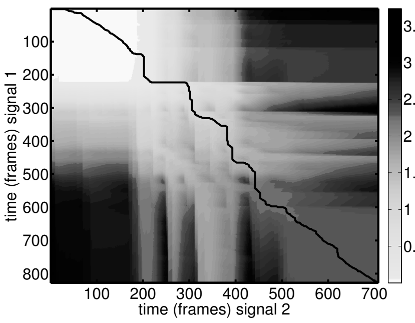
A Support Vector Machine is a linear classifier which is trained to be maximally discriminative between classes, possibly augmented by working in a high dimensional (kernalized) space [17]. SVMs are known to generalize well to unseen data, but it may be difficult to accurately train the hyperplane dividing classes with only few data points.

A SVM can be used for binary classification; to separate the m commands in our speech classification task we use the one-versus-one multi-label approach in this paper. For each pair of commands ω_k and ω_l, a binary classification problem is solved, which results in n_y binary classification problems with

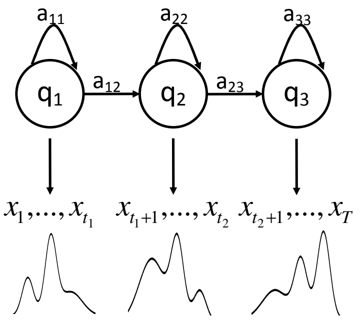
$$n_y = \frac{m(m-1)}{2}. \quad (8)$$

The hyperplane of the linear classifier that separates two commands ω_k and ω_l can be formulated as

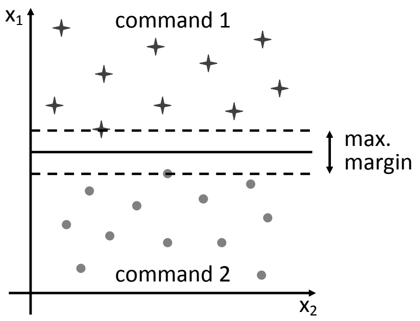
$$y_{\omega_k,\omega_l}(\mathbf{x}) = \mathbf{w}_{\omega_k,\omega_l}^T \mathbf{x} + b_{\omega_k,\omega_l} = \text{constant}, \quad (9)$$



(a) *Example of the optimal path in the distance matrix* $\mathbf{D}$ *between two speech signals as found by DTW.*



(b) *Example of a left-to-right HMM with 3 states* q_i*, transition probabilities* a_{ij} *and emissions* $b_i(\mathbf{x_t})$.



(c) *Optimal hyperplane of the SVM classifier maximizing the margin dividing two classes.*

Figure 1: *Graphical representation of the classification methods DTW, HMM and SVM.*

with $\mathbf{w}_{\omega_k,\omega_l} \in \mathbb{R}^n,\ \mathbf{x} \in \mathbb{R}^n,\ b_{\omega_k,\omega_l} \in \mathbb{R},\ y_{\omega_k,\omega_l} \in \mathbb{R}$. The unique hyperplane for a binary classification problem (Figure 1(c)) can be found by rescaling the problem such that the $\mathbf{x}$ closest to the hyperplane satisfy

$$|\mathbf{w}_{\omega_\mathbf{k},\omega_\mathbf{l}}{}^T\mathbf{x} + b_{\omega_k,\omega_l}| = 1. \tag{10}$$

The speech signal X is associated with label vector

$$\mathbf{y}(\mathbf{x}) = [y_{\omega_1,\omega_2}(\mathbf{x}), \ldots, y_{\omega_{m-1},\omega_m}(\mathbf{x})]^T, \tag{11}$$

with $\mathbf{x}$ the utterance based feature vector of X.

Each command ω_k is associated with a command label vector $\mathbf{y}^{(\mathbf{k})} \in \{-1,+1\}^{n_y}$. The similarity between the speech signal X and command ω_k is the positive number

$$\Pi(\omega_k, X) = \left(\mathbf{y}^{(\mathbf{k})} == \text{sign}(\mathbf{y}(\mathbf{x}))\right). \tag{12}$$

2.5. Non-negative Matrix Factorization

Non-negative Matrix Factorization is an approach which factorises the training data into a set of recurrent acoustic patterns and their activations. In a supervised setting, these acoustic patterns take on the distribution of the spoken commands, as well as the acoustic patterns of filler words which are shared between commands, e.g. the, and, please. As such, it can potentially leverage shared information between commands.

A scheme of the Non-negative Matrix Factorization (NMF) is given in Figure 2 [18]. The non-negative matrix $\mathbf{V}^{(\text{train})}$ consists of two parts, viz. $[\mathbf{V_0}^{(\text{train})}; \mathbf{V_1}^{(\text{train})}]$. Each of the m commands ω_k is represented by a vector representation $\mathbf{y}^{(\mathbf{k})}$, which is a zero m dimensional vector with a 1 on position k. The columns of the matrix $\mathbf{V_0}^{(\text{train})}$ contain the vector representation $\mathbf{y}^{(\mathbf{k})}$ of the command ω_k for each utterance in the training data, so $\mathbf{V_0}^{(\text{train})}$ is a matrix of dimension $m \times N_{\text{train}}$. The columns of the matrix $\mathbf{V_1}^{(\text{train})}$ contain the utterance based feature vector $\mathbf{x}$ for each utterance X in the training data, so $\mathbf{V_1}^{(\text{train})}$ is a matrix of dimension length($\mathbf{x}$) $\times$ N_{train}. A low rank representation for $\mathbf{V_1}^{(\text{train})}$ is obtained by factorizing $\mathbf{V}^{(\text{train})}$ as the product of two non-negative matrices $\mathbf{W}$ and $\mathbf{H}^{(\text{train})}$, viz.

$$\begin{bmatrix}\mathbf{V_0}^{(\text{train})}\\ \mathbf{V_1}^{(\text{train})}\end{bmatrix} \approx \begin{bmatrix}\mathbf{W_0}\\ \mathbf{W_1}\end{bmatrix}\mathbf{H}^{(\text{train})} = \mathbf{W}\mathbf{H}^{(\text{train})}. \tag{13}$$

The number of columns of $\mathbf{W}$, i.e. the number of acoustic patterns to extract from the training data, is fixed in advance. The matrices $\mathbf{W}$ and $\mathbf{H}^{(\text{train})}$ are obtained by iteratively minimizing the Kullback-Leibler divergence [19] between $\mathbf{V}^{(\text{train})}$ and $\mathbf{W}\mathbf{H}^{(\text{train})}$. After training, the matrix $\mathbf{W_1}$ contains as columns the acoustic patterns that exist in the training data. Element $\mathbf{W_0}(k,j)$ indicates if the acoustic pattern in column j of $\mathbf{W_1}$ is present in command ω_k. The element $\mathbf{H}^{(\text{train})}(i,j)$ indicates how much the acoustic pattern in column i of $\mathbf{W_1}$ is present in utterance $X^{(j)}$ of the training data (column j of $\mathbf{V_1}^{(\text{train})}$).

For speech classification with NMF, the utterance based feature vector $\mathbf{x}$ of speech signal X is used as vector $\mathbf{V_1}^{(\text{test})}$. By minimizing the Kullback-Leibler divergence between $\mathbf{V_1}^{(\text{test})}$ and $\mathbf{W_1}\mathbf{H}^{(\text{test})}$, the vector $\mathbf{H}^{(\text{test})}$ is calculated

$$\mathbf{V_1}^{(\text{test})} \approx \mathbf{W_1}\mathbf{H}^{(\text{test})}. \tag{14}$$

An approximation of $\mathbf{V_0}^{(\text{test})}$, containing the representation of the command for each utterance in the test data, is given by the activation vector $\mathbf{A}$

$$\mathbf{V_0}^{(\text{test})} \approx \mathbf{A} = \mathbf{W_0}\mathbf{H}^{(\text{test})}. \tag{15}$$

The similarity between speech signal X and command ω_k is given by the positive number

$$\Pi(\omega_k, X) = A(k). \tag{16}$$

3. Combining methods

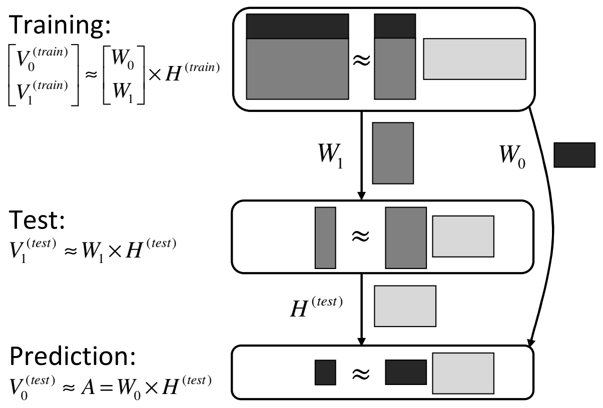
Each classification method uses different mechanisms to assign an unlabelled speech signal X to the most probable command ω_j. As a result, not all classification methods assign an unlabelled speech signal to the same command, so the misclassifications differ. In this paper we investigate whether combining R different classification methods, i.e. boosting [8], might reduce the number of misclassifications. Each method r calculates a positive number $\Pi_r(\omega_k, X)$ for each command ω_k.

The numbers $\{\Pi_r(\omega_k, X)\}_{k=1}^M$ are normalized to construct a probability vector $\mathbf{p_r}$ for each method. The combination of the classification methods assigns the unlabelled speech signal X to the command ω_j with the highest a posteriori probability

$$\text{assign } X \to \omega_j = \underset{\omega_k}{\text{argmax}}\ P(\omega_k|\mathbf{p_1}, ..., \mathbf{p_R}). \tag{17}$$

Possible combination rules are the product rule, sum rule, maximum rule, minimum rule, median rule and majority rule. Table 1 gives the a posteriori probability for these rules.

Figure 2: *Scheme of Non-negative Matrix Factorization.*


Training:
$\begin{bmatrix}V_0^{(train)}\\V_1^{(train)}\end{bmatrix} \approx \begin{bmatrix}W_0\\W_1\end{bmatrix} \times H^{(train)}$
$\approx$
W_1
W_0
Test:
$V_1^{(test)} \approx W_1 \times H^{(test)}$
$\approx$
$H^{(test)}$
Prediction:
$V_0^{(test)} \approx A = W_0 \times H^{(test)}$
$\approx$

The combination rules in Table 1 make the assumption that each method assigns a speech signal to a command independently of the other methods, which results in

$$P(\mathbf{p_1}, ..., \mathbf{p_R}|\omega_k) = \prod_{i=1}^{R} P(\mathbf{p_i}|\omega_k). \tag{18}$$

In this paper the extra assumption of equal a priori probabilities is made for each rule. The a posteriori probability of the product rule is obtained by repeatedly applying Bayes' rule. In the sum rule the assumption is made that the a posteriori probabilities $P(\omega_k|\mathbf{p_i})$ calculated by the different methods do not significantly differ from the a priori probabilities $P(\omega_k)$. The maximum rule is obtained starting from the sum rule and using the relation

$$\frac{1}{R}\sum_{i=1}^{R} P(\omega_k|\mathbf{p_i}) \leq \max_i P(\omega_k|\mathbf{p_i}). \tag{19}$$

The minimum rule is obtained starting from the product rule and using the relation

$$\prod_{i=1}^{R} P(\omega_k|\mathbf{p_i}) \leq \min_i P(\omega_k|\mathbf{p_i}). \tag{20}$$

The a posteriori probability of the median rule is obtained by replacing the mean by the more robust median in the sum rule. In the majority rule the assumption is made that the a posteriori probabilities $P(\omega_k|\mathbf{p_i})$ calculated by the different methods do not significantly differ from the a priori probabilities $P(\omega_k)$.

4. Experimental setup

4.1. Dataset

The experiments are performed on the first home automation dataset (DOMOTICA-1) of the ALADIN project [9]. The dataset consists of non-pathological speech commands which were recorded in a realistic setting, i.e. a fully automated room using a wizard-of-oz device control. The commands were prompted using visual cues (a video) on a computer screen. In order to simulate situations with environmental noise, recordings were also made with a concurrent sound source. In addition to a close-talk microphone, multichannel audio recordings were made with multiple microphone arrays, placed near the user, on walls and near the optional noise source.

Table 1: *A posteriori probabilities for combination rules in assumption of equal a priori probabilities.*

rule	$P(\omega_k|\mathbf{p_1}, ..., \mathbf{p_R})$
product	$\prod_{i=1}^{R} P(\omega_k|\mathbf{p_i})$
sum	$\sum_{i=1}^{R} P(\omega_k|\mathbf{p_i})$
maximum	$\max_i P(\omega_k|\mathbf{p_i})$
minimum	$\min_i P(\omega_k|\mathbf{p_i})$
median	$\operatorname{med}_i P(\omega_k|\mathbf{p_i})$
majority	$\frac{1}{R}\sum_{i=1}^{R} \Delta_{k,i}$
	with $\Delta_{k,i} = \begin{cases} 1 & \text{if } \omega_k = \operatorname{argmax}_{\omega_l} P(\omega_l|\mathbf{p_i}) \\ 0 & \text{otherwise} \end{cases}$

The noisy recordings were created by playing a radio in the background with a sound level of 60dB Sound Pressure Level, which is the sound level of average speech. It was ensured that the measured SNR to the nearest microphone remained above 15dB.

The dataset consists of 27 test subjects of which 20 are of the targeted user group. Each person was asked to go repeatedly through a list of 33 different actions, until a recording time of 30 minutes was reached, yielding a dataset of 1888 commands for the target group. In addition to this set, longer recording sessions with 7 non-target users were carried out, yielding 1699 spoken commands.

The experiments are performed on persons 5,7,20,22 and 26, on the noisy dataset recorded with the close-talk microphone. These speakers were selected because they were the only speakers with at least 3 spoken samples of each command. We will refer to them by these numbers to keep correspondence with other work on the same dataset.

4.2. Acoustic representation

In this paper two classes of feature vectors are used: spectrographic based feature vectors and utterance based feature vectors [20, 21]. The MFCC, MFCCDD and MIDA feature vectors are used as spectrographic based feature vectors. The GMM-supervector and the feature vector based on the histogram of acoustic occurrences and co-occurrences (sumHAC) are used as utterance based feature vector [18]. In this paper a Hamming window of size 30 ms is used with frame shifts of 15 ms. A pre-emphasis of 0.95 is used.

The MFCC feature vectors are obtained by applying an Inverse Discrete Cosine Transform (IDCT) to log-Mel spectra. In this paper, we use 12 MFCCs in each frame, in addition to the log energy, resulting in 13-dimensional feature vectors. To obtain the MFCCDD feature vectors, the 13-dimensional MFCC feature vectors are augmented with their first and second order differences (Δ- and $\Delta\Delta$-features), yielding a total of 39 coefficients per frame.

The Mutual Information Discriminant Analysis (MIDA) feature vectors are obtained with a linear transformation that maximizes the separability between different classes of input frames [22]. In this paper, we determine Δ- and $\Delta\Delta$-features on the 24 log-Mel spectral features, leading to 72-dimensional input vectors. On these representations we then perform the MIDA-transformation, separating the classes in the input space and at the same time reducing its dimensionality from 72 to 39.

The spectrographic based feature vectors are the starting point to construct the utterance based feature vectors. The GMM-supervector combines 60-dimensional MFCCDD spectrographic based feature vectors to one high-dimensional utterance based feature vector [21]. The construction of a GMM-supervector consists of three steps. The first step is training a Gaussian Mixture Model Universal Background Model (GMM UBM). To be more robust, a trained GMM UBM with 512 multivariate Gaussian distributions is used. The development data set used to train the UBM includes over 30,000 speech recordings and was sourced from NIST 2004-2006 SRE databases, LDC releases of Switchboard 2 phase III and Switchboard Cellular (parts 1 and 2) [23]. The second step in constructing a GMM-supervector is adapting the means of the multivariate Gaussian distributions of the GMM UBM according to the spectrographic based feature vectors of the speech signal. In the last step the 512 adapted means of dimension 60 are placed in a column vector, which gives the resulting 30720-dimensional GMM-supervector of the speech signal.

The utterance based feature vector based on the histogram of acoustic occurrences and co-occurrences (sumHAC) is constructed by applying a k-means clustering algorithm [16] to the MFCCDD spectrographic based feature vectors to obtain a codebook [18]. Each spectrographic based feature vector is assigned to a prototype vector in the codebook by means of an extension (softVQ) to vector quantization [24]. With softVQ a frame based feature vector characterized by its proximity to multiple prototypes is obtained. Proximity is measured as the posterior probability of a collection of Gaussians, much like in semi-continuous HMMs.

In this paper Voice Activity Detection (VAD) is used to improve the performance of a classification method [25]. By distinguishing speech and silence frames in the speech signal, both training and classification are only based on the speech frames. The distinction between speech and silence frames is made based on the energy in the frame.

4.3. Classification methods

Below, we detail the acoustic representations and parameter settings for each of the methods. To ensure a best-case scenario for each of the methods, we optimised the settings for each method individually.

DTW employs MFCC feature vectors. The use of MFCCDD and MIDA features was explored in a pilot experiment, but did not result in significant improvements.

Both the GMM and the HMM use the spectrographic MIDA feature vectors. The full-covariance GMM consists of 10 mixtures, while the GMM employed in the HMM uses 5 mixtures and a three state left-to-right HMM. The implementation uses the logarithm of the probabilities to obtain a numerically stable implementation [26].

The linear kernel SVM operates on GMM-supervectors, tuned using a cross-validation grid search. Finally, NMF is applied to the utterance based feature vector sumHAC, constructed starting from the spectrographic based feature vector MFCCDD and using the vector quantization method softVQ. In this paper codebooks of size 50 are used for softVQ and 36 acoustic patterns are extracted from the training data (columns of $\mathbf{W}$), where 3 are used for filler words.

Table 2 gives an overview of the setting for the different classification methods.

Table 2: *Settings for each classification method.*

method	setting
DTW	MFCC
GMM	MIDA, 10 mixtures
HMM	MIDA, 3 states, 5 mixtures
SVM	GMM-supervector (MFCCDD)
NMF	sumHAC (MFCCDD, softVQ)

5. Experiments

5.1. Comparing classifiers

For the experiment to determine the best classification method only 22 commands with 3 examples or more in the noisy recording scenario of person 5, 7, 20, 22 and 26 are considered. There is not enough data for the other persons in the data to investigate the influence of the number of examples for each command on the classification.

In a small dataset, the division of the commands in groups with cross-validation is more important than in larger datasets. Therefore an adaptation of cross-validation is used in the experiments. The commands are randomly divided in groups with the requirement that in each group there is exactly one speech signal belonging to each command. The least frequent command in the dataset determines the number of disjunct groups. The remaining speech signals are not used in the experiments.

To determine the accuracies of the different classification methods for k examples for each command, the classification methods are trained on k disjunct groups and tested on 1 disjunct group. All possible combinations of training groups and test group are considered. To minimize the influence of the division in groups, the experiments are repeated with 5 different random divisions in groups.

5.2. Combining classifiers

For the experiment to determine the influence of boosting on the accuracy only 22 commands with 3 examples or more in the noisy dataset of person 26 are considered. For each method $i \in \{\text{DTW, GMM, HMM, SVM, NMF}\}$ the 22×22 matrix $\mathbf{\Pi}_i$ is constructed with as (k, j) element the number $\Pi_i(\omega_k, X^{(j)})$ where ω_k is the considered command and $X^{(j)}$ a speech signal in the testdata. Each column in the matrix $\mathbf{\Pi}_i$ is scaled to a probability vector, resulting in the matrix $\mathbf{P}_i$. The adaptation of cross validation, as discussed in section 5.1, is used and the experiments are repeated with 5 different random divisions in groups to minimize the influence of the division in groups. The matrices $\mathbf{P}_i$ of the different combinations of test group and trainingsdata with 5 random divisions in groups are concatenated in the matrix $\mathbf{P}_{i,\text{TOT}}$ for n examples in the trainingsdata. The matrix $\mathbf{P}_{i,\text{TOT}}$ is of dimension $22 \times N_{\text{TOT}}$, where N_{TOT} is the product of the number of commands (22), the number of random divisions in groups (5), the number of test groups and the number of different trainingsdata for n examples in the trainingsdata.

The goal of boosting is to achieve an improvement in the accuracies with respect to the individual classification methods. In the boosting experiment, each method i is assigned a positive weight w_i, so $P(\omega_k|\mathbf{p_i})$ becomes $w_i P(\omega_k|\mathbf{p_i})$. For the

Table 3: *Accuracies obtained with classification methods in noisy recording scenario for person 5, 7, 20, 22 and 26.*

Person 5					
N_{EX}	DTW	GMM	HMM	SVM	NMF
1	48.6	30.5	17.9	5.3	**56.4**
2	56.1	72.4	56.1	6.7	**87.0**

Person 7					
N_{EX}	DTW	GMM	HMM	SVM	NMF
1	23.9	28.3	16.5	4.8	**42.4**
2	30.9	51.5	32.7	3.0	**62.7**

Person 20					
N_{EX}	DTW	GMM	HMM	SVM	NMF
1	45.0	47.3	29.2	10.8	**47.6**
2	54.8	71.2	55.8	30.6	**80.0**

Person 22					
N_{EX}	DTW	GMM	HMM	SVM	NMF
1	**67.0**	45.9	29.7	10.2	49.8
2	73.6	82.1	71.2	13.9	**86.7**

Person 26					
N_{EX}	DTW	GMM	HMM	SVM	NMF
1	**66.5**	45.5	34.0	23.9	63.4
2	75.8	69.3	66.0	53.1	**81.6**
3	80.2	78.4	79.2	69.0	**87.5**
4	83.5	83.3	84.7	77.8	**90.8**
5	85.6	85.9	88.2	81.5	**91.8**

majority rule the definition of $\Delta_{k,i}$ becomes

$$\Delta_{k,i} = \begin{cases} w_i & \text{if } \omega_k = \underset{\omega_l}{\text{argmax}}\ P(\omega_l|\mathbf{p_i}) \\ 0 & \text{otherwise.} \end{cases} \quad (21)$$

In this way a better classification method has a higher influence on the a posteriori probability.

The optimal weights w_i of the matrices $\mathbf{P}_{i,\text{TOT}}$ of the methods i in the boosting rules are obtained using grid search for 1 example for each command. The grid search of the weight w_i for each method i is between 0 and 1 with step size 1/4. Each possible combination of the weights $\mathbf{w} = [w_{\text{DTW}}, w_{\text{GMM}}, w_{\text{HMM}}, w_{\text{SVM}}, w_{\text{NMF}}]$ is scaled to one. For the combination rules, the weights $\mathbf{w} = [1/4, 1/4, 1/4, 1/4, 1/4]$ and $\mathbf{w} = [1, 1, 1, 1, 1]$ result in the same normalized weights $\mathbf{w} = [1/5, 1/5, 1/5, 1/5, 1/5]$. Only the unique normalized weights are considered. If there are multiple weights $\mathbf{w}$ which result in the highest accuracies, the first $\mathbf{w}$ is used as optimal weight.

6. Results

6.1. Comparing classifiers

Table 3 shows the accuracies with different N_{EX} examples for each command in the trainingsdata and for person 5, 7, 20, 22 and 26 with the settings of Table 2.

Table 3 shows that NMF gives the highest accuracies for each person, except for person 22 and 26 with one example for each command. The lowest accuracies are obtained with classification method SVM. The accuracies improve with an increasing number examples N_{EX} for each command, except for person 7 with SVM. The accuracies of person 7 are significantly lower than those of the other persons. For person 5 the second highest accuracies are obtained with DTW and NMF for 1 and 2 examples for each command respectively. For person 7 and 20 the second highest accuracies are obtained with GMM. The second highest accuracies for person 22 are obtained with NMF (1 example) and GMM (2 examples). For person 26 the second highest accuracies are obtained with NMF (1 example), DTW (2-3 examples) and HMM (4-5 examples). The accuracies of person 26 with GMM are initially higher than those of HMM (1-2 examples), but for more examples (3-5) the accuracies of HMM are higher.

Table 4: *Weights for person 26 with combination rules.*

rule	DTW	GMM	HMM	SVM	NMF
product	4/5	0	0	0	1/5
sum	4/5	0	0	0	1/5
maximum	4/5	0	0	0	1/5
minimum	1/17	4/17	4/17	4/17	4/17
median	1/6	4/6	0	0	1/6
majority	2/6	1/6	0	1/6	2/6

Table 5: *Accuracies for person 26 with combination rules.*

N_{EX}	product	sum	max	min	med	maj
1	**73.0**	70.6	70.7	23.9	66.7	70.2
2	**85.2**	83.8	83.5	52.7	79.3	83.4
3	**89.3**	88.8	87.8	68.7	84.4	88.3
4	**91.2**	91.0	90.6	77.5	88.2	90.8
5	**92.9**	92.0	91.1	81.2	90.5	91.7

6.2. Combining classifiers

In Table 4 the optimal weights are shown that are obtained by grid search with 1 example for each command in the training data.

Table 4 shows that the product rule, sum rule and maximum rule assign an unlabelled speech signal X based on the probability vectors $\mathbf{p_{DTW}}$ and $\mathbf{p_{NMF}}$. The median rule uses the probability vectors $\mathbf{p_{DTW}}$, $\mathbf{p_{GMM}}$ and $\mathbf{p_{NMF}}$. The majority rule uses all probability vectors except $\mathbf{p_{HMM}}$. The minimum rule bases the assignment on the probability vectors of all classification methods. The according non normalized weights in the grid search are $\mathbf{w} = [1/4, 1, 1, 1, 1]$. In the minimum rule the probability vector $\mathbf{p_{DTW}}$ gets the smallest weight. In Table 5 the accuracies obtained with the individual classification methods and the different boosting rules with weights as in Table 4 are shown.

Table 5 shows that the highest individual accuracies are obtained with the classification method DTW (1 example) and NMF (2-5 examples). The classification method SVM gives the lowest accuracies. The more examples for each command in the training data, the higher the accuracies of the individual classification methods are.

In Table 5 a significant improvement in the accuracies is observed for the product rule and sum rule with respect to the highest accuracies obtained with the individual classification methods. The maximum rule and majority rule give higher accuracies for 1 to 3 examples for each command in the training data with respect to the individual methods. For more examples for each command in the training data, the accuracies obtained with the maximum rule and majority rule are similar to those obtained with NMF. The accuracies obtained with median rule are lower than the accuracies obtained with the best individual classification method NMF, except for 1 example for each command. The median rule performs better than all individual classification methods except NMF. The minimum rule gives similar accuracies as the worst individual classification method SVM.

7. Discussion

7.1. comparing classifiers

When comparing the classifiers in Table 3, we observe a very large difference between classification methods. Since for speaker 26, the difference between methods becomes substantially smaller with increasing numbers of examples per command, we can indeed attribute most of these difference to the amount of training data used. Although it is difficult to set a target accuracy which suffices for practical applicability of these techniques, it is encouraging that for most speakers 2 examples suffice to achieve 80 to 90 % accuracy. The lower accuracies of speaker 7, although still at 62.7 % for NMF, are due to a speech impairment. Informal listening tests revealed that for this speaker, the spoken commands are difficult to understand even for humans.

As expected, DTW achieves good results when presented with only a single training sample, but is outperformed by models which learn their parameters as soon as there is more training data. It is interesting to observe that HMM is outperformed by the simpler GMM until at least 3 training samples are presented. It seems that even though the GMM employed by the HMM is smaller (5 vs 10 mixtures), the larger number of parameters that needs to be trained is still problematic for small training sizes.

Of all the methods, the SVM performs the worst, with results almost at chance level for some speakers. The results on speaker 26 indicate once again, that with more data the results become comparable. NMF, on the other hand, is able to perform well both with little and larger amounts of training data. Presumably, this is due to its capability to use some of its recurrent patterns to model phenomena such as filler words, which allows the recurrent patterns modelling commands to be more discriminative. Although these results do not allow us to investigate the accuracy when presented with much more data (hundreds of examples), but results in [7] do indicate that also in these regimes, NMF be adequate.

7.2. Combining classifiers

Unfortunately, the amount of data available did not allow us to explore methods which learn the boosting weights from the data. Our experiments therefore show an upper limit on the performance gains that can be expected using boosting. Additionally, the weights that are obtained allow us to judge which methods offer complementary information.

The product rule, sum rule and maximum rule only use the 2 classification methods DTW and NMF which have the highest accuracies for 1 example for each command. The weight w_{DTW} is higher than w_{NMF} because the accuracy of DTW is higher than that of NMF for 1 example for each command. Since these rules have the highest weight to the best classification method, the best accuracies are obtained for these rules in comparison with the other boosting rules. This effect is only visible for a few examples for each command. An explanation for this is that the best classification method DTW for 1 example gains little in comparison with the other classification methods when increasing the amount of examples for each command.

The accuracies obtained with the minimum rule are not more than the accuracies obtained with the worst classification method SVM. It is plausible that the minimum rule only gives good results if the entropy of the probabilities of the different commands for the classification methods is small (probabilities of different commands in one classification method are close together). This is not the case for the considered classification methods. In this experiment, the minimum rule has the opposite effect of what is expected of a boosting rule.

The median rule does not take the classification method HMM and SVM, which have the lowest accuracies for 1 example for each command. The obtained accuracies are similar to the accuracies obtained with NMF. This is achieved by giving the best classification methods NMF and DTW a weight such that their weighted probability vector is median.

The majority rule assigns a relative high weight to the 2 best classification methods DTW and NMF for 1 example for each command, as expected. GMM and SVM also get a non zero weight. HMM is assigned a zero weight, which is explained by the fact that HMM makes more and similar misclassifications than GMM, because of the lack of data for 1 example for each command.

The effect of boosting decreases when increasing the amount of examples for each command. This is explained by the weights are trained on 1 example for each command and the changing order of best classification methods. Some classification methods (HMM, SVM) need more data to obtain a higher accuracy, while DTW gains relatively little in accuracy when increasing the number of data.

8. Conclusions

In this paper the performance of the classification methods Dynamic Time Warping (DTW), Gaussian Mixture Model (GMM), Hidden Markov Model (HMM), Support Vector Machine (SVM) and Non-negative Matrix Factorization (NMF) are investigated on a speech classification task. Evaluations on a realistic home automation database show that NMF is best suited for speech classification with a small amount of data. Next, we investigated if combining different methods through boosting might improve the classification. Both the product rule and the sum rule give higher accuracies than the best individual classification method. The gain of combining classification methods is higher for a small amount of data. Future work will focus on exploring methods to learn the boosting weights on small amounts of training data.

9. Acknowledgements

The research of Jort F. Gemmeke is funded by IWT-SBO grant 100049.

10. References

[1] J. Noyes and C. Frankish, “Speech recognition technology for individuals with disabilities,” *Augmentative and Alternative Communication*, vol. 8, no. 4, pp. 297–303, 1992.

[2] H. Christensen, S. Cunningham, C. Fox, P. Green, and T. Hain, “A comparative study of adaptive, automatic recognition of disordered speech,” in *Proc Interspeech 2012*, Portland, Oregon, US, Sep 2012.

[3] K. T. Mengistu and F. Rudzicz, “Comparing humans and automatic speech recognition systems in recognizing dysarthric speech,” in *Proceedings of the Canadian Conference on Artificial Intelligence*, 2011.

[4] H. V. Sharma and M. Hasegawa-Johnson, "State transition interpolation and map adaptation for hmm-based dysarthric speech recognition," in *HLT/NAACL Workshop on Speech and Language Processing for Assistive Technology (SLPAT)*, 2010, pp. 72–79.

[5] F. Rudzicz, "Acoustic transformations to improve the intelligibility of dysarthric speech," in *Proceedings of the Second Workshop on Speech and Language Processing for Assistive Technologies (SLPAT2011)*, 2011.

[6] M. S. Hawley, P. Enderby, P. Green, S. Cunningham, S. Brownsell, J. Carmichael, M. Parker, A. Hatzis, P. O'Neill, and R. Palmer, "A speech-controlled environmental control system for people with severe dysarthria," *Medical Engineering & Physics*, vol. 5, no. 29, pp. 586 – 593, 2007.

[7] J. Driesen, J. Gemmeke, and H. Van hamme, "Weakly supervised keyword learning using sparse representations of speech," in *Proceedings of the 36th International Conference on Acoustics, Speech and Signal Processing*, Kyoto, Japan, 2012.

[8] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, "On combining classifiers," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 3, pp. 226–239, 1998.

[9] ALADIN, "Adaptation and Learning for Assistive Domestic Vocal INterfaces," Project Page: http://www.esat.kuleuven.be/psi/spraak/projects/ALADIN.

[10] T. N. Sainath, B. Ramabhadran, D. Nahamoo, D. Kanevsky, D. V. Compernolle, K. Demuynck, J. F. Gemmeke, J. R. Bellegarda, and S. Sundaram, "Exemplar-based processing for speech recognition," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 98–113, 2012.

[11] D. Ellis, "Dynamic Time Warp (DTW) in Matlab," Web resource, available: http://www.ee.columbia.edu/ dpwe/resources/matlab/dtw/, 2003.

[12] M. De Wachter, M. Matton, K. Demuynck, P. Wambacq, R. Cools, and D. Van Compernolle, "Template-based continuous speech recognition," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 15, no. 4, pp. 1377–1390, 2007.

[13] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of molecular biology*, vol. 48, no. 3, pp. 443–453, 1970.

[14] D. Jurafsky and J. H. Martin, *Speech and language processing: an introduction to natural language processing, computational linguistics and speech recognition*, 2nd ed. Pearson International Edition, 2009.

[15] S. K. Ng, T. Krishnan, and G. J. McLachlan, "The EM algorithm," *Handbook of computational statistics*, vol. 1, pp. 137–168, 2004.

[16] X. Huang, A. Acero, H.-W. Hon *et al.*, *Spoken language processing*. Prentice Hall PTR New Jersey, 2001, vol. 15.

[17] J. A. K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle, *Least Squares Support Vector Machines*. World Scientific Publishing Co. Pte. Ltd., 2002.

[18] J. Driesen, "Discovering words in speech using matrix factorization," Ph.D. dissertation, Ph. D. dissertation, KU Leuven, ESAT, 2012.

[19] T. M. Cover and J. A. Thomas, *Elements of information theory*. Wiley-interscience, 2012.

[20] J. Driesen, J. F. Gemmeke, and H. Van hamme, "Data-driven speech representations for NMF-based word learning," in *Proc. SAPA-SCALE*, 2012, pp. 98–103.

[21] M. H. Bahari *et al.*, "Speaker age estimation using Hidden Markov Model weight supervectors," in *Information Science, Signal Processing and their Applications (ISSPA), 2012 11th International Conference on*. IEEE, 2012, pp. 517–521.

[22] K. Demuynck, J. Duchateau, and D. Van Compernolle, "Optimal feature sub-space selection based on discriminant analysis," in *Proc. Eurospeech*, vol. 3, 1999, pp. 1311–1314.

[23] M. H. Bahari, R. Saeidi, H. Van hamme, and D. van Leeuwen, "Accent recognition using i-vector, gaussian mean supervector and gaussian posterior probability supervector for spontaneous telephone speech," *Proceedings ICASSP 2013*, 2013.

[24] S. Meng and H. Van hamme, "Coding Methods for the NMF Approach to Speech Recognition and Vocabulary Acquisition," 2011.

[25] J. Ramirez, J. M. Górriz, and J. C. Segura, "Voice activity detection. fundamentals and speech recognition system robustness," *Robust Speech Recognition and Understanding*, pp. 1–22, 2007.

[26] T. P. Mann, "Numerically stable hidden Markov model implementation," *An HMM scaling tutorial*, pp. 1–8, 2006.