IDIAP RESEARCH REPORT

Hybrid generative-discriminative models for speech and speaker recognition

Le Quan [a], Samy Bengio [a]

IDIAP–RR 02-06

March 2002

Dalle Molle Institute for Perceptual Artificial Intelligence • P.O.Box 592 • Martigny • Valais • Switzerland

phone +41 − 27 − 721 77 11
fax +41 − 27 − 721 77 12
e-mail secretariat@idiap.ch
internet http://www.idiap.ch

[a] IDIAP, P.O. Box 592, CH-1920 Martigny, Switzerland

IDIAP Research Report 02-06

Hybrid generative-discriminative models for speech and speaker recognition

Le Quan , Samy Bengio

March 2002

Abstract. Generative probability models such as Hidden Markov Models are usually used for modeling sequences of data because of their ability to handle variable size sequences and missing information. On the other hand, because of their discriminative properties, discriminative models like Support Vector Machines (SVMs) usually yield better performance in classification problem and can construct flexible decision boundaries. An ideal classifier should have all the power of these two complementary approaches. A series of recent papers has suggested some techniques for mixing generative models and discriminative models. In one of them a fixed size vector (the Fisher score) containing sufficient statistics of a sequence is computed for a previously trained HMM and can then be used as input to a discriminative model for classification. The purpose of this project is thus to study, experiment, enhance and adapt these new approaches of integrating discriminative models such as SVM into generative models for sequence processing problems, such as speaker and speech recognition.

1 Introduction

It is not easy to deal with speech, videos, text and biosequences using simple statistical classification methods because the data to be classified is often represented as sequences or arrays of variable length and may have been distorted in particular way. The common solution for this problem is to estimate a generative model such as Hidden Markov Models (HMMs) for that data and then using Bayesian criterion to classify the data.

However it is well known that for classification problems, a better solution should in theory be to use a discriminative framework: In that case instead of constructing a model independently for each class, one construct a unique model that decides where the boundaries between classes are.

One of the latest discriminative model developed in the 90's is the Support Vector Machine (SVM), which computes a linear combination of the most important examples (the support vectors) in a high dimensional space (feature space or kernel space). This model has nice theoretical properties but unfortunately can not be used easily for sequence data such as speech or biosequences due to their variable lengths.

Recently some papers have proposed some methods that incorporate both SVMs and statistical models in a way that the robustness and flexibility of the generative models favorably combine with the discriminative power of the SVM. In one of the papers [13], the authors use generative models built from multiple sequences (in this case HMM) as a way of extracting features from sequences. This maps all sequences to points in a Euclidean feature space of fixed dimensions (the Fisher score) and then uses discriminative models to classify the points representing the sequences. This method has recently been applied successfully to biosequences classification. Providing several advantages, such an approach has also been proved to be always at least as good as the results obtained from generative Bayesian Maximum A Posteriori (MAP) criterion.

In this report we propose to study, experiment and eventually enhance these new approaches for sequence processing problems, such as speaker and speech recognition. It also will be the subject for our upcoming thesis.

In speaker verification, one has to validate, on the basis of a speech signal, the claimed identity of a speaker. This is clearly a classification task where the system has to discriminate between the client and the impostor. In speech recognition, one has to translate a speech sequence into a sequence of words, each word being represented by a sequence of phonemes. Each phoneme is usually modeled by a HMM and again the methods discussed above can be used to discriminate between the different phoneme models. In this case, the method will have to be adapted for solving the problem of signal segmentation.

The rest of the paper is organized as follow. Section 2 gives brief introductions to state-of-the-art techniques: generative models, discriminative models, combining methods as well as the speech and speaker recognition problems. Our research plan is described in Section 3. Preliminary experiments on the speaker verification problem and results are presented in Section 4. Finally, Section 5 concludes the report.

2 State of the art

2.1 Generative models and HMMs

Hidden Markov Models (HMMs) are statistical models for modeling sequential data, and have been used successfully in artificial intelligence, pattern recognition, speech processing and biosequences modeling. A good introduction can be found in [22]. The joint probability of a sequence of observations $y_1^T = y_1, y_2, , ..y_T$ can always be factored as

$$P(y_1^T) = P(y_1) \prod_{t=2}^{T} P(y_t | y_1^{t-1}) \tag{1}$$

which appears intractable in general. However, if we assume that the past sequence can be summarized by a *state variable* q_t, then one can rewrite the previous equation as

$$P(y_1^T) = \sum_{q_1}^{T} P(y_1^T, q_1^T) \tag{2}$$

where the sum over q_1^T represents the sum over all possible state sequence $q_1^T = q_1, q_2, ..q_T$ of length T. Fortunately now, this can be factored as follows

$$P(y_1^T, q_1^T) = P(q_1) \prod_{t=2}^{T} P(q_t|q_{t-1}) \prod_{t=1}^{T} P(y_t|q_t) \tag{3}$$

using the first order Markovian assumption (one state depends probabilisticly on just the preceding state) [23]. The joint probability is therefore completely specified in terms of *initial state probabilities* $P(q_1)$, *transition probabilities* $P(q_t|q_{t-1})$ and *emission probabilities* $P(y_t|q_t)$. Since each state variable q_t for the underlying Markov model is not directly observed but is a stochastic function of the previous observations y^{t-1} , such a model is called Hidden Markov Model. It can be trained to maximize the likelihood of a training set of sequences (Maximum Likelihood criterion), using well known algorithms such as Expectation Maximization (EM) [8], Viterbi [29], or gradient ascent [27].

For classification tasks, such as deciding if a given sequence belongs to a given target class, one usually trains a different HMM for each class to maximize the likelihood of the training sequences labeled for that class. Then, given a new sequence to classify, one usually computes the likelihood of the sequence for each HMM, and selects the class that maximizes the Maximum A Posteriori (MAP) criterion, $P(y_1^T|class = c)P(class = c)$, the likelihood of the sequence given an HMM weighted by the prior probability of the class.

We cite here some advantages of generative model for the purposes of both classification and density estimation problems:

- **Better inference algorithms:** Generative models and joint densities can be computed using reliable techniques for Maximum Likelihood or Maximum A Posteriori (MAP) estimation. These estimation techniques include the popular EM algorithm and typically outperform gradient ascent algorithms, which are the workhorses of some discriminative models (such as neural networks). The EM algorithm provably converges monotonically to a local maximum likelihood solution and often needs less parameter tuning than gradient ascent.

- **Modular learning:** In generative model, each class is learned individually and only considers the data whose labels correspond to it. The model does not focus upon inter-model discrimination and avoids considering the data as whole. Thus the learning is simplified and the algorithms proceed faster.

- **New classes:** It is possible to learn a new class (or retrain an old class) without updating the models of previous learned classes in generative models since each model is trained in isolation. In discriminative models, the whole system must be retrained since inter-model dynamics are significant.

- **Missing data:** Unlike discriminative models, a generative model is optimized over the whole dimensionality and thus models all the relationships between the variables in a more equal manner. Thus, if some of the data that was expected to be observed for a given task is missing, some joint model's performance (for ex. Bayesian network or generative models with assumption that all dimensions of data are statistically independent) will degrade gracefully. Discriminative models are trained for a particular task and thus a different model must be trained for missing data tasks.

- **Rejection of poor or corrupted data**: Sometimes, very poor data could be fed into the learning system and a generative model has the ability to detect this corrupt input (input which has low likelihood) and possibly signal the user to take some alternative measure.

2.2 Discriminative models and SVMs

Another approach to solving classification tasks, called discriminative learning, consists in the creation of a unique model trained with examples of all the classes and where the objective is to directly maximize the correct classification rate (which is not guaranteed by the Maximum Likelihood criterion). It allows a discriminative model to better learn the interactions between classes and their relative distributions for discrimination. We outline here some other advantages of this approach:

- **Data that need complex generative model for modeling can be separated easily by discriminative model :** It is often the case that data which need complex generative models can be easily separated by simple decision boundaries.
- **Resource management:** Discriminative models use resources exclusively for the classification task. Thus, they might be more appropriate in limited resources environments.

Examples of discriminative models are Logistic regression, Neural Networks [4], and Generalized Additive Models [12].

The problem here is that the training process for discriminative models is often cumbersome (i.e. neural network back propagation and gradient ascent) and somewhat ad hoc, requiring many re-initializations to converge to a good solution. Recently a new discriminative learning algorithm, the SVM [28] [5], has been proposed. The appeal of SVM is two-fold. Firstly the process of tuning the parameters in the training algorithm is simpler. Secondly they show great ability in generalization performance, not only on classification tasks but also on regression and density estimation problems. The key ideas in SVM are:

- In the case where data is linearly separable, the SVM simply looks for the separating hyperplane with the largest margin, with respect to the labeled training set

$$f^{max} = \arg\max_f \min_i \frac{y_i f(\mathbf{x}_i)}{||\mathbf{w}||}$$

$$\text{where } f(\mathbf{x}) = (\mathbf{x}\cdot\mathbf{w}) + b = \textstyle\sum_i \alpha_i y_i (\mathbf{x_i}\cdot\mathbf{x}) + b.$$
$$\mathbf{w} = \textstyle\sum_i \alpha_i y_i \mathbf{x}_i$$

for $\mathbf{x},\ \mathbf{w} \in \Re^N$, $b \in \Re$, $\alpha_i \in \Re$ is the contribution of the sample i in the final solution , $y_i \in \{-1, 1\}$ are the label corresponding to the training set $\{x_i\}$ and $sign(f(x))$ is the classification rule. α_i and b are determined in the training process. Intuitively this choice seems to be reasonable since a slightly perturbation in data does not affect the resulting classification. This is achieved by minimizing the square of l2-norm of $\mathbf{w}$

$$\frac{1}{2}||\mathbf{w}||^2 \tag{4}$$

subject to the inequalities

$$(\mathbf{x_i}\cdot\mathbf{w} + b)y_i \geq 1$$

for all i. The solution for the optimal hyperplane is a linear combination of a small subset of training set , $\mathbf{x_s}$, $s \in \{1, ..N\}$ known as support vectors. These support vectors satisfy the equality $(\mathbf{x_i}\cdot\mathbf{w} + b)y_i = 1$ This choice also follows Vapnik's *Structural Risk Minimization* principle [28].

- If an algorithm can be described merely by dot product operation, then in principle we can avoid the need to explicitly represent the acting vectors. This trick can be used for the case where data is not linearly separable. We first map data into a very high dimensional (might be infinite dimensional) space (feature space) which is more suitable for classification and then use the linear classifier for doing classification. Note that the only way data appears in the training problem, is in the form of dot product in that space. It can therefore be replaced by *kernels* such as Radial Basis Function (RBF) [5] that map the data into the feature space. The training algorithm's complexity will then depend only on the dimension of input space and the training set size, rather than the dimension of the feature space. Thus some parts of the " curse of dimensionality" (the fact that much more data is needed to ensure good generalization when mapping data to higher dimensional space) might be solved in this way.

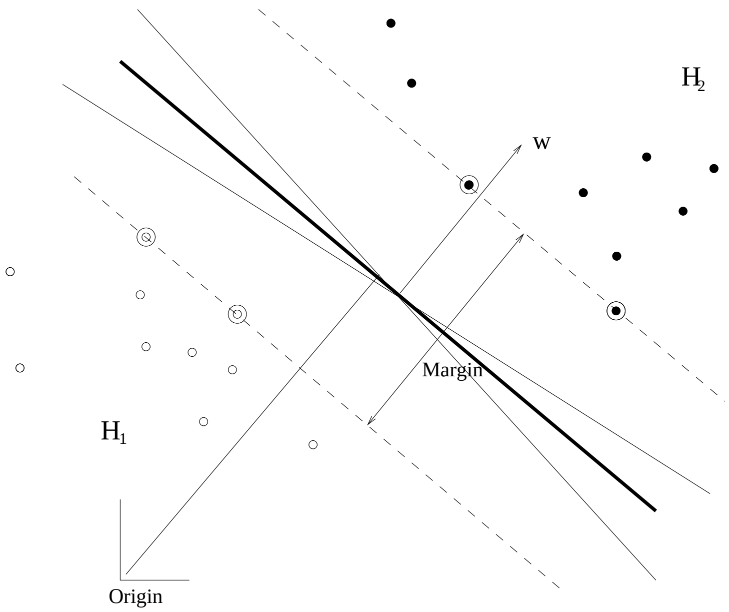

Figure 1: Among the separating hyperplanes, SVM choose the one which has largest margin. The support vectors are circled.

Another important feature of SVM is the soft margin which is applied when the sample is linearly inseparable even in the feature space. To overcome this problem positive slack variables ξ_i are introduced into the inequalities such that

$$\mathbf{x}_i \cdot \mathbf{w} + b \geq 1 - \xi_i \text{ for } y_i = +1$$
$$\mathbf{x}_i \cdot \mathbf{w} + b \geq -1 + \xi_i \text{ for } y_i = -1$$
$$\xi_i \geq 0 \ \forall i.$$

Then ξ_i must exceed unity for an error to occur and $\sum_i \xi_i$ is an upper bound on the number of training errors. A natural way for choosing the resulting problem to minimize is then

$$\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i \tag{5}$$

subject to

$$\mathbf{x}_i \cdot \mathbf{w} + b \geq 1 - \xi_i \text{ for } y_i = +1$$
$$\mathbf{x}_i \cdot \mathbf{w} + b \geq -1 + \xi_i \text{ for } y_i = -1$$
$$\xi_i \geq 0 \ \forall i$$

where C is a parameter to be chosen by the user, a larger C corresponding to assigning a higher penalty to errors.

The output of SVM is then a linear combination of the training examples projected in a high dimensional feature space through the use of kernels (functions that represents the dot product in the feature space)

$$y = sign(\sum_{i=1}^{l} y_i \alpha_i K(\mathbf{x} \cdot \mathbf{x}_i) + b) \tag{6}$$

where (x_i, y_i) are input/class from the training set (of size l), x is the current input and y is the desired class $\in \{-1, +1\}$, and $K(.,.)$ is a kernel function.

2.3 Combining generative and discriminative models

2.3.1 Combination of HMM and SVMs

Although having impressive performances for classical static problems it is not easy to apply SVM to sequence processing problems because of the variable sizes of the sequences. Recently, in some papers from Jaakkola et al [13], the authors have developed a general formalism for deriving kernel functions from generative probability models. The approach here is to derive the kernel function from generative models corresponding to the class of sequences of interest. More precisely, the kernel function specifies a similarity score for any pair of sequences whereas the likelihood score from generative models only measures the closeness of the sequence to the model itself. So generative models (HMMs) can assign the same likelihood to two totally different sequences. Suppose we use the well-known HMMs to construct a generative model for each class. Let that model be $P(\mathbf{X}|\theta)$, where parameter θ includes the emission and the transition probabilities of the trained HMM. The forward-backward algorithm can then be used to evaluate the likelihood of a sequence. In addition to obtaining the likelihood for the query sequence, the forward-backward algorithm also extracts the sufficient statistics for the parameters. For HMMs, they are the posterior frequencies of having taken a particular transition or having generated one of the residues of the query sequence $\mathbf{X}$ from a particular state. Now instead of computing the likelihood we modify the HMM to compute a fixed size vector that contains a value (sufficient statistics) for each independent parameter in the model. This vector should provide a summary of the sequence in the parameter space of model. By this way we can encode the descriptive power of generative models in a fixed size vector [13] which is an analogous quantity to the model's sufficient statistics. One such statistics is called the Fisher score:

$$U_\theta(\mathbf{X}) = \nabla_\theta \log(p(\mathbf{X}|\theta)). \tag{7}$$

Each component of U_X is the derivative of the log-likelihood of the sequence $\mathbf{X}$ with respect to a particular parameter. This is thus a very interesting way to turn a variable-length sequence problem into a static problem, which can be handled by advanced kernels methods. A natural kernel in this case is the inner product between these feature vectors, scaled by a positive definite matrix M :

$$K(\mathbf{X}_i, \mathbf{X}_j) = U_{\mathbf{X}_i} M^{-1} U_{\mathbf{X}_j} \tag{8}$$

where $M = I = E_{\mathbf{X}}(U_{\mathbf{X}} U_{\mathbf{X}}^T)$ is the Fisher Information matrix. Using Fisher score of all positive and negative sequences computed on an HMM model, we can train an SVM to decide the class of such vectors. The result is a discriminative model using information from generative models. It has been proved [13] that the resulting method gives performance that is asymptotically never inferior to the MAP method. Another choice is to set M as identity matrix if it is too difficult to compute I.

2.3.2 Ensemble of generative models

Machine learning algorithms work by searching in a hypothesis space H, which is supposed to contain an "acceptable" classifier , in order to find the most accurate one. Two very important aspects of the hypothesis space H are its size and whether it contains good approximations for the optimal classifier. Because of the difficulty in finding a single optimal classifier, another method for improving accuracy of a classifier is to use an ensemble of classifiers (a set of individually trained classifiers whose decisions are combined in some ways). Recent research [16] [9] has shown that an ensemble is often more accurate than any of the single classifiers in the ensemble. There are lots of appeals for using such method.

First of all, most learning algorithms work in a very large hypothesis space, then the training data might not provide sufficient information for choosing a single best classifier from H. After the training process there are still many hypotheses which have the same performance on the training data, but their generalization performance vary largely. Since there is no other information for choosing the best one among them, an ensemble of classifiers built from these hypotheses will ensure a good generalization performance.

The second reason for using ensemble methods is that the high complexity of the search problem might require heuristics in the learning algorithm. For example neural networks algorithms employ local search methods (such as gradient descent) to find a locally optimal set of weights for the networks. The resulting classifier is then a suboptimal hypothesis. Changing the parameters of the learning algorithms can lead to another suboptimal hypothesis. In this case ensemble methods can be used as a way of compensating for imperfect search algorithms.

Another reason is that the hypothesis space H might not contain a good enough classifier. Instead it might contain several weaker classifiers. Then combining these hypotheses can be used as a way to expand the searching space in which we can find a better classifier lying outside of H.

Our approach here is to train many classifiers using generative models from the training data. All outputs from these classifiers will be combined and used as inputs for training another discriminative model. If classifiers are chosen such that output errors are uncorrelated, this method will improve the accuracy of the classifier.

2.4 Speech and speaker recognition

Speech recognition is the most common and successful application of HMMs. A good survey of statistical methods for speech recognition can be found in [14]. The basic task can be stated as follows: given a sequence of acoustic vectors (usually obtained by preprocessing a speech signal, for instance spectral information represented by vectors of 10 to 40 dimensions sampled at a rate of around one centi-second per time frame), to find the corresponding sequence of words pronounced by the speaker. This is usually done by training a big HMM which embodies one sub-HMM per word or per phoneme and then using some optimization technique (such as Viterbi decoding [29]) find the best possible sequence of words corresponding to the acoustic sequence. Since every sequence of phonemes does not represent valid sequences of words, a language model is also used to constrain the search during decoding.

Speaker identification and verification are related tasks [6]: the identification task is to decide, given a sequence of acoustic descriptors, who is talking from a set of known voices. In the verification task, the client name is given together with the sequence of acoustic descriptors, and the task is then to decide if the speaker is really who he pretends to be or not.These two tasks are very similar and are usually solved by training one HMM per client and eventually one HMM for the *world* (which represents the *anti-client* in the verification task), and then using some kind of thresholding method and the MAP criterion to take a decision.

3 Research plan

There are various schemes for combining generative models and discriminative models, which can be categorized in one of these architecture: (i) parallel, (ii) serial combination and (iii) hierarchical. In the parallel architecture, individual models are invoked independently and their results are combined. As for the serial combination architecture, individual models are invoked in a linear sequence. And the hierarchical architecture is a mixture of the first twos. In the first stage, we plan to try different serial combination architectures with application to speech and speaker recognition. Different methods will be used for mapping variable size sequences to fixed size vectors, such that they can be used as inputs for discriminative models.

3.1 Combination using fixed size statistics computed from sequence data and generative model

Different kinds of fixed size statistics computed from the sequence data and their corresponding generative model can be used as input for discriminative models, for example the likelihood score and Fisher score.

The speaker verification is clearly a binary classification task. We will use state-of-the-art HMMs or Gaussian Mixture Models (GMMs) already proposed in the speaker verification community (as well as at IDIAP), but instead of using the MAP criterion and some special tricks to estimate the thresholds for deciding if a speaker is really who he claims to be, we propose to train a discriminative SVM with statistics (likelihood scores, Fisher scores,...) computed from both clients and impostors as inputs. The trained SVM is used for taking decision with statistics computed from the utterance of speaker as input. In speaker identification, since the name of the client is not given, the task is to decide to which client the given sequence correspond. It becomes a multi-class problem. Some special treatment will have to be done in order to apply the discriminative model (SVMs) in that case. Although the extension to multi-class problem is not trivial, some researchers have already proposed methods to solve multiclass problems with SVMs [17].

In speech recognition, the combining methods will not be so clear. One of the difficulties of speech recognition is the alignment problem: a training sequence usually corresponds to a speech signal sequence associated with a word or phoneme sequence. But no information is available regarding the exact alignment between these two sequences. So the task of speech recognition is more than a classification task: one has to learn not only the correct class (here the class is a sequence of words or phonemes) but also correct alignment. Moreover, there is also a decoding step where one selects the best sequence of phoneme using a constrained graph (some sub-sequences of phonemes are more probable than others, some are simply impossible, etc). Different solutions to this problem will be investigated. As often done in advanced speech recognition system, one possible approach that will be investigated is to rescore N-best word and phoneme sequence hypotheses, generated at the output of a standard HMM-based speech recognition system, and thus providing us with possible phonetic segmentation points. These phonetic segments can then be rescored using their fixed size statistics taking into account the whole segment.

In most modern speaker recognition tasks where the sequence pronounced by the speaker is composed by more than one word, one of the preprocessing tasks might be to try first to recognize what the speaker said and use the obtained alignment in the decoding and classification task. In that case, combination of discriminative models (SVMs) and generative models (HMMs, GMMs) could be tried at both leves, recognition and classification.

3.2 Combining ensemble of generative models and discriminative models

Another general method for improving accuracy is to use ensemble methods (2.3.2). In our approach, an ensemble of generative models will be generated for each class of sequence data. Their outputs (likelihood scores...) will then be combined and used as inputs for discriminative models. The ensemble methods can be applied in the generative-discriminative hybrid system at different levels.

At the first level, the set of input features of sequence data will be divided into different subsets, each feature subset being modeled by one generative model. This technique will work when the input features are highly redundant.

The learning process can also be changed to create ensembles of generative models of each class. A set of generative models with different capacities can be used for modeling the sequence data. In Gaussian Mixture Models for example, one can use ensembles of GMMs with different number of Gaussians. Different structures of generative models (GMMs, egordic HMM as in the speaker verification problem) used in modeling sequence data also lead to different generative models.

One of the major techniques for estimating generative models is the EM algorithm. But it is well-known that the EM algorithm only converges to a local maximum likelihood solution. By changing the initial points in the model space the algorithms will converge to different generative models. This can also be used to create ensembles of generative models, which also help solving the local optimum problem.

4 Preliminary experiment in speaker verification

4.1 Introduction

Speech carries information on several level. It includes linguistic message, speaker specific information, and information about the transmission environment... Speaker specific information contains the identity of the speaker, the gender of speaker, the idiosyncrasy and dialect of speaker, and even the emotional condition. Such information can be used in many applications, like access control, transaction authentication, or voice mail.The process of recognizing speaker's identity from the acoustic signal is then called speaker recognition. Broadly speaking, speaker recognition can be classified in two specific tasks: *speaker identification* and *speaker verification* [11].In *speaker identification* the task is to determine whom is talking from a set of known voices without identity claim from the speaker. In *speaker verification*, the task is to use acoustic signal to determine whether a person is who he claims to be.

Speaker recognition can also be divided into text dependent and text independent methods. The former requires the person to say the same text during the training and testing phase, whereas the latter does not constrain speaker on the content of his spoken text. Since text dependent methods can exploit speaker specific information associated with each phoneme or syllable, it generally achieves higher recognition performance than text-independent methods. But there are many applications where one can not predetermine the text being spoken, in which we have to use text-independent speaker verification methods.

In this experiment, we try to use a combination of Generative Gaussian Mixture Models (GMMs) and SVMs in the text-independent speaker verification task.

4.2 System outline

4.2.1 Traditional speaker verification system using generative models

The speaker verification problem can be considered as a statistical hypothesis testing problem where we test the hypothesis that the speaker is the true person that he claims to be (in which case, he is called a client) against the hypothesis that he is not (in which case he is called an impostor).

Given an utterance $\mathbf{X} = \{\mathbf{x}_1, .., \mathbf{x}_T\}$, we are interested in $P(S_i|\mathbf{X})$ the probability that speaker S_i has pronounced utterance $\mathbf{X}$. Using Bayes theorem, we can write it as follows:

$$P(S_i|\mathbf{X}) = \frac{p(\mathbf{X}|S_i)P(S_i)}{p(\mathbf{X})}. \qquad (9)$$

where $p(\mathbf{X}|S_i)$ is the likelihood that utterance $\mathbf{X}$ was generated by speaker S_i, $P(S_i)$ is the prior probability of speaker S_i and $p(\mathbf{X})$ is the likelihood of utterance $\mathbf{X}$.

Let us assume that $P(\overline{S_i}|\mathbf{X})$ is the probability that utterance $\mathbf{X}$ was pronounced by any other speaker. When $P(\overline{S}_i|\mathbf{X})$ is the same for all clients, we replace it by a speaker independent model $P(\Omega|\mathbf{X})$. Using Bayesian criterion, we then derive the decision rule:

$$\text{if } P(S_i|\mathbf{X}) > P(\Omega|\mathbf{X}) \text{ then } \mathbf{X} \text{ was generated by } S_i. \tag{10}$$

Using equation (9) , inequality (10) can be rewritten as:

$$Test(\mathbf{X}) = \frac{p(\mathbf{X}|S_i)}{p(\mathbf{X}|\Omega)} > \frac{P(\Omega)}{P(S_i)} = \delta_i. \tag{11}$$

Since it is more convenient to deal with *log-likelihood ratio statistic* rather than *likelihood ratio statistic*, taking logarithm of (11) leads us to inequality:

$$test(\mathbf{X}) = \log p(\mathbf{X}|S_i) - \log p(\mathbf{X}|\Omega) > \log \delta_i = \Delta_i. \tag{12}$$

With the assumption (most probably false) that $\mathbf{x}_1, \mathbf{x}_2..\mathbf{x}_T$ are conditionally independent and identically distributed, the log-likelihood ratio statistic becomes:

$$test(\mathbf{X}) = \sum_{t=1}^{T} (\log p(\mathbf{x}_t|S_i) - \log p(\mathbf{x}_t|\Omega)) \tag{13}$$

where $test(\mathbf{X})$'s are also independent, and identically distributed with

$$E(test(\mathbf{X})) = TE(test(\mathbf{x}))$$

$$Var(test(\mathbf{X})) = TVar(test(\mathbf{x}))$$

Normalizing by T will give us average log-likelihood ratio statistic which does not depend on the length of the utterance:

$$\frac{1}{T} test(\mathbf{X}) = \frac{1}{T} \sum_{t=1}^{T} (\log p(\mathbf{x_t}|S_i) - \log p(\mathbf{x_t}|\Omega)). \tag{14}$$

The system might have two types of errors: *false acceptance* (FA), when the system accepts an impostor, and *false rejection* (FR), when the system rejects a client. In order to be independent on the specific dataset distribution, the performance of the system is measured in terms of these two different errors, as follows:

$$FAR = \frac{\text{number of FAs}}{\text{number of impostor accesses}}, \tag{15}$$

$$FRR = \frac{\text{number of FRs}}{\text{number of client accesses}}. \tag{16}$$

An evaluation measure can be constructed as a combination of these two ratios, called the *decision cost function* (DCF):

$$DCF = Cost(FR) \cdot P(client) \cdot FRR + Cost(FA) \cdot P(impostor) \cdot FAR \tag{17}$$

where $P(Client)$ is the prior probability that a client will use the system, $P(impostor)$ is the prior probability that an impostor will use the system, $Cost(FR)$ and $Cost(FA)$ are the cost associated with a false rejection and false acceptance respectively.

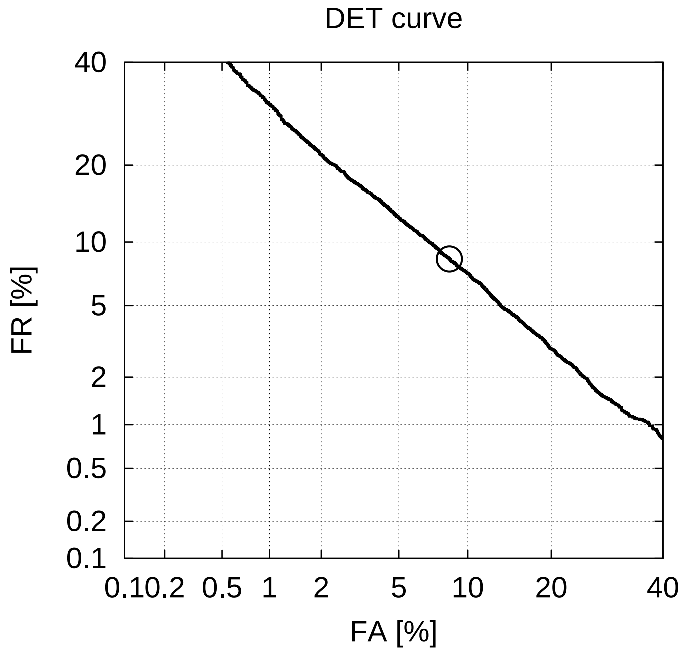

Figure 2: An example of a DET curve. The circle represents the threshold at equal error rate(EER), i.e. when FA=FR.

The tradeoff between FAR and FRR will depend on the application. In order to see the performance of a verification system with respect to this tradeoff, we can use the *detection error tradeoff* (DET)[19] plot (which represents FAR as a function of the FRR in logarithmic scale). An example of this curve is shown in figure 2

For this experiment, we chose a particular case of the DCF where the costs are equal to 1 and the probabilities are 0.5 each, which is known as *half total error rate* (HTER):

$$HTER = \frac{FAR + FRR}{2}. \tag{18}$$

A traditional speaker verification system can be built following three main steps [2]:

- **Preprocessing:** This step transforms the original speech waveforms into some specific feature vectors adapted to the training and decision steps.
- **Training:** During this step, one generative model(GMMs) is trained for the world model and then adapted for each client model.
- **Decision:** In this step, a threshold parameter Δ is chosen to optimize a criterion such as HTER.

4.2.2 Speaker verification system using hybrid GMM/SVM approach

Based on the idea discussed in section (2.3.1), instead of using the average log-likelihood ratio we will use the Fisher score computed from input data and generative models in order to train a discriminative model and take a decision. Our speaker verification system is then built following the traditional speaker verification system with some additional steps:

- **Preprocessing**
- **Dimensionality reduction:** Because the size of the Fisher score is equal to the number of parameter in the generative model, which is very large, the purpose of this step is to reduce the dimension of the acoustic vector, hence the number of coefficients in generative models.
- **Training generative model**
- **Computing Fisher score:** From the trained generative model, Fisher score for each acoustic sequence will be computed.
- **Training discriminative model:** The Fisher score is then used for training the discriminative model (SVM) in supervised mode.

In the next part of the paper, we will present the main components in the hybrid GMM/SVM speaker verification system as follows: section (4.3) describes the statistical (baseline) system, section (4.4) discusses the construction of a hybrid GMM/SVM classification system using Fisher scores, and the next section (4.5) mentions the dimensionality reduction problem. The results of some preliminary experiments are presented in section (4.6).

4.3 The baseline system

In the first stage of our system, a statistical generative model is chosen to serve as parametric basis for the Support Vector Machine and also as the baseline for performance evaluation.

For text-independent speaker verification, where there is no prior knowledge of what the speaker will say, the most successful statistical generative model has been Gaussian mixture models [25] [24]. Because of that reason and its other favorable properties [30], Gaussian mixtures have been chosen as the baseline system in our experiment.

The distribution of feature vectors extracted from a speaker's speech is then modeled by a Gaussian mixture density. Using i.i.d assumption, the likelihood of a sequence $\mathbf{X} = \{\mathbf{x}_1, .., \mathbf{x}_T\}$ given a GMM can be computed as follows:

$$p(\mathbf{X}|\theta) = \prod_{t=1}^{T} p(\mathbf{x}_t) = \prod_{t=1}^{T} \sum_{n=1}^{N} w_n \cdot \mathcal{N}(\mathbf{x}_t; \boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n) \tag{19}$$

where the parameter set of the GMM is $\theta = \{w_n, \boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n\}$ with $w_n \in \Re, \boldsymbol{\mu}_n \in \Re^d, \boldsymbol{\Sigma}_n \in \Re^{d^2}$ being respectively the prior probability, the mean vectors, and the covariance matrices of the n^{th} Gaussian component and d is the dimension of acoustic vectors:

$$\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n) = \frac{1}{(2\pi)^{\frac{d}{2}} \sqrt{|\boldsymbol{\Sigma}_n|}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_n)^T \boldsymbol{\Sigma}_n^{-1} (\mathbf{x} - \boldsymbol{\mu}_n)\right). \tag{20}$$

Usually diagonal covariance matrices are used in order to limit the model size, which implies the hypothesis that all features in acoustic vectors are uncorrelated.

From a large amount of speech data, maximum likelihood estimates of the speaker-independent model's parameters (also called the *world model*) are obtained using Expectation-Maximization algorithm [8]. Then, based on sequences of training vectors belonging to a particular speaker, the client model's parameters are trained via Bayesian adaptation technique from the world model GMM [1].

4.4 The Fisher kernel

From the approach discussed in the first part of the paper (section 2.3.1), we first map the input data to a (fixed dimension) feature space by computing Fisher scores from the client model and the world model:

$$U_{\theta(S_i)}(\mathbf{X}) = \nabla_{\theta(S_i)} \log(p(\mathbf{X}|\theta(S_i))) \tag{21}$$

$$U_{\theta(\Omega)}(\mathbf{X}) = \nabla_{\theta(\Omega)} \log(p(\mathbf{X}|\theta(\Omega))). \tag{22}$$

Theses scores are then concatenated to form the input of the discriminative model (SVM).

4.5 Parameter reduction

Since the size of the Fisher score is the number of parameters in the model, which is some tens of thousands (in diagonal covariance matrix GMM model, the number of parameters is: 2 $\times$ number of Gaussians $\times$ number of features in frame $\approx 2 \times 100 \times 30$) in state-of-the-art speaker verification systems, doing parameter reduction is an important step so that the computational complexity of the training algorithm for SVM becomes feasible. In this experiment parameter reduction is done in two stages: doing dimensionality reduction in input data space and parameter selection in generative models.

4.5.1 Dimensionality reduction

In general two approaches are available to perform dimensionality reduction:

- **Feature extraction:** in this approach, a subset of new features is created by combination of the existing features.
- **Feature selection:** here we reduce the dimensionality by choosing a subset of all features which are most informative.

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \\ x_D \end{bmatrix} \xrightarrow{\text{feature selection}} \begin{bmatrix} x_{i_1} \\ x_{i_2} \\ \\ x_{i_M} \end{bmatrix} \qquad \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \\ x_D \end{bmatrix} \xrightarrow{\text{feature extraction}} \begin{bmatrix} y_1 \\ y_2 \\ \\ y_M \end{bmatrix} = f\left(\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \\ x_D \end{bmatrix}\right)$$

Our first approach is using feature extraction methods, which reduce dimensionality by projecting the original D dimensional feature space onto a smaller subspace through a transformation. The optimum transformation $\mathbf{y} = f(\mathbf{x})$ will be the one that results in no increase in the *minimum probability of error* (when the probability of error is the same when a Bayes decision rule is applied on original space $\Re^D$ and in the reduced space $\Re^M$). Two commonly used linear feature extraction methods in multivariate statistic analysis are principal component analysis (PCA) and linear discriminant analysis (LDA). Whereas PCA method tries to represent the samples accurately in a lower-dimensional space, the goal of LDA method is to enhance the class-discriminatory information in the lower-dimensional space. Since our purpose is to keep as much as possible information in the high-dimensional space and doing PCA also helps decorrelate features in input space (which makes data fit better to the assuming diagonal covariance matrices GMMs), principal component analysis is chosen in this experiment.

In the method of principal components [15], dimensionality reduction is achieved by finding the orientation of a subspace which best preserves the information available in the original space and projecting the original space onto that subspace. The first principal component (PC) of a pattern vector is the projection of that sample onto the direction of largest variance as estimated over a training set. The rationale behind principal component analysis is the assumption that the direction of maximum variance contains most of the information about the various classes that the input pattern represents. Indeed, the PCA method is identical to the Karhunen-Loéve transformation (KLT) [10] used in speech coding. The resulting transformation is given by a $D \times M$ unitary matrix U whose columns are the eigenvectors corresponding to the M largest eigenvalues of the covariance matrix of the data.

Applying this method in our speaker verification system, we first compute the M largest principal component from the training data for the world model. Then all data for the world model as well as client models is transformed by projecting into these components.

4.5.2 Parameter selection

In this experiment, parameter selection is simply done by the selection of a set of important mixture components. The components of the world model are sorted according to their occurrence frequencies in the speaker training data. The occurrence frequencies are established by the weights of the components in the world model. Only the components corresponding to the highest weights are kept and their weights are normalized (Normalization is done by multiplying the weights of selected components by a constant such that the sum of these weights is equal to 1).

4.6 Results

All experiments described in this report were done using the publicly available SVMTorch toolkit [7] and the state-of-the-art GMM based speaker verification system developed at IDIAP [2] with some modifications to adapt systems to our purpose. The database for our experiments is the configuration 2 of the XM2VTS speech database [20] and its associated experimental protocol, the *Lausanne Protocol* [18]. The database contains four recording sessions of 295 subjects taken at one month intervals. On each session, one speech shot consisting of three sentences was made. The three sentences were the same for all speaker to allow the simulation of impostor accesses by all subjects. Sentences were chosen to compensate for prosodic and co-articulation effects. The database was divided into three sets: training set for building client models, evaluation set for computing the decision threshold and test set for estimating the performance of different verification algorithms.

During the preprocessing step, the speech signal was sampled every 10 ms and then parameterized into *Mel Frequency Cepstral Coefficients* (MFCC) frames [23], keeping 12 coefficients and their first derivative (also known as delta), as well as the energy together with its first derivative, for a total of 26 features computed every 10 ms. Then a *bi-Gaussian method* [2] is used for removing silence frames from data.

The world model (GMM) is then trained from the world data set (taken from another speaker verification database because of the size limit of the XM2VTS database) and adapted to client models using the training data set. For the baseline system, the HTER threshold is computed from log-likelihood ratio statistics of the evaluation data set and verified on the test data set. In the hybrid system, the input data for training and testing SVM is computed from the evaluation data set (including 40,000 impostor accesses and 400 client accesses) and the test data set (112,000 impostor accesses and 400 client accesses) respectively. Parameters of the SVM are chosen by cross-validation

4.6.1 Fisher score experiment

In initial experiments different dimensionality reduction methods were tried. At first, the PCA method was used for reducing the dimensionality of the data. Experiments with different number of principal components were done on two baseline systems (one with 100 Gaussians and one with 60 Gaussians in the GMM). The result is shown in figure 3.

Independently, the parameter selection method (4.5.2) was used for reducing the number of components in generative models. For each number of components to be selected in the parameter selection

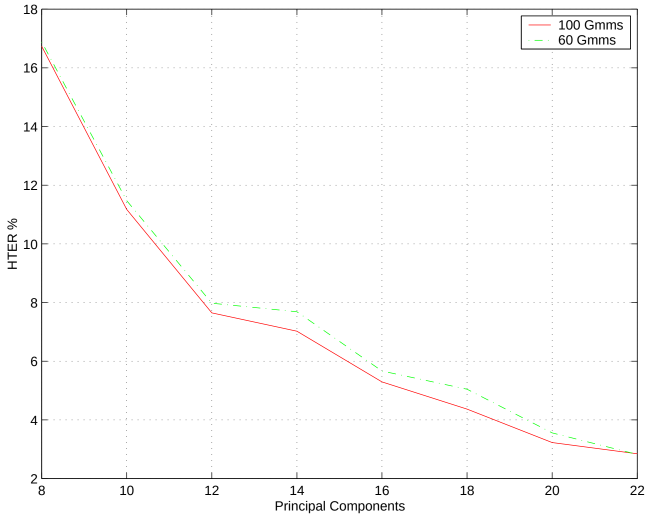

Figure 3: Plot of HTER using different number of principal components.

experiment, we also built a traditional speaker verification system with the same number of Gaussians for comparing purpose. The method was tested with different number of components to be selected (from the 100 Gaussians baseline system), as shown in table 1.

System	HTER(%)
100 Gaussians baseline	2.4
20 best Gaussians	5.459
20 Gaussians	4.06
40 best Gaussians	3.324
40 Gaussians	3.078
60 best Gaussians	2.247
60 Gaussians	2.596
80 best Gaussians	2.426
80 Gaussians	2.447

Table 1: Comparing results from the parameter selection experiments.

Our initial purpose was to use a state-of-the-art speaker verification system (with 100 Gaussians GMM) and then to do dimensionality reduction to reduce the number of components in the Fisher score. However applying dimensionality reduction methods with the 100 Gaussians baseline made the HTER increase dramatically. So we decided to use the 20 Gaussians baseline system for the Fisher score experiment instead. Noticing that the mean coefficients are more important in the generative model (only mean adaptation is applied in our system), we keep only derivatives of the means in the Fisher score. The result from this experiment (table 2) shows that the hybrid system still has not obtained competitive results compared to the performance of other state-of-the-art speaker verification systems.

4.6.2 A simple GMM/SVM hybrid system experiment

Because of the inferior performance of the hybrid GMM/SVM system in the previous experiment, another simple experiment was done for verifying the advantage in combining generative and discrim-

System	HTER(%)
20 Gaussians baseline	4.06
Hybrid GMM-SVM using Fisher score	8.864

Table 2: Fisher score experiment.

inative models. This approach has been mentioned in [3] in which instead of using log likelihood ratio (the Bayesian decision criterion) the log likelihood scores from the world model and the client model are used as input for training another discriminative model (SVM). From the experimental protocol we know that the impostors in the test data are different from the ones in the training data. A minor change was made in the cross-validation method to model the difference between the training population and the test population. In this database tailored cross-validation method, the training data were divided into the train data and the validation data in a way such that impostors in the validation data are different from impostors in the train data. This change has helped improve a little bit the performance of the hybrid system and the result is shown in table 3.

System	HTER (%)
The best system using Bayesian decision (with 400 Gaussians)	1.432
The hybrid GMM-SVM system using loglikelihood score	1.139

Table 3: Loglikelihood score experiment

5 Conclusion and future objectives

Although results from experiment in 4.6.2 shows that combining generative and discriminative models is a promising approach, the hybrid GMM/SVM system using Fisher score is still inferior to other state-of-the-art speaker verification systems. We mention here some problems which might be reasons of these results:

- The dimensionality reduction algorithm does not work well. By projecting client data to the principal components of the training data of the world model we might loose speaker specific information in this data.
- The distribution of our training data is extremely biased, the number of negative training samples is one hundred times higher than the number of positive training samples. It is even more biased in the test data. This makes the training of SVMs much harder.
- The optimization criterion in training algorithm of SVM is not the HTER but the number of classification errors.
- Because of the small size of the database, we had to use one universal discriminative models for all Fisher scores computed from different client models. This is somehow like comparing data from different spaces and might be an essential reason for the inferior performance of the system.

Our work in the following months will be to propose solutions to these problems, including:

- Finding more appropriate methods for dimensionality reduction and parameter selection.We might have to modify the PCA method or apply other nonlinear transformations (for example Kernel PCA [21] or Locally Linear Embedding [26]). These dimensionality reduction methods might be applied not only to the input data space but also to the Fisher score space.

- Changing the parameters of the training algorithm of SVM so that the optimization criterion becomes HTER.
- The GMM model (can be considered as HMM model with one node) does not keep information about the temporal evolution of speech data. A better HMM model might be used for modeling data.
- Trying other discriminative model, such as the Multilayer Perceptron (MLP).
- Finding other methods for extracting speaker specific information from speech data.
- Trying some kinds of speaker specific discriminative models.
- Trying other approach using ensemble methods in combining generative and discriminative models (3.2).

References

[1] S. Bengio and J. Mariéthoz. Comparison of client model adaptation schemes. IDIAP-RR 25, IDIAP, 2001.

[2] S. Bengio, J. Mariéthoz, and S. Marcel. Evaluation of biometric technology on XM2VTS. IDIAP-RR 21, IDIAP, 2001.

[3] Samy Bengio and Johny Mariéthoz. Learning the decision function for speaker verification. IDIAP-RR 25, IDIAP, 2001.

[4] C. Bishop. *Neural Networks for Pattern Recognition.* Clarendon Press, Oxford, 1995.

[5] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data mining and Knowledge Discovery*, 2(2):1–47, 1998.

[6] G. Chollet and F. Bimbot. Spoken language resources and assessment. In *Handbook of Standards and Resources for Spoken Language Systems*, volume 1. De Gruyter, Berlin, 1995.

[7] R. Collobert and S. Bengio. SVMTorch: Support vector machines for large-scale regression problems. *Journal of Machine Learning Research*, 1:143–160, 2001.

[8] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum-likelihood from incomplete data via the EM algorithm. *Journal of Royal Statistical Society B*, 39:1–38, 1977.

[9] Thomas G. Dietterich. Ensemble methods in machine learning. *Lecture Notes in Computer Science*, 1857, 2000.

[10] K. Fukunaga. *Introduction to Statistical Pattern Recognition, Second Edition.* Academic Press, Boston, MA, 1990.

[11] S. Furui. Recent advances in speaker recognition. *Lecture Notes in Computer Science*, 1206:237–252, 1997.

[12] T. J. Hastie and R. J. Tibshirani. *Generalized Additive Models*, volume 43 of *Monographs on Statistics and Applied Probability.* Chapman & Hall, 1990.

[13] T. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In M. S. Kearns, S. A. Sola, and D. A. Cohn, editors, *Advances in Neural Information Processing System 11: Proceedings of the 1998 Conference*, pages 487–493. MIT Press, 1999.

[14] F. Jelinek. *Statistical Methods for Speech Recognition.* MIT Press, Cambridge, Massachussetts, 1998.

[15] I. T. Jolliffe. *Principal Component Analysis.* Springer Verlag, NewYork, 1986.

[16] Josef Kittler, Mohamad Hatef, Rober P. W. Duin, and Jiri Matas. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):226–239, 1998.

[17] U. Kressel. Pairwise classification and support vector machines. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods: Support Vector Learning.* MIT Press, Cambridge, Massachussetts, 1999.

[18] J Lüttin. Evaluation protocol for the the XM2FDB database (lausanne protocol). Technical Report COM-05, IDIAP, 1998.

[19] A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki. The DET curve in assessment of detection task performance. In *Proceedings of Eurospeech'97, Rhodes, Greece*, pages 1895–1898, 1997.

[20] K Messer, J Matas, J Kittler, J Luettin, and G Maitre. Xm2vtsdb: The extended m2vts database. In *Second International Conference on Audio and Video-based Biometric Person Authentication*, March 1999.

[21] K. R. Muller, S. Mika, G. Ratsch, K. Tsuda, and B. Scholkopf. An introduction to kernel-based learning algorithms. *IEEETNN: IEEE Transactions on Neural Networks*, 12:181–201, 2001.

[22] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceeding of the IEEE*, 77(2):257–285, February 1989.

[23] Lawrence Rabiner and Biing-Hwang Juang. *Fundamentals of speech recognition.* Prentice Hall, first edition, 1993.

[24] Douglas A. Reynolds. Automatic speaker recognition using gaussian mixture speaker models. *The Lincoln Laboratory Journal*, 8(2):173–192, 1995.

[25] Douglas A. Reynolds, Thomas F. Quatieri, and Robert B. Dunn. Speaker verification using adapted gaussian mixture models. *Digital Signal Processing*, 10:19–41, 2000.

[26] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

[27] David E. Rumehart, G. E. Hinton, and R. J. Williams. Learning internal representation by error propagation. In David E. Rumelhart and James L. McClelland, editors, *Parallel Distributed Processing*, volume 1. MIT Press, Cambridge, MA., 1986.

[28] V. N. Vapnik. *The Nature of Statistical Learning Theory.* Springer-Verlag, New-York, NY, USA, 1995.

[29] A. Viterbi. Error bounds for convolutional code and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, pages 260–269, 1967.

[30] S. van Vuuren. *Speaker Verification in a Time-Feature Space.* PhD thesis, Oregon Graduate Institute, 1999.