Client Dependent GMM-SVM Models for Speaker Verification

Quan Le, Samy Bengio*

IDIAP, P.O. Box 592, CH-1920 Martigny, Switzerland
{quan,bengio}@idiap.ch

Abstract. Generative Gaussian Mixture Models (GMMs) are known to be the dominant approach for modeling speech sequences in text independent speaker verification applications because of their scalability, good performance and their ability in handling variable size sequences. On the other hand, because of their discriminative properties, models like Support Vector Machines (SVMs) usually yield better performance in static classification problems and can construct flexible decision boundaries. In this paper, we try to combine these two complementary models by using Support Vector Machines to postprocess scores obtained by the GMMs. A cross-validation method is also used in the baseline system to increase the number of client scores in the training phase, which enhances the results of the SVM models. Experiments carried out on the XM2VTS and PolyVar databases confirm the interest of this hybrid approach.

1 Introduction

Speaker verification techniques use acoustic signal to determine whether a person is who he claims to be or not. They have many applications, such as access control, transaction authentication or voice mail. A good introduction to the field can be found in [5].

State-of-the-art speaker verification systems are based on a generative model of speech sequences for each client, and another generative model for modeling impostors. Every time a client tries to access the system, the decision is taken using the ratio between the likelihood that the utterance was generated by the client model and the likelihood that the utterance was generated by the impostor model. For text independent speaker verification, where there is no prior knowledge about what the speaker will say, the most successful generative models have been Gaussian Mixture Models (GMMs) [8]. Such a system has several advantages. It provides a framework for handling variable length sequences, can be trained using reliable techniques like the Expectation Maximization algorithm (EM) [4], and is scalable with respect to the number of clients.

However, it is well known that for a classification problem, a better solution should in theory be to use a discriminative framework: in that case instead of

* The authors would like to thank the Swiss National Science Foundation for supporting this work through the National Center of Competence in Research (NCCR) on "Interactive Multimodal Information Management (IM2)".

constructing a model independently for each class, one constructs a unique model that decides where the boundaries between classes are.

In this paper we combine these two models using an idea mentioned in [1] in which instead of using the so-called log likelihood ratio criterion (the Bayesian decision) the GMM scores (scores from the world and client models as well as the log likelihood ratio or both) are used as input to train a discriminative model. Depending on the amount of data, we can have one discriminative model for all clients as in [1], or it can be extended to having one model for each speaker.

Based on the fact that in real world tasks it is not easy to collect lots of data from each client, a cross-validation technique is applied in order to increase the number of client scores used to train the discriminative model.

The rest of the paper is organized as follows. Section 2 presents the baseline speaker verification system using GMMs. Section 3 describes the hybrid system, including the combining method and the cross-validation technique to create more client accesses. Experiments on the XM2VTS and PolyVar databases are presented in Section 4. Finally, conclusions are drawn in Section 5.

2 The Baseline Speaker Verification System

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:

$$P(S_i|\mathbf{X}) = \frac{p(\mathbf{X}|S_i)P(S_i)}{p(\mathbf{X})} \tag{1}$$

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 unconditional 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:

$$\text{if } P(S_i|\mathbf{X}) > P(\Omega|\mathbf{X}) \text{ then } \mathbf{X} \text{ was generated by } S_i. \tag{2}$$

Using equation (1) , inequality (2) can be rewritten as:

$$\mathrm{Test}(\mathbf{X}) = \frac{p(\mathbf{X}|S_i)}{p(\mathbf{X}|\Omega)} > \frac{P(\Omega)}{P(S_i)} = \delta_i. \tag{3}$$

Since it is more convenient to deal with *log-likelihood ratio statistics* rather than *likelihood ratio statistics*, taking the logarithm of (3) leads us to inequality:

$$\mathrm{test}(\mathbf{X}) = \log p(\mathbf{X}|S_i) - \log p(\mathbf{X}|\Omega) > \log \delta_i = \Delta_i. \tag{4}$$

The distribution of feature vectors $\mathbf{x}_t$ extracted from a speaker's speech is often modeled by a Gaussian mixture density. Using the 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:

$$p(\mathbf{X}|\theta) = \prod_{t=1}^{T} p(\mathbf{x}_t|\theta) = \prod_{t=1}^{T} \sum_{n=1}^{N} w_n \cdot \mathcal{N}(\mathbf{x}_t; \boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n) \tag{5}$$

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

$$\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n) = \frac{1}{(2\pi)^{\frac{d}{2}}\sqrt{|\boldsymbol{\Sigma}_n|}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_n)^T \boldsymbol{\Sigma}_n^{-1}(\mathbf{x} - \boldsymbol{\mu}_n)\right). \tag{6}$$

In general, diagonal covariance matrices are used to limit the model size.

From a large amount of speech data, maximum likelihood estimates of $P(\mathbf{X}|\Omega)$, the world model, is obtained using the Expectation-Maximization algorithm [4]. Then, based on sequences of training vectors belonging to a particular speaker S_i, the client model $P(\mathbf{X}|S_i)$ is trained via a Bayesian adaptation technique from the world model [6, 8].

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 often measured in terms of these two different errors as follows:

$$\text{FAR} = \frac{\text{number of FAs}}{\text{number of impostor accesses}}, \tag{7}$$

$$\text{FRR} = \frac{\text{number of FRs}}{\text{number of client accesses}}. \tag{8}$$

Various evaluation measures can be constructed based on FAR and FRR. In this paper, we used the *Half Total Error Rate* (HTER):

$$\text{HTER} = \frac{FAR + FRR}{2}. \tag{9}$$

Moreover, in order to select a decision threshold (Δ_i), the system is often tuned on a validation set to optimize a criterion which could be different. For instance, the *Equal Error Rate* (EER), where FAR is equal to FRR, is often used.

3 Hybrid System

3.1 Support Vector Machines

Support Vector Machines (SVMs) [9, 2] are built upon two key ideas: *maximizing the margin* and the *kernel trick*.

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

$$f^{max} = \arg\max_f \min_i \frac{y_i f(\mathbf{x}_i)}{\|\mathbf{w}\|} \tag{10}$$

$$\text{where } f(\mathbf{x}) = (\mathbf{x} \cdot \mathbf{w}) + b = \sum_{i=1}^{l} \alpha_i y_i (\mathbf{x_i} \cdot \mathbf{x}) + b \tag{11}$$

$$\text{and } \mathbf{w} = \sum_{i=1}^{l} \alpha_i y_i \mathbf{x}_i \tag{12}$$

where l is the number of training examples, $\mathbf{x}, \mathbf{w} \in \mathbb{R}^N$, $b \in \mathbb{R}$, $\alpha_i \in \mathbb{R}$ is the contribution of sample i in the final solution , $y_i \in \{-1, 1\}$ are the label corresponding to the training set $\{\mathbf{x_i}\}$ and $sign\,(f(\mathbf{x}))$ is the classification rule. α_i and b are determined in the training process. This choice follows Vapnik's *Structural Risk Minimization* principle [9].

Since data only appears in the training problem in the form of dot products, 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 space (also called the feature space) which is more suitable for classification, and then use a linear classifier. The dot product operation in the data space can therefore be replaced by *kernels* such as Radial Basis Functions (RBF) [2]. The training algorithm's complexity will then depend only on the dimension of the input space and the training set size, rather than the dimension of the feature space.

The final output of an SVM is then a linear combination of the training examples projected in the feature space through the use of the kernel:

$$y = \text{sign}\left(\sum_{i=1}^{l} y_i \alpha_i K(\mathbf{x}, \mathbf{x}_i) + b\right) \tag{13}$$

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

3.2 Postprocessing GMM Scores by SVMs

In most speaker verification systems, the decision is taken using inequality (4) in which a universal threshold Δ is chosen to minimize the optimization criterion (HTER or EER for instance). It is equal to choosing a line from a family of parallel lines $\log p(\mathbf{X}|S_i) - \log p(\mathbf{X}|\Omega) = C$ which optimizes the criterion. This choice is the optimal solution when the distribution of data is perfectly estimated, which is usually not the case. Replacing this line by a more general and possibly non-linear function such as SVMs [1] might help to correct the scores from GMMs.

When the kernel chosen is the dot product, the discriminative model is the line which maximizes the *margin* between positive and negative examples. Since the log-likelihood ratio is known to contain important information (inequality (4)) which is not easy to recover when using SVMs with only scores from GMMs, it is put together with two log-likelihood scores from the world and client models as a 3-dimensional input vector for SVMs. "Client dependent" discriminative models were used in our experiments with the meaning that SVMs were trained and tested on the same population of clients to learn somehow their statistics[1]. If there is enough data for each client (enough client accesses), we can use speaker specific SVMs. It will better adapt discriminative models to each particular client[2], and also reduce the training time for the discriminative models since the complexity of the training algorithm for SVMs is quadratic on the number of examples.

One of the problems with SVMs is then the imbalance between the number of client and impostor accesses (with the XM2VTS database there are about 100 times more impostor accesses than client accesses). Here we use an empirical solution to increase the number of client scores based on a cross-validation method.

3.3 Using Cross-Validation to Increase the Number of Client Scores

It is generally not possible to collect a lot of speech data from each client, so it is important to use properly the collected data. In most speaker verification systems, the speech data set of each client is divided into two sets. The first part (called training client data set) is used to build the client model, while the second part is used to compute client scores for training the decision boundary for other clients or for measuring the performance of the system. We propose here to use cross-validation to obtain client scores from the training client data set. Firstly, the speech data set for each client is divided into N disjoint parts of equal size. For each part, a client model is built from the $(N-1)$ other parts, and off-training set scores are computed from the left-out part (Figure 1). This process is repeated for all N parts. The final client model is then trained using all data from the client training data set. When N is big enough the union from the $(N-1)$ data parts will be almost the same as the whole data set. So it is expected that the model obtained from these data will be similar to the final client model, and all off-training set scores computed from all N data parts in this way can be put together for training the decision model. By this way we will have scores from training client data set. These additional client scores can be used for various purposes, here we simply put them with other data for training a more accurate SVM model.

4 Experiments

4.1 The PolyVar Database

Database Description In the first experiment, we used the *PolyVar* telephone database [3] that contains two sets (called development population and evalua-

[1] For this approach, choosing whether or not to retrain the SVM for a new client is a trade-off between the accuracy and the use of computing resources.

[2] The speaker specific discriminative model approach is scalable with respect to the number of clients. For each new client, one only needs to train a new SVM for him.

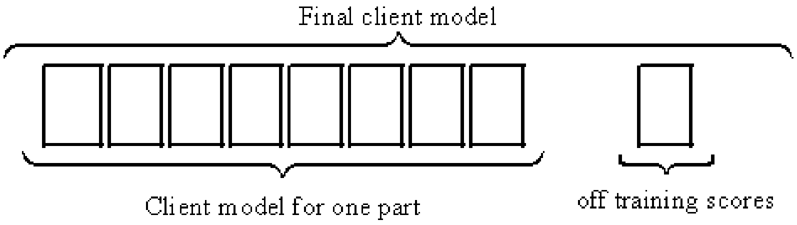

Fig. 1. Illustration of the cross-validation idea on one part of training client data set.

tion population) of 19 clients as well as another population of 56 speakers (28 men and 28 women) used for the world model. The database contains different numbers of recording sessions for each speaker, where one recording session contains 17 words. For each client 20 sessions were used, 10 of them (called training client model data set) for adapting the world model to the client model and the other 10 sessions for test only.

Results from Text Independent Experiments The speech signal was sampled every 10 ms and then parameterized into 16 *Linear Frequency Cepstral Coefficients* (LFCC) coefficients as well as energy, complemented by their first derivatives, for a total 34 coefficients[3].

To determine the number of Gaussians for the world model, we used a simple validation technique, training on 90% of the available training set for the world model and selecting the model with the highest likelihood over the remaining 10%. This led us to a GMM with 256 Gaussians. The client models were adapted from the world model using the training client data set (10 sessions) of each client. For the baseline system a global threshold is estimated based on scores from the test set of the development population and the performance of the system is estimated on the scores of the test set of the evaluation population. In the hybrid system, the cross-validation technique is applied to create scores (called training scores) from the training client data, in which training client data set is divided into 10 parts (according to 10 sessions). All other experiments were done on the system using the cross-validation technique. In the first experiment, an universal decision threshold Δ is estimated on the cross-validation system using the HTER criterion (as in the baseline system). Then one universal linear SVM is trained from the training scores (with 17860 impostor accesses and 3230 client accesses) of the evaluation population (for comparing with the baseline system), and the performance is measured on the test data (the evaluation population consisting of 3230 client accesses and 13262 impostor accesses). In the last experiments, we estimated one decision threshold per client, and correspondingly

[3] For all experiments with the XM2VTS and PolyVar databases, a voice activity detection module was used to discard silence frames.

estimated one linear SVM per client (trained by 170 client accesses and 940 impostor accesses).

Table 1 gives the results of these experiments on the test set of the evaluation population. We can see that the universal linear SVM obtains better results than the baseline system, and the speaker specific linear SVMs system yields the best performance.

System	HTER(%)
256 Gaussians baseline	5.770
Global threshold on Cross-validation system	5.765
Universal linear SVM on Cross-validation system	5.320
Threshold per client on Cross-validation system	5.710
Linear SVM per client on Cross-validation system	5.146

Table 1. Results from the PolyVar database.

4.2 Experiments on the XM2VTS Database

Database Description In a second series of experiments, we used the XM2VTS database [7] and its associated experimental protocol, the Lausanne protocol. The database contains four recording sessions of 295 subjects taken at one month intervals. In each session, one speech shot consisting of two sentences was made. The sentences were the same for all speakers 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.

Results During the preprocessing step, the speech signal was sampled every 10 ms and then parameterized into LFCC coefficients, keeping 16 coefficients and their first derivative, as well as the energy together with its first derivative, for a total of 34 features.

The world model (GMM with 600 Gaussians, the number of Gaussians was chosen using the same technique as described above) was then trained from the world data set (taken from another speaker verification database because of the limited amount of data in the XM2VTS database) and adapted to client models. In the baseline system using the configuration 2 of the XM2VTS database, two sessions per client (each session has two sentences) were used for training client models, one session for estimating decision threshold, and one for measuring performance of the system. Using the cross-validation idea, we merged the client data from the training set and the evaluation set into one training client model data set consisting 3 sessions. This data set is divided into 6 parts (according to 6 sentences in 3 sessions) and the cross-validation technique is used to create client scores. The resulting client model is better estimated (trained by 3 sessions) and we also have more client scores in the enrollment phase. In the first experiment, to test the cross-validation system we simply compute the decision threshold as in the baseline system and measure the performance. In the second experiment, one linear SVM is trained from the training scores (including 40,000 impostor accesses and 1,200 client accesses), and the performance is then measured on the test data (112,000 impostor accesses and 400 client accesses). Because there are only six utterances for one client in the enrollment phase, we did not have enough data to obtain a separate threshold per client (with or without SVM postprocessing).

Results from Table 2 show that the cross-validation system got better result than the baseline system, and using SVM further improved the result. In fact, to the best of our knowledge, the result obtained here is the best ever reported on that subset.

System	HTER(%)
600 Gaussians baseline	1.155
Global threshold on Cross-validation system	1.060
Universal linear SVM on Cross-validation system	0.92

Table 2. Results from the XM2VTS database.

5 Conclusions

In this paper we proposed the use of a cross-validation technique to increase the number of client scores used to select the decision function for speaker verification systems. These scores are then used to train SVM models for taking the decision, instead of using the classical thresholding method. Results from experiments on the XM2VTS and PolyVar databases show that the hybrid generative-discriminative model is a promising approach.

References

1. S. Bengio and J. Mariéthoz. Learning the decision function for speaker verification. In *IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing ICASSP*, 2001.
2. C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data mining and Knowledge Discovery*, 2(2):1–47, 1998.
3. G. Chollet, J.-L. Cochard, A. Constantinescu, C. Jaboulet, and P. Langlais. Swiss french polyphone and polyvar: telephone speech databases to model inter- and intra-speaker variability. IDIAP-RR 1, IDIAP, 1996.
4. A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum-likelihood from incomplete data via the EM algorithm. *Jrnl. of Royal Statistical Society B*, 39:1–38, 1977.
5. S. Furui. Recent advances in speaker recognition. *Lecture Notes in Computer Science*, 1206:237–252, 1997.
6. J. Mariéthoz and S. Bengio. A comparative study of adaptation methods for speaker verification. In *Intl. Conf. on Spoken Language Processing ICSLP*, 2002.
7. 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 AVBPA*, March 1999.
8. D. A. Reynolds, T. F. Quatieri, and R. B. Dunn. Speaker verification using adapted gaussian mixture models. *Digital Signal Processing*, 10:19–41, 2000.
9. V. N. Vapnik. *The Nature of Statistical Learning Theory.* Springer-Verlag, New-York, NY, USA, 1995.