3. Feature Selection for Speaker Recognition

Often, in pattern recognition problems, there is large number of features that may be used. Usually, it is not possible to work with a very large dimensional feature space, since recognition errors may increase when using larger feature vectors ("the curse of dimensionality" [Jain et al., 2000]). In addition, application forces constraints in memory and computation power, which limit the dimension of the feature space.

Feature selection is the process of selecting a features subset, which is most effective for preserving class separability. The problem of feature selection can be described as follows: Given a set Y of K features $Y=\{y_i \mid i=1,2,...,K\}$ select a subset X (of $k<K$ features) $X=\{x_i \mid i=1,2,...,k,\ x_i \in Y\}$ such that the performance criterion $J(\bullet)$ is optimized.

A feature selection method can be specified in terms of two components:

1) *Selection procedure*,

2) *Performance criterion*, $J(\bullet)$

In speaker verification/identification tasks, the aim of this selection is to determine the feature space of size $k<K$ for which the recognition error is minimized. Minimizing the recognition error is not always easy to implement; hence separability measures are often introduced as criteria.

3.1. Selection Procedure

Several selection procedures are discussed in the pattern recognition literature [Fukunaga, 1990] [Jain et al., 2000]. On the following, we describe some of the most commonly used procedures.

3.1.1. Exhaustive search

Exhaustive search is an optimal method for selecting a subset of k best features among the entire set K. It considers all the combinations of k out of K. Implementation of such a search requires an enormous amount of computation, namely $\binom{K}{k} = \frac{K!}{k!(K-k)!}$ searches. For example, with $k = 24$ and $K = 120$, the number of searches is $\sim 10.872 \times 10^{24}$ (!). This is obviously cost prohibitive. There is, therefore, a need for some more effective procedures.

3.1.2. K-best Method

This method is probably the simplest. The best subset of k features is composed of the k best features considered one at a time. However, a set of the best **individual** k features is not necessarily the best set of k features.

3.1.3. Forward Selection

This method is sometimes called "*bottom-up*" [Jain et al., 1997], "*ascendant selection*" [Charlet et al., 1997], or "*add-on*" [O'Shaughnessy, 1986]. The *forward selection* procedure starts with the empty set and adds features iteratively. Initial tests are done with each of K features, one at a time, to select the best single feature. Then, tests with two features, including the best one selected at the previous stage, and each (one at a time) of the remaining $K - 1$ features. The cycle is repeated until the desired number of features has been chosen. The number of searches in this forward selection is $\frac{1}{2}k(2K-k+1)$. For the previous example, with $k = 24$ and $K = 120$, the number of searches is 2604, which is much less than the exhaustive search.

Both *k-best* and *forward selection* procedures are simple search techniques, which avoid exhaustive search enumeration. However, the selection of the optimal subset is not guaranteed.

3.1.4. Backward Selection

This method is a simple stepwise search technique, sometimes called the "*knock-out*" strategy [Sambur, 1975] or "*top-down*" [Jain et al., 1997]. The *backward selection* procedure starts from the full set of K features. All K subsets of $K-1$ features are used in the performance criterion calculation to determine the best subset (of $K-1$ features). The feature not used in this best subset is "knock-out" of consideration. The process is repeated with $K-1$ subsets of $K-2$ features, etc.

3.1.5. The l-r Algorithm

The *l-r* algorithm [Pandit et al., 1998] uses the forward and the backward selection procedures in order to yield a better performance selection procedure. For every iteration, the algorithm uses the forward procedure to add l features, and the backward procedure to remove the r worst features from the augmented subset.

3.1.6. The Sequential Floating Forward Sequence (SFFS)

The Sequential Floating Forward Sequence (SFFS) [Pudil et al., 1994] can be viewed as a "dynamic" l-r algorithm. It consists of applying, after each forward step, a number of backward steps as long as the resulting subsets are better than the previously ones evaluated at that level. Consequently, there are no backward steps at all if the performance cannot be improved. The SFFS method can be described algorithmically as follows:

Initialization:

$X_0 = \varnothing; \quad k = 0$

Termination:

Stop when k equals the number of features required

Step 1

$x^+ = \arg\max_{x \in Y - X_k} J(X_k + x)$

$X_{k+1} = X_k + x^+; \quad k = k + 1$

Step 2

$x^- = \arg\max_{x \in X_k} J(X_k - x)$

if $J(X_k - x^-) > J(X_{k-1})$ then

$X_{k-1} = X_k - x^-; \quad k = k - 1$

go to step 2

else

go to step 1

where X_k is the feature subset in the kth step, $J(\bullet)$ is the criterion, and Y is the full feature set.

3.1.7. Branch-and-Bound (BB)

The *Branch-and-Bound (BB)* feature selection algorithm can be used to find the optimal subset of features much more quickly than the exhaustive search method [Fukunaga, 1990]. One drawback is that the BB procedure requires that the feature selection criterion function be monotone (non-decreasing).

3.1.8. Dynamic Programming (DP)

Dynamic programming is utilized to find an optimal set of features using much fewer calculations than *exhaustive search*. Dynamic programming is a multistage optimization technique that exploits the principle of optimality which states: whatever the initial state and decision are, the remaining decisions must constitute an optimal policy with regard to the

state resulting from the first decision. When applied to the selection of features, this principle in conjunction with a functional equation permits the choice of attributes that have the maximum effectiveness [Cheung et al., 1978]. One may view the dynamic programming procedure as a tree search method as shown in figure 3.1.

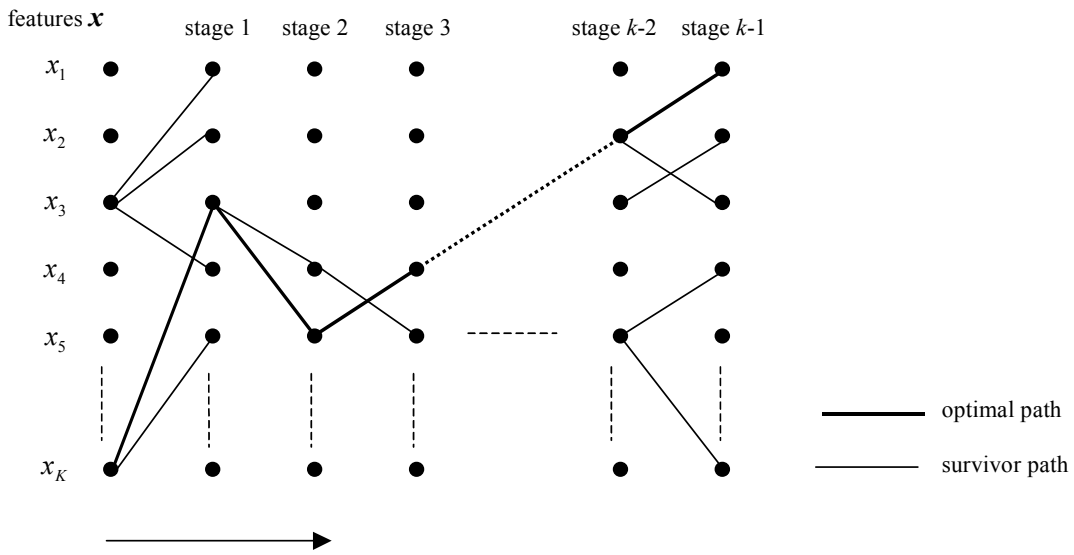

Figure 3.1: Feature subset selection using dynamic programming.

In this representation, the features x_j ($j = 1,2,...,K$) are depicted by the nodes of the tree. Subsets can be interpreted as paths or branches joining the nodes of subsequent stages. There are *k* stages in this iterative algorithm, as the number of features in the optimal subset. Let $\mathbf{q}_n^j = \left(q_1^j, q_2^j, ..., q_n^j\right)$ ($j = 1,2,...,K$) be one of the *K* possible subsets selected after *n* stages where q_n^j represents a feature in *X*. For every x_j ($j = 1,2,...,K$) at the *n*th stage, the subset $\mathbf{q}_n^j$ is chosen such that:

$$J\left(\mathbf{q}_n^j\right) = \max_i J\left(\mathbf{q}_{n-1}^i, x_j\right);\ i = 1,2,...,K;\ x_j \notin \mathbf{q}_{n-1}^i \qquad (3.1)$$

where J is defined as a feature performance criterion. For a detailed discussion of the DP algorithm, see [Cheung et al., 1978].

To guarantee optimal results, the performance criterion, J, has to be a monotonic, non-decreasing function of n and can be separated into two parts, one corresponding to the history of the process up to the n-1 stage and the other corresponding to the behavior of the process at the nth stage [Nemhauser, 1966]. Most of the criteria used in practice cannot guarantee these characteristics, especially when the features are dependent. In this case the DP is a sub-optimal selection method.

3.1.9. Genetic Algorithms (GA)

In a *Genetic algorithm* (GA) approach, a given feature subset is represented as a binary string (a "chromosome") of length K, with zero or one in position i denoting the absence or presence of feature i in the set [Jain et al., 1997]. A population of chromosome is maintained. Each chromosome is evaluated to determine its "fitness," which determines how likely the chromosome is to survive and breed into the next generation. New chromosomes are created from old chromosomes by one of two processes: (1) crossover, where parts of two different parent chromosomes are mixed to create offspring; and (2) mutation, where the bits of a single parent are randomly perturbed to create a child.

3.2. Performance Criteria

In speaker recognition systems, a meaningful performance criterion is the speaker recognition error over some evaluating data. This measure can be determined experimentally by employing the attributes in the recognition experiment and tallying up the mistakes made. However, implementation of the criterion requires a great amount of computation, especially for HMM/GMM-based classifiers. Moreover, the resolution of such recognition error criterion, using relatively small amount of a given training/evaluating data, is not enough. An alternative is to exploit the statistical properties of the features and derive information on the probability of error from the talker's PDF.

3.2.1. F-ratio

One common measure of effectiveness for scalar features (a single feature) is the *F-ratio* [Sambur, 1975]. The F-ratio compares inter- and intra-speaker variances

$$F = \frac{\text{variance of inter - speaker feature mean}}{\text{mean of intra - speaker feature variance}} \tag{3.2}$$

The numerator is large when values for the speaker-averaged feature are widely spread for different speakers, and the denominator is small when feature values in utterance repetitions by the same speaker vary little (the denominator averages intra-speaker variances over all speakers). High F-ratios are desirable. However, F-ratio measures the features individually, and a set of the best **individual** k features, are not necessarily the best set of k features. A criterion is necessary to decide whether a feature set A is better than a feature set B.

In *discriminant analysis* of statistics, there are two types of criteria which are frequently used in practice. One is based on a family of functions of scatter matrices. The criteria measure the class separability of N classes, but do not relate to the Bayes error directly. The

other is a family of criteria which gives upper bounds of the Bayes error. The Bhattacharyya distance is one of these criteria. However, the criteria only apply for two-class problems, and are based on a normality assumption [Fukunaga, 1990].

3.2.2. Scatter Matrices and Separability Criteria

In discriminant analysis, within-class, between-class, and mixture scatter matrices are used to formulate criteria of class separability [Fukunaga, 1990]. Given N classes $(\omega_i; i = 1,2,\ldots,N)$, and $\boldsymbol{\mu}_i$, the ith class expected vector, defined by

$$\boldsymbol{\mu}_i = E(\mathbf{o} \mid \omega_i) \tag{3.3}$$

where $\mathbf{o}$ is the features vector. The ith class covariance matrix, $\mathbf{W}_i$, defined by

$$\mathbf{W}_i = E\left\{(\mathbf{o}-\boldsymbol{\mu}_i)(\mathbf{o}-\boldsymbol{\mu}_i)^T \mid \omega_i\right\} \tag{3.4}$$

The averaged *within-class scatter matrix*, $\mathbf{S}_\mathbf{w}$, shows the scatter of samples around their respective class expected vectors, and is expressed by

$$\mathbf{S}_\mathbf{w} = \sum_{i=1}^{N} P(\omega_i) E\left\{(\mathbf{o}-\boldsymbol{\mu}_i)(\mathbf{o}-\boldsymbol{\mu}_i)^T \mid \omega_i\right\} = \sum_{i=1}^{N} P(\omega_i)\mathbf{W}_i \tag{3.5}$$

where $P(\omega_i)$ is the ith class a-priori probability. On the other hand, a *between-class scatter matrix*, $\mathbf{S}_\mathbf{b}$, is the scatter of the expected vectors around the mixture mean

$$\mathbf{S}_\mathbf{b} = \sum_{i=1}^{N} P(\omega_i)(\boldsymbol{\mu}_i - \boldsymbol{\mu}_0)(\boldsymbol{\mu}_i - \boldsymbol{\mu}_0)^T \tag{3.6}$$

where $\boldsymbol{\mu}_0$ represents the expected vector of the mixture distribution and is given by

$$\boldsymbol{\mu}_0 = E\{\mathbf{o}\} = \sum_{i=1}^{N} P(\omega_i)\boldsymbol{\mu}_i \tag{3.7}$$

The *mixture scatter matrix* is the covariance matrix of all samples regardless of their class assignments, and is defined by

$$\mathbf{S}_{\mathbf{m}} = E\left\{(\mathbf{o}-\boldsymbol{\mu}_{\mathbf{0}})(\mathbf{o}-\boldsymbol{\mu}_{\mathbf{0}})^{T}\right\} = \mathbf{S}_{\mathbf{w}} + \mathbf{S}_{\mathbf{b}} \tag{3.8}$$

The scatter matrices are designed to be invariant under coordinate shifts.

In order to formulate criteria for class separability, these matrices have to be converted to a scalar. This scalar should be larger when the between-class scatter is larger or the within-class is smaller. Typical criteria are the following:

(1) $J_1 = \mathrm{tr}\left(\mathbf{S}_2^{-1}\mathbf{S}_1\right)$, (3.9)

(2) $J_2 = \ln\left|\mathbf{S}_2^{-1}\mathbf{S}_1\right|$, (3.10)

(3) $J_3 = \dfrac{\mathrm{tr}\,\mathbf{S}_1}{\mathrm{tr}\,\mathbf{S}_2}$, (3.11)

where $\mathbf{S}_1$ and $\mathbf{S}_2$ are one of $\mathbf{S}_{\mathbf{b}}, \mathbf{S}_{\mathbf{w}}$, or $\mathbf{S}_{\mathbf{m}}$.

3.2.3. Bhattacharyya Distance

Bhattacharyya distance is a convenient measure of class separability if the number of classes is two. The Bhattacharyya distance of the normal distributions of class ω_i and class ω_j, also referred to as $\boldsymbol{\mu}(1/2)$ [Fukunaga, 1990], is

$$d_B = \frac{1}{2}\ln\frac{\left|\dfrac{\mathbf{W}_i+\mathbf{W}_j}{2}\right|}{\sqrt{|\mathbf{W}_i||\mathbf{W}_j|}} + \frac{1}{8}(\boldsymbol{\mu}_i-\boldsymbol{\mu}_j)^{T}\left(\frac{\mathbf{W}_i+\mathbf{W}_j}{2}\right)^{-1}(\boldsymbol{\mu}_i-\boldsymbol{\mu}_j) \tag{3.12}$$

where $\boldsymbol{\mu}_i$ is the *i*th class expected vector, and $\mathbf{W}_i$ is the *i*th class covariance matrix.

Furthermore, (3.12) gives an upper bound of the Bayes error (where the distributions are normal). Neglecting scaling, the second term is the *Mahalanobis* distance using an average covariance matrix.

3.2.4. Bhattacharyya Shape

The Bhattacharyya distance (3.12) is the sum of two components, one based solely upon the covariance matrices and the other involving differences between the mean vectors. These components can be characterized, respectively, as an average shape and the difference in size of the PDFs [Campbell, 1997]. This shape component, the *Bhattacharyya shape* is defined as

$$d_{Bs} = \ln \frac{\left| \frac{\mathbf{W}_i + \mathbf{W}_j}{2} \right|}{\sqrt{|\mathbf{W}_i||\mathbf{W}_j|}} \tag{3.13}$$

3.2.5. Divergence Distance

Divergence is another criterion of class separability, similar to the Bhattacharyya distance. It is defined as

$$d_D = E\left\{ -\ln \frac{p(\mathbf{o}\,|\,\omega_i)}{p(\mathbf{o}\,|\,\omega_j)} \,|\, \omega_j \right\} - E\left\{ -\ln \frac{p(\mathbf{o}\,|\,\omega_i)}{p(\mathbf{o}\,|\,\omega_j)} \,|\, \omega_i \right\} \tag{3.14}$$

where $p(\mathbf{o}\,|\,\omega_i)$ is the class conditional density function. When two density functions are normal, the divergence becomes [Campbell, 1997],

$$d_D = \frac{1}{2}\mathrm{tr}\left[(\mathbf{W}_i - \mathbf{W}_j)(\mathbf{W}_j^{-1} - \mathbf{W}_i^{-1}) \right] + \frac{1}{2}\mathrm{tr}\left[(\mathbf{W}_i^{-1} + \mathbf{W}_j^{-1})(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)^T \right] \tag{3.15}$$

3.2.6. Divergence Shape

The Divergence (3.15) is the sum of two components, one based solely upon differences between the covariance matrices and the other involving differences between the mean vectors. Like the Bhachattaryya distance, these components can be characterized, respectively, as differences in shape and size of the PDFs [Campbell, 1997]. This shape component, the *divergence shape*, defined as,

$$d_{Ds} = \mathrm{tr}\left[\left(\mathbf{W}_i - \mathbf{W}_j\right)\left(\mathbf{W}_j^{-1} - \mathbf{W}_i^{-1}\right)\right] \tag{3.16}$$

3.3. Speaker Recognition with Feature Selection - State of the Art

Most of the published speaker recognition systems do not use a feature selection process. It is often assumed that features used successfully in speech recognition tasks, will also perform best in speaker recognition tasks. Thus, in most reported speaker recognition systems, the feature space used is that of cepstral and delta cepstral (MFCC). Most of the literature on speech feature selection deals with speech recognition systems [Bocchieri et al., 1993] [Biem et al., 1993] [Sharma et al., 2000]. However, features that have been selected for speech recognition systems are not necessarily the best selection for speaker recognition systems.

The specific problem of feature selection for speaker recognition has been published in several papers. Some of them use common feature space for all the speakers, few use individual feature space, and some take group of features and test them without the use of selection algorithm. The next paragraphs describe some published papers on feature selection for speaker recognition systems.

[Cheung et al., 1978] used *dynamic programming* for text-independent speaker identification. In this work the *divergence* criterion was chosen. The database consisted of reading parts by ten male speakers. A set of 32 initial features was determined from the input speech: the pitch value, log energy, ten PARCOR coefficients, ten cepstral coefficients, normalized absolute prediction error energy, and nine normalized autocorrelation coefficients. Each feature was averaged over some input text. The dynamic programming procedure was implemented to select the subset of ten ($k = 10$) out of the 32 features that had the maximum divergence. This selected feature subset was utilized in the linear classifier

for identifying talkers text-independently. The identification results of the dynamic programming feature subset (~11% identification error) were superior to the results that were achieved by PARCOR coefficients (~21%), cepstral features (~25%), and features selected by the *knock-out* strategy [Sambur, 1975] (~12%). It was concluded also that not much improvement in identification error is gained for $k > 7$.

Another paper proposes a framework for feature selection in an HMM-based text-dependent speaker verification system published by [Charlet et al., 1997]. These authors developed a performance criterion (called *Score*) for feature selection from the HMM emission probability function. Assuming the HMM emission probability function is Gaussian, and the covariance matrices of the features are diagonal. The performance criterion of a feature subset was defined as

$$Score(\mathbf{O}) = \sum_{i \in X} Score_i(\mathbf{O}) = \sum_{i \in X} \sum_{\tau=1}^{T} \left(\log\left(\sqrt{2\pi}\sigma_i(\tau)\right) + \frac{\left(o_i(\tau) - \mu_i(\tau)\right)^2}{2\sigma_i(\tau)^2} \right) \qquad (3.17)$$

where $o_i(\tau), \mu_i(\tau)$, and $\sigma_i(\tau)$ are the *i*th feature value, mean, and variance, respectively, at time frame τ (from *T* frames*)*. *X* is the tested feature subset. Four selection procedures were implemented: k-best method, forward, backward, and dynamic programming. The proposed framework was applied to study cepstral coefficients and their first and second derivatives. Experiments were conducted on a large-scale telephone database of 55 speakers, and a distinct database of 130 imposters. For each target, training was performed with three repetitions of a password phrase collected during a single call. The four selection procedures were tested, and the k-best method was worse than others. The other three procedures yielded equivalent performances. In their experiences, the optimal set contained 14 features (from an overall set of 27 features). It was found that cepstral coefficients of high order and

first derivatives of all cepstral coefficients were the most useful for speaker verification. Performance comparison was made between the verification system with the selected 14-feature set and the initial 27-feature set. It was also found that the selected feature set gives an EER of 4.6%, compared to 5.9% with the initial set of 27 features. Moreover, it was shown that when a great deal of training data was available, the optimized feature set contained much more features than the one obtained with little (three repetitions) training data.

[Campbell, 1997] evaluated the effectiveness of LP-based features and information theoretic measures by a simple speaker recognition system. The features included: LARs, LSP frequencies, and LP cepstra. The measures evaluated included divergence shape, Bhattacharyya shape, Bhattacharyya distance, divergence measure, Mahalanobis distance, and Euclidean distance. The decision criterion was to choose the closest speaker according to the selected feature and measure. The LSP frequencies were found to be the most effective features using the divergence-shape measure. A speaker-identification test yielded 98.9% correct closed-set speaker identification, using cooperative speakers with high-quality telephone-bandwidth speech.

[Cohen et al., 1989] investigated whether speakers can be optimally recognized in an **individual feature space**. They suggested using a quadratic classifier for text independent speaker identification. Individual features were selected by a dynamic programming procedure. The criterion for feature selection was chosen to be the divergence, $D_i(k)$, that was defined as the mean distance (in the *k* dimensional feature space) of the *i*th class from all other classes, namely:

$$D_i(k) = \frac{1}{N-1}\sum_{j=1}^{N}\left(\boldsymbol{\mu}_j - \boldsymbol{\mu}_i\right)^T \left(\mathbf{W}_i\right)^{-1}\left(\boldsymbol{\mu}_j - \boldsymbol{\mu}_i\right) \tag{3.18}$$

where $\boldsymbol{\mu}_i$ and $\mathbf{W}_i$ are the expected feature vector and the covariance matrix of the *i*th speaker, respectively, and N is the number of speakers. The problem of optimal feature selection was defined as follows: given a set of K feature, find a subset of $k < K$ features so that the $D_i(k)$ criterion is maximized. The system consisted of two main "libraries" representing the N speakers. The *general* library stored the complete set of $K = 77$ mean features and covariance matrices of all N speakers. The features used by the system included: normalized autocorrelation coefficients (model order 10), linear prediction coefficients, PARCORs, cepstral coefficients, prediction error, mean frame energy, and mean pitch frequency. The *active* library stored a reduced ($k < K$) set of individual feature vectors and covariance matrices. All features were calculated separately for voiced and unvoiced segments. The quadratic classifier identification scheme was compared with the linear classifier (with non-individual features) using an identification experiment with six male speakers. The training data contained about six minutes of Hebrew speech from each speaker, using non-overlapping 15-second segments to extract the mean features. The test data consisted of about two minutes of text, using overlapping 15-second segments for the extraction of test features. Feature vectors of order $k = 10$ were used. Although individual covariance matrices are not easily and accurately estimated, this work demonstrated that the quadratic classifier with individual optimal feature spaces is superior to conventional linear classifier speaker identification. Moreover, the authors showed that when each speaker is represented in his own optimal space, unvoiced utterances have almost equal importance to the voiced segments, in contrary to the common assumption.

[Haydar et al., 1998] experiments further supported the idea of a speaker's individual feature space. The authors introduced a genetic algorithm to reduce a 24-feature (12 LPC cepstra + 12 delta-cepstra coefficients) set to a 5-10 feature set, for each speaker in a text-

independent speaker identification system. For each speaker, not only may the features selected be different, but also the space dimensionality could vary. The speech signal was taken from 15 male speakers (selected from the SPIDRE database), and was segmented into 22.5 msec non-overlapping frames. For the speaker identification system, a very simple statistical model was used: two Gaussian distributions for each speaker. The experiment results showed that an increase in recognition rate of ~5% was achieved when the feature selection was made. Moreover, it was shown that cepstral parameters 3-4-5 were selected more frequently than the others. Delta-cepstrals, when compared with cepstrals, seemed to be less important on average.

Individual Feature selection for a DTW-based text-dependent speaker verification system was reported by [Pandit et al., 1998]. The feature selection technique was based on the l-r algorithm, and was applied to study LPC-cepstrums and their first order orthogonal polynomial coefficients. Experimental results on French database (0-9 digits, 33 speakers) showed that an optimum feature set could be obtained without degrading the performance of the system. While experiments using Spanish database (0-9 digits, 40 speakers) showed improvement of verification error rate: FA of 6% for 20 features vs. FA of 3.87% for 10 features (while FR is 0%).

3.4. Proposed Performance Criterion for Speaker Verification

In verification systems, the decision to accept or reject an identity claim is based on the comparison of a score, $s(\mathbf{O})$, with a threshold, τ:

$$s(\mathbf{O})\begin{cases}\geq \tau & \rightarrow accept \\ < \tau & \rightarrow reject\end{cases} \tag{3.19}$$

The simplest score for stochastic model-based verification systems is the log likelihood, which is the log probability of the (utterances) observations, **O**, given the target's (claimed speaker) model, λ_T:

$$s(\mathbf{O}) = \log p(\mathbf{O} \mid \lambda_T). \tag{3.20}$$

Systems based on a heuristic threshold decision (3.19) were shown to be very sensitive to the value of τ, to the text, the noise and other parameters of the system. It has been shown that normalized scores are preferred over the un-normalized scores. Several normalization methods have been suggested, for example, the log-likelihood ratio between the target's model and a background model [Reynolds, 1995]:

$$s_n(\mathbf{O}) = \log\left(\frac{p(\mathbf{O} \mid \lambda_T)}{p(\mathbf{O} \mid \lambda_B)}\right) = \log p(\mathbf{O} \mid \lambda_T) - \log p(\mathbf{O} \mid \lambda_B) \tag{3.21}$$

where λ_B is the background model. Sometimes, cohort based normalized scores are used [Zigel and Cohen, 2003]. Note that the normalization may be seen as a dynamic threshold, namely the threshold depends on the test utterance (the text) and on the background model.

A common measure for testing performances of speaker verification systems is the Equal Error Rate (EER), namely the case when the false accept error is equal to the false reject (miss) error. It is therefore logical to use the EER (or some function thereof), as the performance criterion.

The estimation of the EER requires large computation loads. The feature selection procedures requires the estimation of the criterion at each stage. Moreover, **given a relatively small amount of training data available, the EER has very low resolution as a criterion**. Hence, the use of EER as a criterion becomes impractical.

The performance criterion for feature selection we propose, is an estimation of a function of the EER, based on the assumption that the scores' Probability Density Function (PDF) of the target $\left(f\left[s(\mathbf{O})|\mathbf{O}\in\mathbf{O}_T\right]\right)$ and imposters $\left(f\left[s(\mathbf{O})|\mathbf{O}\in\mathbf{O}_I\right]\right)$, are both Gaussians. Here $\mathbf{O}_T$ and $\mathbf{O}_I$ are the observations uttered by the target and imposters respectively. Figure 3.2 schematically describes the estimation of the EER.

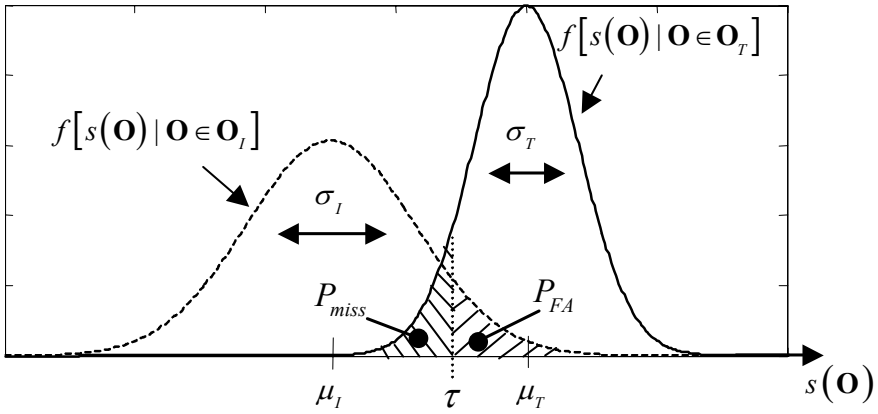

Figure 3.2: Estimation of verification errors from target and impostors Gaussian-like histograms.

The PDF of the target's score is assumed to be Gaussian:

$$f\left[s(\mathbf{O})|\mathbf{O}\in\mathbf{O}_T\right]=\frac{1}{\sqrt{2\pi}\sigma_T}\exp\left[-\frac{\left(s(\mathbf{O})-\mu_T\right)^2}{2\sigma_T^2}\right] \tag{3.22}$$

and the PDF of the impostors is similarly assumed to be Gaussian with parameters (μ_I, σ_I). Given a threshold, τ, the False Accept (P_{FA}) and False Reject (or "miss" P_{miss}) errors may be calculated by the areas under the appropriate curves as shown in figure 3.2.

To verify that the PDF is indeed Gaussian, a χ^2 goodness-of-fit test was successfully performed on the targets' scores as well as on the impostors' scores.

3.4.1. Gaussian Goodness of Fit Test

To perform χ^2 goodness of fit test to show that $s(\mathbf{O}) \sim N(\mu, \sigma)$, we first divide the range of $s(\mathbf{O})$ into *m* subintervals [Huang et al., 2001], such that the expected number of values, E_i, in each interval is at least 5. The actual number of points ($s(\mathbf{O})$ values) in the i^{th} subinterval is denoted by N_i. It can be proven [Mood et al, 1974] that the following random variable λ

$$\lambda = \sum_{i=1}^{m} \frac{(N_i - E_i)^2}{E_i} \tag{3.23}$$

converges to the χ^2 distribution with $m-k-1$ degrees of freedom as the sample size $n \to \infty$, where *k* is the number of parameters that must be estimated from the sample data in order to calculate the expected number of values, E_i. To make a decision, whether $s(\mathbf{O}) \sim N(\mu, \sigma)$ or not, we need to find the critical point, *c*, and compare it to λ (3.23). The critical point is calculated using:

$$P(\lambda > c) = 1 - F_{\chi^2}(x = c) = \alpha_0 \tag{3.24}$$

where $F_{\chi^2}(x)$ is the distribution function for χ^2 distribution, and α_0 is the pre-determined level of significance. The test procedure simply rejects $s(\mathbf{O}) \sim N(\mu,\sigma)$ when the realized value λ (3.23) is such that $\lambda > c$.

Using $m = 10$ subintervals, a Gaussian goodness-of-fit test was performed for targets' scores as well as for impostors' scores. The left column in table 3.1 shows the subintervals edges, were $\mu = E[s(\mathbf{O})]$ and $\sigma = std[s(\mathbf{O})]$. Table 3.1 shows also the corresponding probability falling in each subinterval, the expected number of points falling in each subinterval and the actual number of points falling in each subinterval, for one example target's scores (speaker #3) and impostors' scores from 39 speakers.

Table 3.1: the corresponding probability falling in each subinterval, the expected number of points falling in each subinterval and the actual number of points falling in each subinterval, for one example target's scores (speaker #3) and impostors' scores from 39 speakers.

Subinterval, I_i	$P(s(\mathbf{O}) \in I_i)$	$E_i = nP(s(\mathbf{O}) \in I_i)$		N_i	
		Target $(n = 306)$	Impostors $(n = 657)$	Target	Impostors
$[-\infty, -1.6\sigma + \mu]$	0.0548	16.7686	36.0031	21	28
$[-1.6\sigma + \mu, -1.2\sigma + \mu]$	0.0603	18.4427	39.5976	19	55
$[-1.2\sigma + \mu, -0.8\sigma + \mu]$	0.0968	29.6164	63.5882	25	76
$[-0.8\sigma + \mu, -0.4\sigma + \mu]$	0.1327	40.6132	87.1989	36	82
$[-0.4\sigma + \mu, 0.0\sigma + \mu]$	0.1554	47.5591	102.1121	42	94
$[0.0\sigma + \mu, 0.4\sigma + \mu]$	0.1554	47.5591	102.1121	47	89
$[0.4\sigma + \mu, 0.8\sigma + \mu]$	0.1327	40.6132	87.1989	47	86
$[0.8\sigma + \mu, 1.2\sigma + \mu]$	0.0968	29.6164	63.5882	34	64
$[1.2\sigma + \mu, 1.6\sigma + \mu]$	0.0603	18.4427	39.5976	21	41
$[1.6\sigma + \mu, \infty]$	0.0548	16.7686	36.0031	14	42

For the target's scores, the value for λ can be calculated as follows:

$$\lambda_T = \sum_{i=1}^{m} \frac{\left(N_i - E_i\right)^2}{E_i} = 5.4494 \tag{3.25}$$

Since λ can be approximated as a χ^2 distribution with $m-k-1=10-0-1=9$ degrees of freedom, the critical point c at the $\alpha_0 = 0.05$ level of significance is calculated, using cumulative distribution function table [Mood et al, 1974], to be 16.919, according to equation (3.24). Thus we should accept the hypothesis $s(\mathbf{O}) \sim N(\mu,\sigma)$, for target's scores, because the calculated λ_T is less than the critical point c. The same conclusion is also for the impostors' scores.

$$\lambda_I = \sum_{i=1}^{m} \frac{\left(N_i - E_i\right)^2}{E_i} = 13.89 < 16.919 = c \tag{3.26}$$

Similar results were achieved for other tested target speakers and for test-independent scores as well.

Figure 3.3 shows an example of a target's score histogram and its impostors' score histogram, with the best-fitted Gaussians. The scores were calculated in the target's selected feature space (24 features).

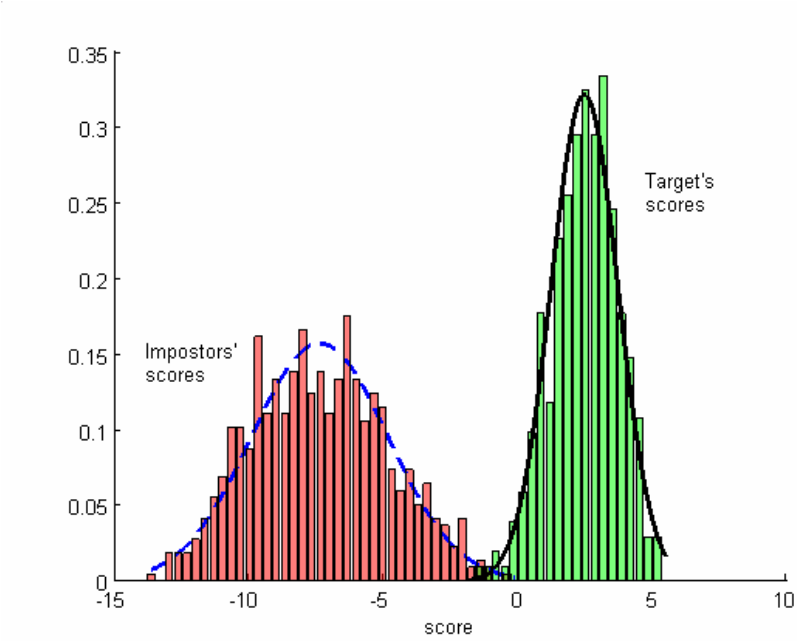

Figure 3.3: Gaussian fit for the histogram of target (#3) and impostors' scores.

3.4.2. The Recognition Related Criterion (RRC) [Zigel and Cohen 2004]

Under the Gaussian assumption, the false reject, or P_{miss} errors and the false accept P_{FA} may be written as

$$P_{miss} = \int_{-\infty}^{\tau} f\left[s(\mathbf{O}) \mid \mathbf{O} \in \mathbf{O}_T\right] ds = \int_{-\infty}^{\tau} \frac{1}{\sqrt{2\pi}\sigma_T} \exp\left[-\frac{1}{2}\left(\frac{s-\mu_T}{\sigma_T}\right)^2\right] ds = \text{erf}\left(\frac{\tau-\mu_T}{\sigma_T}\right) + \frac{1}{2} \tag{3.27}$$

$$P_{FA} = \int_{\tau}^{\infty} f\left[s(\mathbf{O}) \mid \mathbf{O} \in \mathbf{O}_I\right] ds = \int_{\tau}^{\infty} \frac{1}{\sqrt{2\pi}\sigma_I} \exp\left[-\frac{1}{2}\left(\frac{s-\mu_I}{\sigma_I}\right)^2\right] ds = -\text{erf}\left(\frac{\tau-\mu_I}{\sigma_I}\right) + \frac{1}{2} \tag{3.28}$$

where τ is the threshold for which $P_{miss} = P_{FA}$ $(= EER)$ (figure 3.2), and:

$$\mathrm{erf}(x)=\frac{1}{\sqrt{2\pi}}\int_0^x \exp\left[-\frac{1}{2}t^2\right]dt$$

$$\mu_T = E\{s(\mathbf{O})\,|\,\mathbf{O}\in\mathbf{O}_T\},\ \mu_I = E\{s(\mathbf{O})\,|\,\mathbf{O}\in\mathbf{O}_I\}$$
$$\sigma_T = \mathrm{std}\{s(\mathbf{O})\,|\,\mathbf{O}\in\mathbf{O}_T\},\ \sigma_I = \mathrm{std}\{s(\mathbf{O})\,|\,\mathbf{O}\in\mathbf{O}_I\}$$

Let us first find the value of τ for which $P_{miss}=P_{FA}$:

$$P_{miss}=P_{FA} \Rightarrow \mathrm{erf}\left(\frac{\tau-\mu_T}{\sigma_T}\right)+1/2=-\mathrm{erf}\left(\frac{\tau-\mu_I}{\sigma_I}\right)+1/2$$
$$\Rightarrow \mathrm{erf}\left(\frac{\tau-\mu_T}{\sigma_T}\right)=\mathrm{erf}\left(-\frac{\tau-\mu_I}{\sigma_I}\right)$$

Since $\mathrm{erf}(\bullet)$ is a monotonically injected (one to one) function, the last equation yields

$$\frac{\tau-\mu_T}{\sigma_T}=-\frac{\tau-\mu_I}{\sigma_I}$$

hence, the value of τ for which $P_{miss}=P_{FA}$ is given by:

$$\tau=\frac{\mu_I\sigma_T+\mu_T\sigma_I}{\sigma_I+\sigma_T} \tag{3.29}$$

By introducing the value of τ (3.29) in the $P_{miss}\ (=EER)$ equation (3.27)

$$EER=\mathrm{erf}\left(\frac{\mu_I-\mu_T}{\sigma_I+\sigma_T}\right)+1/2 \tag{3.30}$$

Since we are interested in minimizing EER, the constant ½ is irrelevant. Moreover, since erf(·) is a monotonically injected function, its argument may be used as a criterion. Thus, the proposed performance criterion, RRC, is

$$RRC=\frac{\mu_T-\mu_I}{\sigma_I+\sigma_T} \tag{3.31}$$

The criterion of equation (3.31) is to be maximized. Note that this criterion is somewhat similar to the F-ratio for the two Gaussian curves [Cohen and Zigel, 2002].

3.4.3. Generalized Performance Criterion

In many cases speaker verification application calls for some ratio of P_{miss} to P_{FA} (other than the EER). For example, in high-security systems, false accept errors are much less desirable than false reject errors, while in "convenience" applications, false reject errors are much less desirable.

Let C be the required ratio between the false reject error, P_{miss}, and the false accept error P_{FA}:

$$C = \frac{P_{miss}}{P_{FA}} \quad ; \quad 0 < C < \infty \tag{3.32}$$

Let us define the generalized error E_C

$$E_C = \rho P_{miss} + (1-\rho) P_{FA} \quad ; \quad 0 \le \rho \le 1 \tag{3.33}$$

where ρ is some weighting coefficient.

We want to minimize E_C with the given constraint (3.32). Introducing (3.32) in (3.33):

$$E_C = P_{miss}\left(\rho + \frac{1-\rho}{C}\right) = P_{FA}\left(1 + \rho(C-1)\right) \tag{3.34}$$

Since the parenthesis part in the last equation is constant, minimizing E_C is equivalent to minimizing P_{miss}, or minimizing P_{FA}, with the given constant.

Let us find the value of τ which fulfills equation (3.32). Introducing (3.27) and (3.28) into (3.32):

$$\mathrm{erf}\left(\frac{\tau-\mu_T}{\sigma_T}\right) + 1/2 = C\left[-\mathrm{erf}\left(\frac{\tau-\mu_I}{\sigma_I}\right) + 1/2\right] \tag{3.35}$$

There is no analytic solution to equation (3.31) (unless $C=1$). A numerical search technique may be used to find the value of τ. Let us denote this value, τ^c.

Introducing the value of τ^c into equation (3.27):

$$P_{miss} = \text{erf}\left(\frac{\tau^c - \mu_T}{\sigma_T}\right) + \frac{1}{2} \tag{3.36}$$

Since we are interested in minimizing P_{miss}, as before, we may use here the (minus) argument of the $\text{erf}(\bullet)$ as a (maximization) criterion:

$$P_{miss}^c = \frac{\mu_T - \tau^c}{\sigma_T} \tag{3.37}$$

This criterion is somewhat more complicated to implement than the *RRC* criterion. Equation (3.37) requires a search technique in order to find the value of τ^c. The search technique may be somewhat simplified by the fact that the search range is bounded by (using equation (3.29)):

$$\tau > \frac{\mu_I\sigma_T + \mu_T\sigma_I}{\sigma_I + \sigma_T} \quad ; \quad \text{for } C > 1$$
$$\tau < \frac{\mu_I\sigma_T + \mu_T\sigma_I}{\sigma_I + \sigma_T} \quad ; \quad \text{for } C < 1$$

The generalized performance criterion described above, has not been evaluated in this work due to lack of time.