Pattern Recognition Using a Family of Design Algorithms Based Upon the Generalized Probabilistic Descent Method

SHIGERU KATAGIRI, SENIOR MEMBER, IEEE, BIING-HWANG JUANG, FELLOW, IEEE, AND CHIN-HUI LEE, FELLOW, IEEE

Invited Paper

This paper provides a comprehensive introduction to a novel approach to pattern recognition, which is based on the generalized probabilistic descent method (GPD) and its related design algorithms. The paper contains a survey of recent recognizer design techniques, the formulation of GPD, the concept of minimum classification error learning that is closely related to the GPD formalization, a relational analysis between GPD and other important design methods, and various embodiments of GPD-based design, including segmental-GPD, minimum spotting error training, discriminative utterance verification, and discriminative feature extraction. GPD development has its origins in basic pattern recognition and Bayes decision theory. It represents a simple but careful re-investigation of the classical theory and successfully leads to an innovative framework. For clarity of presentation, detailed discussions about its embodiments are provided for examples of speech pattern recognition tasks that use a distance-based classifier. Experimental results in speech pattern recognition tasks clearly demonstrate the remarkable utility of the family of GPD-based design algorithms.

Keywords— *Bayes decision theory, discriminant function approach, discriminative feature extraction, discriminative training, generalized probabilistic descent method, minimum classification error learning, pattern recognition, speech recognition.*

I. INTRODUCTION

Pattern recognition has long been a topic of fundamental importance in a wide range of science and technology. In these days of rapidly growing information-oriented societies, improvement in its performance is an urgent technological issue. In particular, a mathematically proven, effective, and efficient method is desired for designing highly accurate recognizers. As one of the solutions for meeting this need, a discriminative training method called the generalized probabilistic descent method (GPD) was developed for classifier design [54]. This method has been shown to be very useful in various speech pattern classification tasks. Since its development it has been deeply analyzed and further extended to a more general methodological framework for pattern recognition (the terminological difference between "classification" and "recognition" will be shown later). This paper is therefore devoted to providing a comprehensive review of the GPD-based approach to pattern recognition.

GPD is a general pattern recognition framework. For clarity of presentation, we consider here the problem of speech pattern recognition, which is one of the crucial research areas in the development of multimedia and artificial intelligence technologies. In the following paragraphs of this section, we shall summarize the motivations for GPD's development, addressing problems in speech recognizer design.

A. Speech Pattern Recognition Using Modular Systems

We refer to the acoustic output of the human speech production system as a speech instantiation, and it can be considered as a sequence of linguistic units, such as phonemes and words. The goal of speech recognition then is to map a speech instantiation to its corresponding correct sequence of linguistic units.

As can be easily observed, the duration of each linguistic unit is highly variable, mainly due to speaking-rate changes. This indicates that a speech instantiation is a dynamic (variable-durational) temporal sample. In addition, the acoustic properties of speech waves are highly variable due to various factors such as the speakers themselves and the speaking fluency. The size of a vocabulary, i.e., the number of words used, often exceeds several tens of thousands of words. From this, it is obvious that coping with various issues appropriately and comprehensively is an

Manuscript received November 1, 1997; revised April 17, 1998.

S. Katagiri is with ATR Human Information Processing Research Laboratories, Kyoto 619-02 Japan (e-mail: katagiri@hip.atr.co.jp).

B.-H. Juang and C.-H. Lee are with Bell Laboratories, Lucent Technologies, Murray Hill, NJ 07974-0636 USA (e-mail: bjuang@lucent.com; chl@research.bell-labs.com).

Publisher Item Identifier S 0018-9219(98)07864-5.

indispensable requirement in the design of the recognizer. However, it is not necessarily recommended to start introductory discussions with such a large-scale complicated design framework. Let us therefore start preparations on discussions by using the following basic statement: speech recognition involves a process of mapping a dynamic speech instantiation, which is *a priori* correctly extracted from its surrounding acoustic signal and belongs to one of a given set of M speech classes, to a class index; C_j $(j = 1, \ldots, M)$. We specially consider the design problem of training the adjustable parameter set Ψ of a recognizer (see below for a more precise description), aiming at achieving the optimal (i.e., best in recognition accuracy for all future instantiations) recognition decision performance.

One of the fundamental approaches to this problem is the Bayes approach using the following Bayes decision rule, the rigorous execution of which is well known to lead to the minimum recognition error rate:

$$C\left(u_1^{T_0}\right) = C_i, \quad \text{iff } i = \arg\max_j \, p\left(C_j | u_1^{T_0}\right) \tag{1}$$

where $u_1^{T_0}$ is a dynamic speech instantiation with length $T_o, C(\cdot)$ represents the recognition operation, and it is assumed that the *a posteriori* probability for the dynamic instantiation $p(C_j|u_1^{T_0})$ exists and is known. A training goal in this approach is to find a state of Ψ that enables the corresponding estimate $p_\Psi(C_j|u_1^{T_0})$, which is a function of Ψ, to precisely approximate the true *a posteriori* probability (density) $p(C_j|u_1^{T_0})$, or in other words, to adjust Ψ so that $p_\Psi(C_j|u_1^{T_0})$ can approximate $p(C_j|u_1^{T_0})$ as precisely as possible. For example, one could attempt a direct estimate of this *a posteriori* probability for the dynamic instantiation by using a system having a sufficiently large approximation capability. However, to the best of the authors' knowledge, there has not been a successful design example based on such an optimistic strategy. Therefore, an alternative to this ideal but simple-minded attempt is obviously needed.

Actually, most speech recognizers are transparent (i.e., an internal process is explicitly described) modular systems. Such a recognizer consists of several observable modules, each of which is carefully designed based on scientific experiences. As illustrated in Fig. 1, a typical recognizer consists of: 1) a feature extractor (feature extraction module) and 2) a classifier (classification module) that is further divided into a language model and an acoustic model. Let us represent the adjustable parameter sets of the feature extractor, the acoustic model, and the language model by Φ, Λ and Ξ, respectively: $\Psi = \Phi \cup \Lambda \cup \Xi$.

The feature extraction module converts a speech wave sample $u_1^{T_0}$ to a dynamic pattern $x_1^T = (z_1, z_2, \ldots, z_t, \ldots, z_T)$ that is a sequence of static (fixed-dimensional) F-dimensional acoustic feature vectors, where T is the duration of the dynamic pattern and z_t is the tth feature vector of the sequence. The feature vector is generally made up of cepstrum or bank-of-filters output coefficients. Φ is then the designable parameter set, such as lifter and bank-of-filters functions, that controls the nature of the feature vectors.

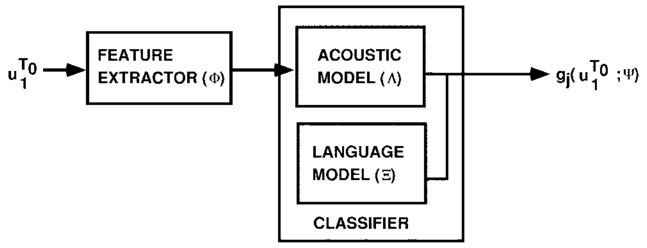


Fig. 1. Typical structure of a modular speech recognizer.

Next, the classification module assigns a class index to this converted feature pattern. This assignment is generally performed by using the classification rule

$$C\left(x_1^T\right) = C_i, \quad \text{iff } i = \arg\max_j \, p\left(x_1^T | C_j\right) p(C_j) \tag{2}$$

which is conceptually equivalent to (1). Note in (2) that in accordance with the Bayes rule of probability, the *a posteriori* probability is replaced by the conditional probability (density) and the *a priori* probability, which are both suited for the estimation based on the well-analyzed maximum likelihood (ML) method. In fact, the conditional probability [density] $p(x_1^T|C_j)$ is often computed as the estimate $p_\Lambda(x_1^T|C_j)$ using hidden Markov models (HMM's) for the acoustic models; Λ corresponds to a set of model parameters, such as the HMM state transition probabilities and the mean vectors of the Gaussian continuous HMM's. Also, the *a priori* probability $p(C_j)$ is often computed as the estimate p_Ξ (C_j) using a language model such as an N-gram and a probabilistic context-free grammar; Ξ then corresponds to the probability of the N-gram and the parameters that determine the grammatical rules.

B. Classifier Design Based on ML Method

In the same sense as (1), substituting accurate estimates for the probabilities in (2) enables one to fundamentally achieve the optimal, minimum classification error status. Thus, it seems plausible to consider the accurate estimation of these conditional and *a priori* probabilities to be a desirable design objective. Actually, the classifiers of most existing recognizers have been designed based on the design principle of this ML approach (to classification); that is, the expectation-maximization method, which is an extended ML estimation method for incomplete data[1] [9], [10], [28], and segmental k-means clustering [50] are used for training the HMM acoustic model, and a simple computation of the relative frequency of occurrence of symbols (e.g., phonemes or words) is used for designing the N-gram language model [49]. Note here that the minimum distortion principle, which underlies the design of the reference pattern-based models of a distance classifier (widely used prior to HMM classifiers) is fundamentally equivalent to this ML principle.

However, this conventional ML-based approach actually has a basic problem in that the functional form of the class

[1]The incompleteness here corresponds to the fact that one cannot observe the state transition behavior of HMM.

distribution (the conditional probability density) function to be estimated is in practice rarely known, and the likelihood maximization of these estimated functions, performed to model each entire class distribution individually, is not direct with regard to the minimization of classification errors (the accurate estimate of class boundaries). Also, the ML-based approach covers only the classifier design; it does not optimize the overall recognizer, or in other words, its design target is too far from emulating the original decision strategy (2).

The techniques of feature extraction and probability estimation are described in detail in textbooks such as [85].

C. Classifier Design Based on a Discriminant Function Approach

Recently, an alternative to the common ML approach, based on the concept of discriminative training, has been vigorously investigated to especially improve the acoustic model parameter Λ. Discriminative training has been called many different names because it has various backgrounds, including multivariate analysis, artificial intelligence, and artificial neural networks (ANN). In fact, it is sometimes called competitive learning and discriminant analysis. In this paper, we refer to it as the discriminant function approach (DFA) that has been widely used in pattern recognition.

In DFA, a discriminant function $g_j(x_1^T;\Lambda)$ is introduced (for C_j) to measure the class membership of the input x_1^T (the degree to which x_1^T belongs to one class), where one should note that the discriminant function is a function of the classifier parameters Λ. This discriminant function does not need to be a probability function; it can be any reasonable type of measure, such as distance or similarity. In the approach, the following decision rule is used in place of (2):

$$C(x_1^T) = C_i, \quad \text{iff } i = \arg\max_j \, g_j(x_1^T;\Lambda). \tag{3}$$

In this approach, Λ is trained in order to reduce a loss that reflects a classification error in a certain manner. Since the classification result is evaluated in the design stage, this approach is fundamentally more direct with regard to the minimization of classification errors than the ML-based approach where class model parameters are designed independently of each other. In fact, designs using this approach have successfully improved the classification accuracy of ML-based baseline systems in various speech classification tasks, as will be discussed later. However, there was plenty of room left for improvement in this apparently powerful DFA: each of these designs had a mathematical or procedural inadequacy, as summarized in the following.

1) Execution of rule (3) using an arbitrary measure as the discriminant function does not necessarily lead to the minimum Bayes error probability situation.
2) The design scope stays within the acoustic modeling and does not cover the overall recognizer.
3) Most of the existing training procedures are empirical or heuristic, and their mathematical optimality is thus unclear.

The following is a review of the research situation concerning DFA-based acoustic model training in the years around 1990, which was actually a direct motivation for GPD development. In the early stages of investigation, attempts were made to improve HMM acoustic models. The concept of maximum mutual (interclass) information was incorporated in the model design [7], and corrective training similar to traditional error-correction training was developed [8]. In the next stage, ANN concepts such as feed-forward network (FFN) [91] and learning vector quantization (LVQ) [61] were applied to the acoustic modeling. Typical examples of such applications are categorized as follows, based on system structure and training methods.

1) Discriminative (of DFA) ANN with a time-delay structure.
   a) FFN- and LVQ-based systems with a time-delay structure were proposed, aiming to accurately classify a short speech fragment [70], [101].
   b) An analog ANN with a time-delay structure was developed for continuous speech pattern classification [99].
2) A hybrid of an ANN and nonlinear dynamic time warping (DTW).
   a) ANN classifiers with a time-delay structure were used for front-end processing of DTW [74].
   b) FFN estimation of local probabilities of discriminative HMM's were used for front-end processing of DTW [16].
   c) DTW was used for front-end processing of ANN static pattern classifiers [34], [43], [57], [92].
3) A discriminative HMM based on ANN design concepts.
   a) LVQ was used for designing the codebook of a discrete HMM [47], [48], [60], [109].
   b) Empirical training rules similar to LVQ were applied to the design of the mean vectors of the Gaussian distributions of a continuous (Gaussian) HMM [75].
   c) The concept of a recurrent network was applied to an HMM, and discriminative HMM classifiers were designed by using a training objective, which is equivalent in its fundamentals to the maximization of mutual information [18], [78].

Actually, these methods led to successful results to some extent. However, as cited before, the resulting recognizers were not necessarily satisfactory. There are two likely causes. The first is that, as summarized in the following, the discriminative training procedures used therein were mathematically inadequate.

1) Empirical rules such as error correction learning and LVQ did not have enough of a mathematical basis to guarantee design optimality.
2) The mathematical properties, such as training convergence, of an adaptive version of the error-back propagation (EBP) used for designing the FFN's were unclear. Note that here the term "adaptive" means a procedure in which learning (adjustment) was performed every time one design sample was presented.
3) As is well known, the minimization of the squared error (MSE) loss between a classifier output and its corresponding supervising signal is not necessarily equivalent to the minimization of misclassifications [29], [39].
4) The maximization of mutual information does not necessarily imply the minimization of misclassifications (see Section IV).

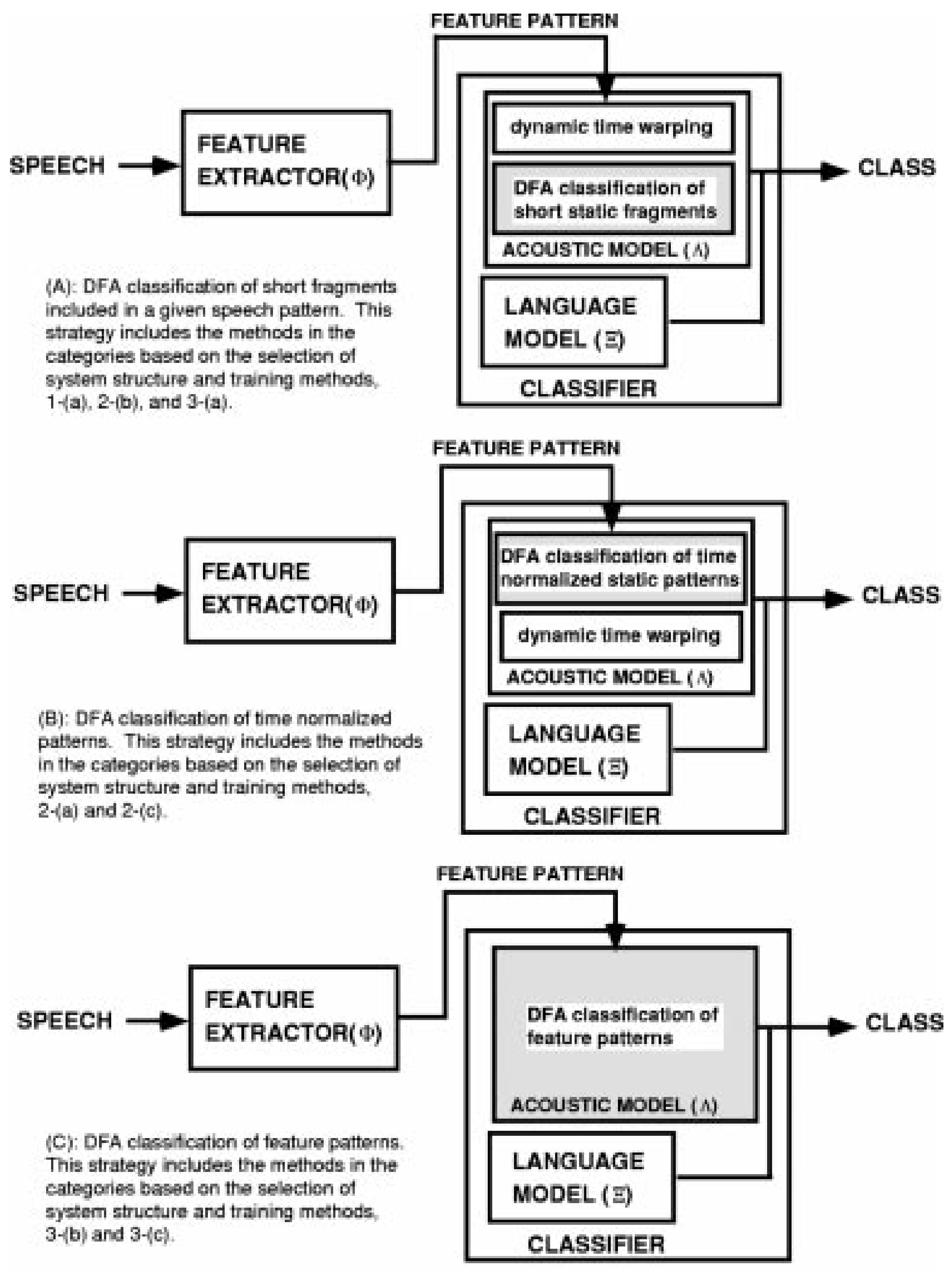
The second possibility is that improvement efforts were too limited to acoustic modeling and lacked the global scope of designing an overall recognizer. With this in mind, we reexamine the classifier examples introduced above and recategorize them according to the location of the DFA-based design execution in Fig. 2. Fig. 2(A) illustrates a method of using DFA to increase the classification accuracy of short speech fragments, each being merely a part of the input feature pattern (usually consisting of one or several adjacent feature vectors). The original input dynamic feature pattern is first mapped to a new dynamic pattern by using highly discriminative (DFA-designed) fragment models, then this new pattern is classified with DTW; that is, this acoustical modeling is performed using a hybrid form of the highly discriminative modeling of short fragments and the DTW-based concatenation of these models. Note here that the effort made to increase classification capability does not bear directly on the classification of the overall dynamic pattern, and also that the DTW process is based on the minimum distortion or ML principles, which is not necessarily relevant to an increase of classification accuracy. It is therefore obvious that this improvement attempt is not consistent with the optimal classification of the dynamic pattern. On the other hand, Fig. 2(B) illustrates a method of first mapping the input dynamic pattern to a static pattern with DTW and then performing classification with highly discriminative acoustic models, which are designed for this new static pattern representation. Here, the static pattern models are designed based on a minimum loss principle (e.g., MSE loss) that is different from the minimization of classification error counts. It is thus obvious that these hybrid methods are also inconsistent with the optimal structure for classifying the dynamic pattern. The method shown in Fig. 2(C) trains HMM parameters by using a discriminative training method, aiming at a direct increase of dynamic pattern classification accuracy. This is clearly a more advanced approach than before because it attempts to directly improve the HMM acoustical models corresponding to the dynamic pattern representation. However, the figure obviously shows that even this method suffers from the problem of narrow design scope, as had the others.

D. Motivation of GPD Development and Paper Organization

From the above survey, one may recognize the necessity of a novel design method for pursuing the overall optimality of a recognizer that covers the acoustic modeling process as well as the feature extraction and language modeling processes. Note here that although we have discussed the problems in speech recognition, most of the arguments hold true in many other cases of pattern recognition. The development of GPD was motivated by such circumstances in the problems of pattern recognizer design.

GPD relies on traditional adaptive discriminative training, which is called the probabilistic descent method (PDM) [1], [2], for pattern classification. In the early stage of development, GPD was formalized as a training algorithm for classifiers [54]. It was then extended to a more general method for designing an entire recognizer, e.g., [58]; it was also extended to particular tasks such as keyword spotting, e.g., [63]; it was applied to speaker recognition as well, e.g., [66]. Its extension through such continued research efforts has resulted in a novel family of discriminative training methods, members of which are based upon the original formalization concept of GPD.

In this paper, we shall comprehensively present the GPD-based approach to pattern recognizer design. The paper is organized as follows. Following the development history of the original formalization of GPD and its extensions, we start our description by focusing on design problems in pattern classification. In Section II, we provide the fundamentals of the conventional, DFA-based pattern classification, and discuss the background of GPD development. In Section III, we introduce GPD. Its formalization and mathematical nature, such as the training optimality, are described in detail. So as to maintain the concreteness of the formulation as well, we focus in this section on classifier design problems using a multiple reference distance classifier which has long been used for speech pattern classification and is the structure for the LVQ pattern quantizer. In Section III we also summarize the concept of minimum classification error learning (MCE), which is closely related to the GPD formalization and has a significant theoretical contribution to pattern classification study [52]. In Section IV we discuss the relationship between GPD and existing design methods. In Section V we introduce derivatives of GPD, or in other words, algorithms based on extensions of the GPD concept. The paper is then concluded in Section VI. Several additional issues related to GPD implementation are finally provided in the Appendixes.



(A): DFA classification of short fragments included in a given speech pattern. This strategy includes the methods in the categories based on the selection of system structure and training methods, 1-(a), 2-(b), and 3-(a).

(B): DFA classification of time normalized patterns. This strategy includes the methods in the categories based on the selection of system structure and training methods, 2-(a) and 2-(c).

(C): DFA classification of feature patterns. This strategy includes the methods in the categories based on the selection of system structure and training methods, 3-(b) and 3-(c).

Fig. 2. Schematic explanation of the role of the discriminant function approach classification in recent ANN-based approaches to speech pattern recognition.

II. Discriminative Pattern Classification

A. Bayes Decision Theory

We first summarize the Bayes decision theory, which underlies most approaches to pattern classification, including the DFA approach. We assume for simplicity of discussion that given an observed feature pattern sample, we aim to classify it accurately. What is "accuracy" here? The meaning of this common term is not necessarily clear for a particular operation of classification. A significant contribution of the Bayes decision theory is to give this accuracy a general statistics-based definition, i.e., the minimum expected loss (risk) situation, and to show that this situation can be achieved by observing the Bayes decision rule.

For the general task of classifying M-class patterns, the Bayes theory is formulated as follows (see [29] for details). According to the conventional approach in this theoretical framework, we assume a sample to be static. A static feature pattern $\boldsymbol{x}(\in C_k)$ is given. To measure the accuracy, the theory first introduces an (individual) loss $\ell_k(C(\boldsymbol{x}))$ that is incurred by judging the given pattern's class C_k to be one of the M possible classes, where $C(\cdot)$ denotes the classification operation, as in (1). It is obvious that the accuracy of the task should be evaluated over all of the possible samples. Thus, based on statistics, the theory next introduces an expected loss incurred by classifying $\boldsymbol{x}$, called the conditional risk, as

$$L(C(\boldsymbol{x})|\boldsymbol{x}) = \sum_k \ell_k(C(\boldsymbol{x}))1(\boldsymbol{x} \in C_k)P(C_k|\boldsymbol{x}) \tag{4}$$

and also introduces an expected loss associated with $C(\cdot)$, called the overall risk, as

$$\mathcal{L} = \int L(C(\boldsymbol{x})|\boldsymbol{x})p(\boldsymbol{x})\, d\boldsymbol{x} \tag{5}$$

where $1(\mathcal{A})$ is the following indicator function

$$1(\mathcal{A}) = \begin{cases} 1, & \text{(if } \mathcal{A} \text{ is true)} \\ 0, & \text{(otherwise).} \end{cases} \tag{6}$$

The accuracy is accordingly defined by the overall risk. The smaller the overall risk, the better its corresponding classification result. A desirable decision is thus the very one that minimizes the overall risk. Consequently, the theory leads to the following well-known Bayes decision rule that is justified by (5).

Rule 1 (Bayes Decision Rule): To minimize the overall risk, compute all of the (M) possible conditional risks $L(C_i|\boldsymbol{x})$ $(i = 1, \cdots, M)$ and select the C_j for which $L(C_j|\boldsymbol{x})$ is minimum.

B. Minimum Error Rate Classification

Using the loss enables one to evaluate classification results flexibly. This is one of the big attractions of the Bayes decision theory. However, in practice, a loss that evaluates results uniformly for all of the possible classes (i.e., class-independent loss) has been used widely due to the difficulty of setting losses in a reasonable class-by-class manner. A natural loss in this simple case is the following error count loss:

$$\ell_k(C(\boldsymbol{x})) = \begin{cases} 0, & (C(\boldsymbol{x}) = k) \\ 1, & \text{(otherwise)} \end{cases} \tag{7}$$

and then its corresponding conditional risk becomes

$$L(C(\boldsymbol{x})|\boldsymbol{x}) = 1 - P(C(\boldsymbol{x})|\boldsymbol{x}) \tag{8}$$

and the overall risk becomes the average probability of error. Thus, the minimization of the overall risk using (7) leads to the minimization of the average probability of error, or in other words, the minimum error rate classification (e.g., [29]).

According to the above Bayes decision rule, it is obvious that the desirable classification decision is the one that minimizes the conditional risk (8), or maximizes the *a posteriori* probability $P(C(\boldsymbol{x})|\boldsymbol{x})$. As far as minimum error classification is concerned, the best classifier design is theoretically achieved by simulating (8) as accurately as possible. Hence this way of thinking justifies the Bayes approach of aiming at a correct estimation of the *a posteriori* probability, or its corresponding *a priori* probability and conditional probability. Note here that we assume the Bayes approach includes the ML method and the so-called Bayesian approach [29]. The principal attraction of the Bayes approach may be that the classifier design is primarily based on a mathematically well-analyzed probability computation. However, in realistic situations where only limited resources are available, an accurate estimation of these probability functions is difficult, and generally this approach does not achieve satisfactory design results.

Originally, the probability functions are given for the task at hand. In the Bayes approach, the classifier design is replaced by the estimation of these functions. Taking into account the fact that these given functions are unobservable and essentially difficult to estimate accurately, one may have to find an alternative approach to the design; that is, one can consider that the design is to execute an accurate evaluation based on (7) for every sample. This is the very concept of DFA design. However, unfortunately, this point of view has not been explicitly indicated in most conventional DFA design attempts. Usually, the design formalization has started with (3), i.e., merely selecting arbitrary discriminant functions.

C. Discriminant Function Approach

Let us discuss in more detail the DFA design for a task of classifying the static pattern $\boldsymbol{x}$. The decision rule is

$$C(\boldsymbol{x}) = C_i, \quad \text{iff } i = \arg\max_j g_j(\boldsymbol{x}; \Lambda). \tag{9}$$

As cited before, DFA design is fundamentally performed in a competitive way (with regard to classes), aimed at realizing the discriminant function set (consequently Λ) that achieves the least classification error over the design sample set. Its implementation is basically characterized by the following factors:

1) functional form of the discriminant function (classifier structure or measure);
2) design (training) objective;
3) optimization method;
4) consistency with unknown samples (robustness, generalization capability, or estimation sensitivity).

Classifier structure, which determines the type of Λ, is generally selected based on the nature of the patterns, and this selection often determines the selection of the measure or the selection of $g_j(\boldsymbol{x}; \Lambda)$. The linear discriminant function is a typical example of a classical classifier structure, and the measure used therein is a linearly-weighted sum of input vector components. A distance classifier that uses the distance between an input and a reference vector as the measure is another widely used example.

The design objective is a function used for evaluating a classification result in the design stage and is equivalent to the concept of risk in Bayes decision theory. Usually, an individual loss that is a design criterion for an individual design sample is first introduced; the individual loss for

$\boldsymbol{x}$ $(\in C_k)$ is denoted by $\ell_k(\boldsymbol{x};\Lambda)$. As discussed in the previous section, a natural loss form is the classification error count as

$$\ell_k(\boldsymbol{x};\Lambda) = \begin{cases} 0, & (C(\boldsymbol{x}) = k) \\ 1, & \text{(otherwise)} \end{cases} \tag{10}$$

where the fact that Λ is included in the loss definition means that the individual loss is a function of Λ. Next, similar to the overall risk, an ideal overall loss, i.e., the expected loss, is defined using the individual loss as

$$L(\Lambda) = \sum_k P(C_k) \int \ell_k(\boldsymbol{x};\Lambda) p(\boldsymbol{x}|C_k)\, d\boldsymbol{x}. \tag{11}$$

However, since sample distributions are essentially unknown, it is impossible to use this expected loss in practice. For this reason, an empirical average loss is usually used, as defined in the following:

$$L_0(\Lambda) = \frac{1}{N} \sum_k \sum_n \ell_k(\boldsymbol{x}_n;\Lambda) 1(\boldsymbol{x}_n \in C_k) \tag{12}$$

where n of $\boldsymbol{x}_n$ explicitly means that the sample $\boldsymbol{x}_n$ is the nth sample of the finite (consisting of N samples) design sample set $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n, \ldots, \boldsymbol{x}_N\}$. In the case of using the error count loss of (10), this empirical average loss becomes the total count of classification errors measured over X.

In addition to (10), several different forms of loss have been used, e.g., perceptron loss, squared error loss (between a discriminant function and its corresponding supervising signal), and mutual information. However, these are merely temporary expedients, and it is known that the design results obtained by using them are not necessarily consistent with minimum classification error probability condition (e.g., see [29]; also see Section IV-B for the use of mutual information).

Optimization is a method of finding the state of Λ that minimizes loss over X, and its embodiments are grouped into a heuristic method and a mathematically proven algorithm.

Among the many examples of heuristic optimization, error correction training has been widely used (e.g., see [79]). Most of these heuristic methods do not guarantee, however, the achievement of a true optimal situation due to the lack of a theoretical justification.

The mathematically proven methods are further grouped into ones which search for locally optimal conditions and ones which search for a globally optimal condition. Traditionally, a practical, gradient search-based, local optimization method such as the steepest descent method has been used. In recent ANN frameworks, global optimization methods, such as simulated annealing and its special case, called the Boltzmann machine, have also been extensively studied [35], [91]. Their fundamental capability of globally optimizing the objective function is fascinating, even though the optimization is done in a probabilistic sense. Moreover, these ANN-related methods do not require the continuity, or the smoothness (being at least the first differentiable in system parameters), of the loss function (see the following paragraphs in this section and Section II-D3). In fact, simulated annealing was applied to the minimization of the unsmooth, average empirical loss based on (10) [4]. However, these global optimization methods are usually impractical due to their time-consuming nature.

Optimization methods are categorized from another point of view, i.e., the batch type search versus the adaptive search. The batch type search aims to minimize (or locally minimize) the average empirical loss. The steepest descent method is a typical example of the batch type local minimization method. The adaptive search minimizes the individual loss $\ell_j(\cdot;\Lambda)$ for a small set of design samples (usually one sample) randomly selected from X; the adjustment of Λ is repeated while changing this set of samples. Methods based on stochastic approximation are traditional examples of adaptive local minimization (e.g., [29], [32]).

Ideally, the purpose of classifier design is to realize a state of Λ that leads to accurate classification over all of the samples of a task at hand instead of the given design samples. Remember that the expected loss was defined as an ideal overall loss. Thus it is desirable that the design result obtained by using design samples be consistent with unknown samples too. Obviously, the pursuit of such a result for unknown data requires some assumptions concerning unobservable, unknown samples or about the entire sample distribution of the task. In contrast with the Bayes approach, measures to increase the consistency in DFA are generally moderate: the loss is individually defined for every design sample, and the design is essentially formed using the given design samples, as shown in the use of the average empirical loss (12).

Naturally, the most practical method of DFA is to search for the state of Λ that corresponds to the minimum of the average empirical loss consisting of (10). However, (10) is not a smooth function (not differentiable in Λ). Thus, the corresponding average empirical loss may be difficult to minimize directly. Moreover, this average empirical loss is sensitive only to design samples (see Section III-C), which means that a design using this loss may not clearly contribute to increasing the consistency with unknown samples. Consequently, a fundamental improvement for DFA is desired.

We have considered the classification of static patterns in this section. The issues discussed above also hold true in the classification of dynamic speech patterns: the discussion does not depend upon the static nature of patterns. Hence, the most effective design method for dynamic pattern classification can again be the local minimization of the average empirical, classification error count loss. Detailed discussions about dynamic pattern classification will be given in Section II-D3.

D. Probabilistic Descent Method

1) Formalization: Almost 30 years ago, PDM was developed as a practical method for locally minimizing the expected loss based on the gradient search [1], [2]. In particular, its minimization mechanism is based on stochastic approximation. Clearly, this method can be applied to locally minimizing the practical, average empirical loss consisting of individual classification error count losses. The use of stochastic approximation for pattern classification has a long history and, in fact, many studies have been done to deal with it, as shown in [29] and [32]. However, PDM is clearly distinguished from other stochastic approximation-based methods by the following two points.

1) It proves that the reduction of individual losses leads to a reduction of the expected loss in the probabilistic sense.
2) It proposes a design concept that enables the classifier design problem to be handled systematically by incorporating the classification process in a three-step functional formalization.

Our purpose in this section is to show the background of GPD development by reviewing classical PDM. Let us consider the previous M-class pattern classification task. It is assumed in PDM that a pattern is static. As cited in Section I-D, we use a distance classifier consisting of multiple reference vectors as our formalization framework: $\Lambda = \{\{\boldsymbol{r}_j^b\}_{b=1}^{B_j}\}_{j=1}^{j=M}$, where $\boldsymbol{r}_j^b$ is the bth closest reference vector of C_j (in the sense of squared Euclidean distance to an input pattern $\boldsymbol{x}$, described later) defined in the same vector space as $\boldsymbol{x}$, and B_j is the number of C_j's reference vectors. The classification rule that we consider is

$$C(\boldsymbol{x}) = C_i, \quad \text{iff } i = \arg\min_j g_j(\boldsymbol{x};\Lambda) \tag{13}$$

which is essentially the same as (3).

Suppose our design is to be done adaptively; that is, we update the classifier parameters Λ every time one design sample is presented, and we pursue the optimal status of the parameters by repeating this updating procedure. Let us assume that $\boldsymbol{x}(t)$ of C_k is presented at the design stage time index t (natural number). PDM is then formalized in the following three-step manner.

The first step is to choose the discriminant function $g_j(\boldsymbol{x}(t);\Lambda)$. Naturally, for every class, we define the function as the squared Euclidean distance between the input and its closest reference vector

$$\begin{aligned} g_j(\boldsymbol{x}(t);\Lambda) &= d_E\big(\boldsymbol{x}(t), \boldsymbol{r}_j^{c_j}\big) = \|\boldsymbol{x}(t) - \boldsymbol{r}_j^{c_j}\|^2, \\ \text{where } c_j &= \arg\min_b d_E\big(\boldsymbol{x}(t), \boldsymbol{r}_j^b\big). \end{aligned} \tag{14}$$

In the second step, a misclassification measure $d_k(\boldsymbol{x}(t);\Lambda)$ is introduced so as to emulate the decision process of (13), i.e., the comparison/decision among the competing classes

$$d_k(\boldsymbol{x}(t);\Lambda) = \frac{1}{N_t}\sum_{C_j\in\Im}\{g_k(\boldsymbol{x}(t);\Lambda) - g_j(\boldsymbol{x}(t);\Lambda)\} \tag{15}$$

where $\Im$ is a set of confusing classes defined as

$$\Im = \{C_j; g_j(\boldsymbol{x}(t);\Lambda) > g_k(\boldsymbol{x}(t);\Lambda)\} \tag{16}$$

and N_t is the number of $\Im$ classes. Note in (15) that (13) is represented by a scalar value decision. A positive value of $d_k(\,)$ means a misclassification, and a negative value means a correct classification.

The third step conforms to the same loss evaluation as DFA. The misclassification measure is embedded in a loss in order to evaluate the corresponding classification result. A general form of the loss is represented as

$$\ell_k\ (\boldsymbol{x}(t);\Lambda) = l(d_k(\boldsymbol{x}(t);\Lambda)) \tag{17}$$

where $l(\,)$ is a monotonically increasing function. Various definitions of the loss are possible; for example

$$\ell_k(\boldsymbol{x}(t);\Lambda) = \begin{cases} \{d_k(\boldsymbol{x}(t);\Lambda)\}^{\varrho_1}, & (d_k(\boldsymbol{x}(t);\Lambda) > 0) \\ 0, & \text{(otherwise)} \end{cases} \tag{18}$$

and

$$\ell_k(\boldsymbol{x}(t);\Lambda) = \begin{cases} 1, & (\varrho_2 < d_k(\boldsymbol{x}(t);\Lambda)) \\ \dfrac{d_k(\boldsymbol{x}(t);\Lambda)}{\varrho_2}, & (0 < d_k(\boldsymbol{x}(t);\Lambda) \le \varrho_2) \\ 0, & (d_k(\boldsymbol{x}(t);\Lambda) \le 0) \end{cases} \tag{19}$$

where ϱ_1 and ϱ_2 are positive numbers. The loss in (19) is a linear approximation of the classification error count. In particular, the loss in (18) approximates the classification error count when $\varrho_1 \to 0$; similarly, the loss in (19) approximates the classification error count when $\varrho_2 \to 0$.

2) Probabilistic Descent Theorem: The PDM training by which the parameters are adjusted through the above three-step formalization is summarized in the following probabilistic descent theorem.

Theorem 1 (Probabilistic Descent Theorem): Assume that a given design sample $\boldsymbol{x}(t)$ belongs to C_k. If the classifier parameter adjustment $\delta\Lambda(\boldsymbol{x}(t), C_k, \Lambda)$ is specified by

$$\delta\Lambda(\boldsymbol{x}(t), C_k, \Lambda) = -\epsilon \boldsymbol{U}\nabla\ell_k(\boldsymbol{x}(t);\Lambda) \tag{20}$$

where $\boldsymbol{U}$ is a positive-definite matrix and ϵ is a small positive real number, then

$$E[\delta L(\Lambda)] \le 0. \tag{21}$$

Furthermore, if an infinite sequence of randomly selected samples $\boldsymbol{x}_t$ is used for learning (designing) and the adjustment rule of (20) is utilized with a corresponding [learning] weight sequence $\epsilon(t)$ which satisfies

$$\sum_{t=1}^{\infty} \epsilon(t) \to \infty \tag{22}$$

$$\sum_{t=1}^{\infty} \epsilon(t)^2 < \infty \tag{23}$$

then the parameter sequence $\Lambda(t)$ (the state of Λ at t) according to

$$\Lambda(t+1) = \Lambda(t) + \delta\Lambda(\boldsymbol{x}(t), C_k, \Lambda(t)) \tag{24}$$

converges with probability one to Λ^* which is at least a local minimum of $L(\Lambda)$.

The nature of the adjustment convergence, such as accuracy and speed, is analyzed in detail in [2].

It is obviously unrealistic to rigidly observe the infinitely repeated probabilistic descent adjustments. In practice, the learning coefficient $\epsilon(t)$ is often approximated by a finite monotonically decreasing function such as

$$\epsilon(t) = \epsilon(0)\left(1 - \frac{t}{N_{\max}}\right) \tag{25}$$

where $N_{\max}$ is a preset number of adjustment repetitions. In particular, when the individual loss of (19) is used and $\boldsymbol{U}$ is assumed for simplicity to be a unit matrix, the adjustment rule for our multireference distance classifier is given "speciously" (the reason for using this particular "critical" term will be shown later) as follows. If and only if $0 < d_k(\boldsymbol{x}(t); \Lambda) \leq \varrho_2$ holds

$$\boldsymbol{r}_j^q(t+1) = \begin{cases} \boldsymbol{r}_j^q(t) + \dfrac{2\epsilon(t)\big(\boldsymbol{x}(t) - \boldsymbol{r}_j^q(t)\big)}{\varrho_2}, \\ \qquad \left(\text{for } j = k \text{ and } q = \arg\min_b d_E\big(\boldsymbol{x}(t), \boldsymbol{r}_j^b\big)\right) \\ \boldsymbol{r}_j^q(t) - \dfrac{2\epsilon(t)\big(\boldsymbol{x}(t) - \boldsymbol{r}_j^q(t)\big)}{\varrho_2}, \\ \qquad \left(\text{for } C_j \in \Im \text{ and } q = \arg\min_b d_E\big(\boldsymbol{x}(t), \boldsymbol{r}_j^b\big)\right) \\ \boldsymbol{r}_j^q(t), \quad (\text{otherwise}). \end{cases} \tag{26}$$

The probabilistic descent theorem shows that an infinite repetition of the adjustment (20) based on the gradient computation of the individual loss for one training sample leads to a local minimum of the expected loss. Using an infinite number of samples is essentially equivalent to a situation in which the corresponding sample distribution is accessible. Therefore, criticizing that the minimum classification error probability status is known in this ideal situation, one may doubt the technical significance of PDM (or the methods based on the stochastic approximation).

However, the contribution of PDM is clear. In reality, design samples are finite and it is impossible to know the minimum classification error probability situation. In the Bayes approach, the estimation of the class distributions obviously suffers from estimation errors that may spread over the entire sample space instead of the region near the class boundaries. Note here that accurate classification relies on an accurate estimation of the class boundary. In contrast with this approach, PDM aims to minimize the average empirical loss in a manner that corresponds directly to the classification task [the classification rule (13)]; in particular, PDM design using (19) will minimize the actual classification error count, resulting in an accurate class boundary estimation. This point is clearly distinct from other DFA methods as well as the Bayes approach when the exact distribution form is not known to the designer. If this minimization (even the local minimization) is done successfully, the resulting classification performance should be superior to that of the Bayes approach. Moreover, though finite, a huge number of design samples are often used. In fact, it is often difficult to store all of the design samples for training. Given a finite set of design samples X, the PDM adjustment will be repeated by extracting a sample $\boldsymbol{x}_t$ randomly from X. Furthermore, it is often desired that a classifier be adaptable to new circumstances in a realistic situation where future samples cannot be observed in the design stage. PDM provided a fundamental solution based on simple gradient computation to this adaptive design. Therefore, the true fundamental value of PDM should be widely recognized, even though the practical aspects of its adaptive learning mechanism must be further investigated.

Recall the adaptive training version of EBP and LVQ. Both are useful but have only intuitive validity (meaning that the convergence mechanism has not been mathematically elaborated). One should note that the probabilistic descent adjustment is quite similar to adaptive EBP [3], [5]. In fact, the adjustment principle of adaptive EBP is to locally minimize the loss for a given design sample according to its gradient; also, the mechanism of error back-propagation is simply a special case of differential calculus for a functional embedded in a smooth functional form. Moreover, one should note that (26) closely resembles LVQ, especially an improved version of LVQ2 [69]. Furthermore, the monotonically-decreasing learning weight function used in these learning methods is essentially the same as (25). Thus, it appears that PDM may be useful for analyzing these recent implementations of DFA.

3) Problems: Encouraged by the advantages of PDM, i.e., the direct minimization of the classification error count and the use of an adaptive training mechanism, which were described in the last few paragraphs of Section II-D2, one might attempt to classify dynamic patterns with PDM. However, PDM is a method for the classification of static patterns. Some additional procedures will thus be needed for applying this method to dynamic pattern classification. Furthermore, the formulation of PDM actually suffers from a mathematical difficulty. Solving this inadequacy was the motivation for the development of the more general GPD. In this section, we clarify the problems of PDM.

Recall that we used the critical term "speciously" in the derivation of (26). Actually, the derivation is mathematically impossible.

The probabilistic descent adjustment rule is based on gradient computation. The overall process included in the formulation must thus be differentiable. However, the "min" operation selecting the closest reference vector, in (14), is not a differentiable function; $\Im$ can change discretely when Λ changes, with the result that the misclassification measure is also discontinuous. It then becomes apparent that the PDM formalization is inadequate. To apply the probabilistic descent adjustment in the right way, this lack of smoothness must therefore be overcome.

One should note that the above smoothness difficulty does not appear in some special cases. In fact, the problem concerning (14) does not occur for a classifier that represents each class by only one reference pattern. Moreover, the problem concerning (14) does not occur in a two-class task

$(M = 2)$. However, these are very limited and unrealistic situations. A satisfactory method should be able to handle general and realistic tasks.

Let us next consider the task of classifying spoken English letter syllables, such as $\{b\}$ and $\{c\}$, by using a distance classifier. We assume here that each syllable is uttered in an isolated mode and is converted to an acoustic feature vector sequence with some preset feature extractor. It is also assumed that speech signal is correctly extracted from its surrounding signal and input to the system. A dynamic pattern to classify is then represented as

$$x_1^T = (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_\tau, \ldots, \boldsymbol{z}_T) \in \underbrace{\Re^F \times \cdots \times \Re^F}_{T} \tag{27}$$

where $\boldsymbol{z}_\tau$ is the fixed F-dimensional component vector (usually corresponding to an acoustic feature vector or a short sequence of such acoustic feature vectors) at the time τ ($\boldsymbol{z}_\tau \in \Re^F$, where F is a fixed natural number), and T is a variable but finite natural number that represents the duration of the component vector sequence.

A simple application of PDM for classifying this dynamic speech pattern would be similar to the structural hybrid system described in Section I; i.e., PDM would be used to increase the discriminative power of a distance classifier consisting of reference vectors defined in the same F-dimensional component vector space, and DTW would be incorporated with this classifier in order to handle the dynamics of the entire input pattern. However, this local improvement of classification capability is not necessarily consistent with improvement of classification accuracy for the dynamic pattern. Thus, a natural solution to this problem must be to improve the overall classification process, including the dynamics normalization of a DTW distance classifier (which has long been used in speech pattern classification) by using PDM.

A DTW distance classifier is generally given as

$$\Lambda = \{\lambda_j\} = \left\{ \{r_j^b\}_{b=1}^{b=B_j} \right\}_{j=1}^{j=M} \tag{28}$$

where r_j^b is C_j's bth finite-length dynamic reference pattern.[2] The distance computation based on the DTW procedure between dynamic patterns is used for the classification decision. First, a DTW matching path and its corresponding path distance is introduced between an input pattern and each reference pattern. The path distance is given as

$$\begin{aligned} D_\theta(x_1^T, r_j^b) &= \sum_{\tau=1}^{T} w_{j,\tau}^b \delta_{\varpi(j,b,\tau,\theta)} \\ &= \|\boldsymbol{z}_{\varpi(j,b,\tau,\theta)} - \boldsymbol{r}_{j,\tau}^b\|^2 \end{aligned} \tag{29}$$

where $\boldsymbol{z}_{\varpi(j,b,\tau,\theta)}$ is the $\varpi(j,b,\tau,\theta)$th component vector of x_1^T, to which the τth component vector of $r_j^b, \boldsymbol{r}_{j,\tau}^b$ corresponds along the θth matching path of r_j^b; $\delta_{\varpi(j,b,\tau,\theta)}$ which is a local distance defined by the squared Euclidean distance between these corresponding component vectors (note that the local distance can be defined by using any other reasonable distance measure); $w_{j,\tau}^b$ is a weight coefficient. Next, using the path distances, the distance between an input and each reference pattern, i.e., a reference pattern distance, is defined as the smallest (best) path distance for each reference pattern

$$d_A(x_1^T, r_j^b) = \min_\theta \{D_\theta(x_1^T, r_j^b)\}. \tag{30}$$

Usually, the search for the smallest path distance is performed by dynamic programming (DP). Lastly, the discriminant function that represents the degree to which an input belongs to each class is defined as the reference pattern distance of the closest [in the sense of (30)] reference to the input

$$\begin{aligned} &g_j(x_1^T; \Lambda) = d_A(x_1^T, r_j^{c_j}), \\ &\quad \text{where } c_j = \arg\min_b d_A(x_1^T, r_j^b). \end{aligned} \tag{31}$$

The most direct use of PDM for this classifier is to adjust Λ by using the discriminant function (31). However, (31) is doubly unsmooth: this new discriminant function includes the unsmooth operation of "min" in both the best path search and the closest reference search. Obviously, there is a fundamental difficulty in designing the DTW distance classifier with PDM in its original form.

However, the reason for the difficulty in applying PDM to dynamic pattern classification is the same as the problem included in the formulation of the PDM training method. The fundamental problem to be solved is that the functions used are discontinuous, i.e., not differentiable in Λ. Therefore, we come naturally to the motivation for the development of GPD, which is to reformulate PDM using some smoothing function.

III. Generalized Probabilistic Descent Method

A. Formalization Concept

The fundamental concept of the GPD formalization is to directly embed the overall process of classifying a dynamic pattern of speech acoustic vectors x_1^T in a smooth functional form (at least first order differentiable with respect to classifier parameters) that is suited for the use of a practical optimization method, especially gradient search optimization [52], [54], [56]. As can be easily noticed from the previous discussions, a key point for achieving this formalization is to overcome the discontinuity of PDM. For this purpose, GPD uses the L_p norm form and a sigmoidal function that is widely used in ANN implementations.

Similar to PDM, GPD uses an adaptive update mechanism. However, the above formalization concept can be achieved, without any lack of formal rigor, even using a batch-type optimization such as the steepest descent method. Clearly, the selection of optimization methods is distinct from the formalization of the emulation of the classification decision process.

In the following subsection, we present in detail an embodiment of GPD for the distance classifier, defined by

[2] Note that in this simple case for introductory discussions, we consider only the acoustic model of the classifier, assuming that the classifier has no language model.

(28), for the task used in Section II-D3, i.e., the classification of M class feature vector sequence patterns, each corresponding to an isolated spoken syllable.

B. GPD Embodiment for Distance Classifier

1) Discriminant Function for Dynamic Patterns: The problem with PDM in handling dynamic patterns is shown in (30) and (31), i.e., the discriminant function includes a discontinuous search for the closest reference and for the best matching path. To solve this, GPD defines the discriminant function as

$$g_j(x_1^T;\Lambda) = \left[\sum_{b=1}^{B_j} \left\{D(x_1^T, r_j^b)\right\}^{-\zeta}\right]^{-1/\zeta} \tag{32}$$

where $D(x_1^T, r_j^b)$ is a generalized reference pattern distance between x_1^T and r_j^b, and ζ is a positive constant; also it defines this generalized reference pattern distance as

$$D(x_1^T, r_j^b) = \left[\sum_{\theta=1}^{\Theta_j^b} \left\{D_\theta(x_1^T, r_j^b)\right\}^{-\xi}\right]^{-1/\xi} \tag{33}$$

where $D_\theta(\cdot)$ is the path distance of (29) and ξ is a positive constant. Accordingly, the method achieves a smooth discriminant function for the dynamic pattern by replacing the two "min" operations by the L_p norm form functions.

The use of the L_p norm form affords us an interesting flexibility in the implementation. Let ζ and ξ approach ∞. Then, clearly, (32) approximates the operation of searching for the closest reference pattern and (33) for operation of searching for the best matching path. Thus, these smooth definitions are generalized versions of the existing search operation.

2) Formulation: GPD uses the three-step formalization of PDM in order to represent the task at hand (classification in our case) in a smooth functional form. The classification rule here is

$$C(x_1^T) = C_i, \quad \text{iff } i = \arg\min_j g_j(x_1^T;\Lambda) \tag{34}$$

which is essentially the same as (13), except that the pattern is dynamic.

We already described the first step of defining the discriminant function; the function is given as (32).

The second step defines a smooth misclassification measure. Among many possibilities, the following is a typical definition for x_1^T $(\in C_k)$

$$\begin{aligned} d_k(x_1^T;\Lambda) =& g_k(x_1^T;\Lambda) \\ &- \left[\frac{1}{M-1}\left\{\sum_{j,j\neq k} g_j(x_1^T;\Lambda)\right\}^{-\mu}\right]^{-1/\mu} \end{aligned} \tag{35}$$

where μ is a positive constant. Similar to (15), $d_k() > 0$ indicates a misclassification and $d_k() < 0$ indicates a correct classification. Similar to ζ and ξ, controlling μ enables one to simulate various decision rules. In particular, when μ approaches ∞, (35) resembles (34).

The third step of defining the loss is almost the same as in PDM. GPD defines an individual loss as a smooth, monotonically increasing function of the misclassification

$$\ell_k(x_1^T;\Lambda) = l_k(d_k(x_1^T;\Lambda)). \tag{36}$$

There are various possibilities for defining the loss. Among them, the GPD method usually uses a smooth function as

$$\begin{aligned} \ell_k(x_1^T,\Lambda) &= l_k(d_k(x_1^T;\Lambda)) \\ &= \frac{1}{1+e^{-(\alpha d_k(x_1^T;\Lambda)+\beta)}} \qquad (\alpha > 0) \end{aligned} \tag{37}$$

where α and β are constants. We shall mention the selection of the loss function again later.

The embodiment is completed by deriving an adjustment rule such as (26) according to the probabilistic descent theorem. Given a dynamic input pattern at the design stage time index t, i.e., $x_1^T(t) = (\boldsymbol{z}_1(t), \ldots, \boldsymbol{z}_T(t))$ $(\in C_k)$, the resulting adjustment rule using loss (37) for the reference patterns is given as

$$\boldsymbol{r}_{j,\tau}^b(t+1) = \begin{cases} \boldsymbol{r}_{j,r}^b(t) + 2\epsilon(t)\nu_k\omega_{j,\tau}^b\phi_j\rho_j\varphi_j \\ \qquad (\text{for } j = k), \\ \boldsymbol{r}_{j,\tau}^b(t) - \dfrac{2}{M-1}\,\epsilon(t)\nu_k\omega_{j,\tau}^b\sigma_j\phi_j\rho_j\varphi_j \\ \qquad (\text{for } j \neq k) \end{cases} \tag{38}$$

where

$$\nu_k = \alpha l_k(d_k(x_1^T(t);\Lambda))\{1 - l_k(d_k(x_1^T(t);\Lambda))\} \tag{39}$$

$$\phi_j = \left[\sum_{b'=1}^{B_j}\left\{\frac{D(x_1^T(t), r_j^b)}{D(x_1^T(t), r_j^{b'})}\right\}^{\zeta}\right]^{-(1+\zeta)/\zeta} \tag{40}$$

$$\rho_j = \left[\sum_{\theta=1}^{\Theta_1^b}\left\{D_\theta(x_1^T(t), r_j^b)\right\}^{-\xi}\right]^{-(1+\xi)/\xi} \tag{41}$$

$$\varphi = \sum_{\theta=1}^{\Theta_j^b}\frac{\boldsymbol{z}_{\varpi(j,b,\tau,\theta)}(t) - r_{j,\tau}^b}{\{D_\theta(x_1^T(t), r_j^b)\}^{\xi+1}} \tag{42}$$

$$\sigma_j = \left[\frac{1}{M-1}\sum_{j,j\neq k}^{M}\left\{\frac{g_k(x_1^T(t);\Lambda)}{g_j(x_1^T(t);\Lambda)}\right\}^{\mu}\right]^{-(1+\mu)/\mu} \tag{43}$$

and $\varpi(j,b,\tau,\theta)$ indicates the component vector index of $x_1^T(t)$, to which the τth component vector of $r_j^b, r_{j,\tau}^b$, corresponds along the θth matching path of r_j^b. In (38), the adjustment is done for all of the possible paths and all of the possible reference patterns. This is a remarkable distinction from (26), in which the adjustment is selectively done. Furthermore, treating $w_{j,\tau}^b$'s as adjustable parameters, one can achieve an adjustment rule similar to (38), though we omit the result. See this point in [20].

The rule of (38) is rather complicated. In the last paragraph of the implementation, we present for informational

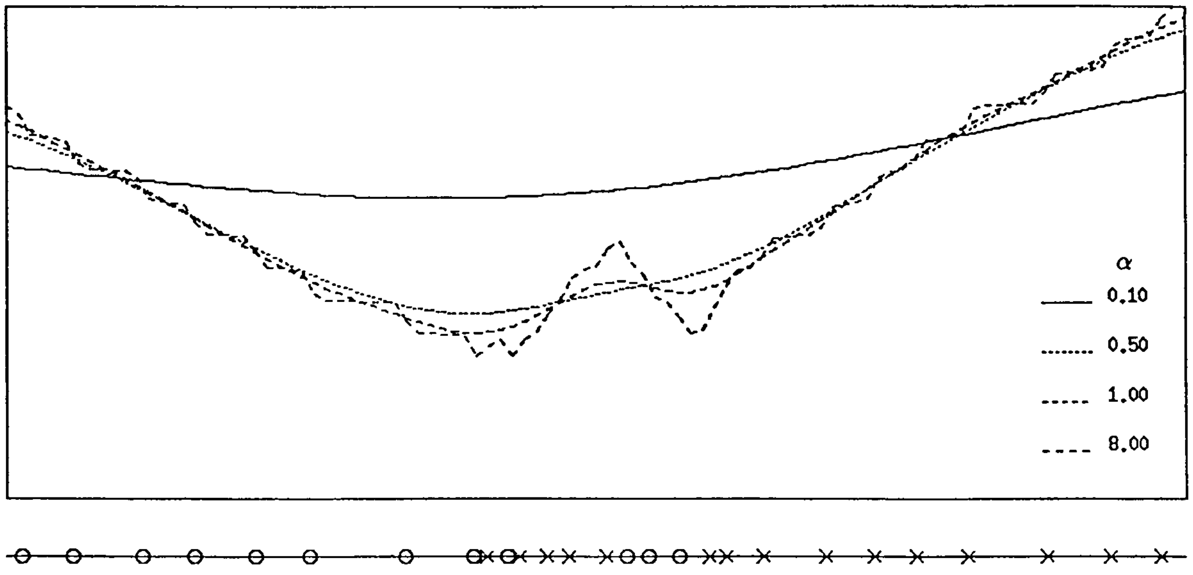


Fig. 3. The relation in smoothness between the smooth classification error count loss and its corresponding empirical average loss.

purposes an extremely simplified version of this rule. That is, letting ζ, ξ, and μ approach ∞, we can rewrite (38) as

$$r^b_{j,\tau}(t+1) = \begin{cases} r^b_{j,\tau}(t) + 2\epsilon(t)\nu_k\omega^b_{j,\tau}\big(z_{1(j,b,\tau)}(t) - r^b_{j,\tau}(t)\big), \\ \qquad \text{(for } j = k \text{ and } b = 1) \\ r^b_{j,\tau}(t) - 2\epsilon(t)\nu_k\omega^b_{j,\tau}\big(z_{1(j,b,\tau)}(t) - r^b_{j,\tau}(t)\big), \\ \qquad \text{(for } j = i \text{ and } b = 1) \\ r^b_{j,\tau}(t), \quad \text{(otherwise)} \end{cases} \tag{44}$$

where $1(j,b,\tau) = \theta(j,b,\tau)|_{\theta=1}$, and C_i is the class having the smallest discriminant function value among classes other than C_k, i.e., the most likely competing class. This simple result weakens the smoothness policy of the GPD formalization, but it increases its feasibility by focusing on the most likely rival class instead of all rival classes. The rule in (44) will be referred to later in Section IV-A.

C. Design Optimality

In a realistic situation where only finite design samples are available, the state of Λ that can be achieved by probabilistic descent training is at most a local optimum over a set of design samples. Morever, in the case of finite learning repetitions, Λ does not necessarily achieve even the local optimum solution. However, GPD addresses this problem in a more advanced way than the previous DFA-based methods.

To discuss this point we consider an illustrative task. The task is to classify one-dimensional static patterns as one of two classes. The distribution of samples is shown on the horizontal axis in Fig. 3. In the same figure, the classification error count, which is a function of the estimated class boundary position, is also shown. This error count is equivalent to the average empirical loss using (10) and is a discontinuous step function. Note that the classification error count graph changes only at the locations of the design samples. Moreover, in Fig. 3, several continuous curves, each corresponding to the average empirical loss using the smooth classification error count loss (37), are shown; the difference among these continuous loss curves relies on the value setting of α in (37). These curves change smoothly, even at positions other than the sample positions, which demonstrates that controlling α in (37) can produce an interesting change in the smoothness of the loss.

As shown in Fig. 3, conventional use of the discontinuous loss does not take into account the sample distribution at locations other than the given sample positions. In contrast, GPD substantially incorporates the region around the given samples in the design process by using the smooth representation [51]. Methods in the Bayes approach usually attempt to solve this sparse sample problem by introducing parametric distribution models. Compared with them, GPD assumes only the continuity of the sample distribution. This modeling is moderate but quite natural. Thus, GPD aims to increase consistency with unknown samples for the estimate of the proper decision boundary simply by making use of the sample distribution's continuity.

Fig. 3 also suggests a way to alleviate the problems of local optima during training; that is, one sets the smoothing larger in the early stage of training and gradually decreases it as training proceeds [59]. Learning is first done in a coarse but global search mode and gradually changes to a fine-adjustment search mode. This is similar to the idea in [42] as well as the simulated annealing global search optimization [35].

Consequently, we can argue that GPD has a larger fundamental possibility of achieving the (global) minimum of the loss than the traditional PDM. This characteristic of GPD can be a useful mathematical framework for advanced analysis, such as for training robustness to unknown samples. It is also worthwhile noting that this characteristic can be applied to the entire recognition problem, as well as for the classification discussed in this section.

D. Minimum Classification Error Learning

The ultimate goal of recognizer/classifier design is to find the parameter set that achieves the condition of minimum error. In this section we summarize the results of [52], which successfully showed that a GPD-formalization based minimization of the expected smooth classification error count loss, i.e., MCE, possesses a fundamental capability for achieving this minimum error condition. Let us indicate here that this discussion holds true in general approaches to minimum-Bayes-risk recognition as well as in classification.

To maintain the mathematical rigor of our discussions for handling dynamic patterns, we begin our summary by providing a new probability measure $p(x_1^T)$ for dynamic patterns. Let us assume that this probability can be computed by an HMM which has been widely used in speech recognition.

First, let us consider an (unrealistic) case where the true functional form of the probability is known. We assume that a parameter set determining the functional form is $\dot{\Lambda}$. We also consider a discriminant function that is computed by the HMM probability

$$g_j\left(x_1^T;\dot{\Lambda}\right) = p_{\dot{\Lambda}}(C_j|x_1^T). \tag{45}$$

The classification rule used here is (2). Then, for example, defining the misclassification measure as

$$\begin{aligned} d_k(x_1^T;\Lambda) = &-g_k(x_1^T;\Lambda) \\ &+ \left[\frac{1}{M-1}\left\{\sum_{j,j\neq k} g_j(x_1^T;\Lambda)\right\}^{\mu}\right]^{1/\mu} \end{aligned} \tag{46}$$

we can rewrite the expected loss, defined by using the smooth classification error count loss (37) in the GPD formalization, as follows:

$$\begin{aligned} L(\dot{\Lambda}) &= \sum_k \int_{\Omega} p(x_1^T, C_k)\ell_k\left(x_1^T;\dot{\Lambda}\right)1(x_1^T \in C_k)\ dx_1^T \\ &\simeq \sum_k \int_{\Omega} p(x_1^T, C_k)1(x_1^T \in C_k) \\ &\qquad \cdot 1\left(p_{\dot{\Lambda}}(C_k|x_1^T) \neq \max_j\ p_{\dot{\Lambda}}(C_k|x_1^T)\right)\ dx_1^T \end{aligned} \tag{47}$$

where Ω is the entire sample space of the dynamic patterns x_1^T's, and it is assumed that $dp(x_1^T, C_k) = p(x_1^T, C_k)\ dx_1^T$. Controlling the smoothness of functions such as L_p norm and the sigmoidal function, we can arbitrarily make $L(\dot{\Lambda})$ closer to the last equation in (47). Note here that we use $\dot{\Lambda}$. Based on this fact, the status of $\dot{\Lambda}$ that corresponds to the minimum of $L(\dot{\Lambda})$ in (47) (which is achieved by adjusting $\dot{\Lambda}$) is clearly equal to the $\dot{\Lambda}^*$ that corresponds to a true probability, or in other words, achieves the maximum *a posteriori* probability condition. Accordingly, the minimum condition of $L(\dot{\Lambda})$ can become arbitrarily close to the minimum classification error probability

$$\mathcal{E} = \sum_k \int_{\Omega_k} p_{\dot{\Lambda}^*}(x_1^T, C_k)1(x_1^T \in C_k)\ dx_1^T \tag{48}$$

where Ω_k is a partial space of Ω that causes a classification error according to the maximum *a posteriori* probability rule, i.e.,

$$\Omega_k = \left\{x_1^T \in \Omega | p_{\dot{\Lambda}^*}(C_k|x_1^T) \neq \max_j\ p_{\dot{\Lambda}^*}(C_k|x_1^T)\right\}. \tag{49}$$

The above result is quite interesting, since it shows that one can achieve the minimum classification error probability situation with DFA. However, the assumption that $\dot{\Lambda}$ is known is obviously unsatisfactory, similar to the criticism of the Bayes approach. Recent results concerning the ANN's approximation capability have provided useful suggestions for studying this inadequacy. In the literature, e.g., [33], it was shown that the three-layer perceptron has the fundamental capability of approximating an arbitrary function. In [40] and [84] it was also shown that the radial-basis function (RBF) network had universal approximation capability. Furthermore, a mixtured Gaussian distribution function was shown to approximate an arbitrary function in the sense of the minimum L_p norm distortion [95]. Based on these results, one could argue that an HMM with sufficient adjustable parameters has the fundamental capability of modeling an [unknown] true probability function. If $L(\Lambda)$ (and $\mathcal{E}$) has a unique minimum, and if Λ and $L(\Lambda)$ (and $\mathcal{E}$) are monotonic to each other, then the minimum corresponds to the case in which $g_j(x_1^T;\Lambda)$ is equal to the true probability function. Thus, it follows that under this assumption, which is softer (more realistic) than that in the preceding paragraph, DFA enables one to fundamentally achieve minimum classification error probability, even if the true parametric form of the probability function is unknown.

The preceding discussions assumed the impractical condition that sufficient design samples and classifier parameters are given, allowing the remarkable results concerning the fundamental capability of DFA to be shown. However, one should notice that MCE shows a high degree of utility and originality in practical circumstances in which only finite resources are available. In fact, the more limited the design resources are, the more distinctive from others the MCE result is: MCE always directly pursues the minimum classification error situation, conditioned by given circumstances, while the Bayes approach aims at estimating the entire probability function and, moreover, the conventional DFA methods aim at minimizing the average empirical loss, which is not necessarily consistent with the classification error count. It may be appropriate in such practical circumstances to treat the probability form discriminant function as a class membership function that expresses the possibility that the input belongs to its corresponding class, instead of as an estimate of probability concerning sample distribution

[83]. This type of understanding would greatly help to extend the application range of MCE.

The above argument is independent of the selection of practical minimization methods, such as the probabilistic update of GPD. Instead, an essential point included therein is the use of the three-step formalization of GPD and the smooth classification error count loss such as (37). This interpretation is the reason why we refer to the result in [52] as MCE and distinguish it from GPD.

Although MCE is a somewhat separate concept from GPD, as shown above, it is quite natural that an implementation of GPD design uses the smooth classification error count loss. Actually, most GPD applications use this smooth classification-oriented loss, and such combined usage is often referred to as an MCE/GPD training, e.g., [58]. In the rest of this section and in Section IV, we use the combined naming, MCE/GPD, so as to make clear the fact that the GPD implementations introduced therein each use a smooth classification error count loss such as (37).

E. Speech Recognition Using MCE/GPD-Trained Distance Classifiers

Let us examine the power of MCE/GPD by summarizing speech pattern classification results produced by the multireference distance classifier represented in (28) [19], [20], [62]. The adjustment rules used were basically the same as (38) and (44). The task was to classify nine-class spoken American E-rhyme letter syllables. In most of the experimental settings of [19], [20], and [62], for a step-by-step analysis only the reference patterns were considered to be adjustable, while the weights $w_{j,\tau}^{b}$'s were fixed to a constant value of one. Results for the training of the weights were reported in [19] and [20]. For simplicity, the smoothing of the sigmoidal error count loss was fixed, and the learning factor $\epsilon(t)$ was set to a finite sequence of small, monotonically decreasing numbers, as in (25), i.e., the training did not use the global optimum search strategy described in Section III-C. Instead, the reference patterns were initialized by using the conventional, modified k-means clustering, which basically relies on the minimum distortion design principle: it was expected that this initialization reduced the risk that the GPD-based gradient search would fall into a local optimum.

The data of this task consisted of the nine-class E-rhyme letter syllables: $\{$b, c, d, e, g, p, t, v, z$\}$. Each sample was recorded over dial-up telephone lines from 100 (50 male and 50 female) untrained speakers. The sampling rate was 6.67 kHz, and each sample was converted to a sequence of 24-dimensional acoustic feature vectors (12 cepstral coefficients and 12 delta-cepstral coefficients) by shifting a time window of 300 samples with a window overlap of 200 samples. Speaking was done in isolated mode. Since all of the samples included the common phoneme $\{\mathrm{e}\}$ and were recorded over telephone lines, this task was rather difficult. Recognition experiments were done in multispeaker mode, i.e., each speaker uttered each of the E-set syllables twice, once for designing and once for testing. Thus, for every class, the design and testing data sets consisted of 100 samples, respectively.

Let us first observe the classification results with the conventional, modified k-means clustering that was used for initialization. The following classification rates are all obtained over the unknown testing data. The comparison in resultant accuracy (classification rates) between the conventional method and MCE/GPD will illustrate the effectiveness of MCE/GPD.

According to [62], the modified k-means clustering results ranged from 55.0%–59.8% in the case of using one reference pattern for every class. MCE/GPD achieved rates ranging from 74.2%–75.4% for the same classifier. In the case of using three reference patterns for every class, the modified k-means clustering resulted in the range of 64.1%–64.9%; with 74.0%–77.2% for MCE/GPD.[3]

References [19] and [20] investigated an implementation somewhat different from (38) and (44), using an exponential-form distance measure and also considering the weights $w_{j,\tau}^{b}$'s to be adjustable. Consequently, in these further experiments the classification accuracy increased to 79.4% with the use of only one reference pattern per class and reached a remarkable 84.4% by using four references for every class.

For comparison, let us summarize the classification results of HMM classifiers, each designed by using the very same acoustic feature extraction module. Each of the HMM systems had a left-to-right, no skip, mixtured Gaussian structure and was designed by the conventional segmental k-means clustering. Its accuracy was 61.7% for the five-state and five-component mixtured Gaussian structure, 66.7% for the ten-state and five-component mixtured Gaussian structure, and 69.0% for the 15-state and five-component mixtured Gaussian structure. These results illustrate that the task was quite difficult and that the achievable classification rate of conventional design methods was 70% at most.

The results clearly demonstrate the high utility of MCE/GPD. In particular, it is worthwhile noting that the method achieved a higher accuracy with the smaller size Λ. For example, the MCE/GPD-designed classifier that used only one reference per class had an error rate about three fourths that of the conventionally designed, 12-times larger classifier, which used 12 references per class (see footnote 3).

F. Remarks

Although we have used the task of classifying dynamic speech patterns as our basis for discussing the GPD formalism, one may notice that the resultant MCE/GPD training rules can be used for the classification of various kinds of (static and dynamic) patterns other than speech sounds. In fact, MCE/GPD applications to signals other than speech are being reported in current literature, e.g., [76], [93], and [107]. However, every class of signals has its own

[3]Additionally, it was reported that a larger size distance classifier having 12 reference patterns for every class produced only 67.6%. For this classifier, no result based on MCE/GPD was reported.

special nature, and there are also a variety of possible task goals, in other words, a variety of recognition criteria (the Bayes decision rule being one among many). Therefore, we should state here that applying GPD to a new task always requires some (usually small) effort to adapt the basic GPD formulations so that they adequately reflect the unique nature of the signals at hand.

IV. Relationships Between MCE/GPD and Others

A. Relation with LVQ

As seen in [61] and [67], several implementations of LVQ have been reported. However, the following fundamental concept of LVQ underlies all of the implementations, that is, only if a given design sample is misclassified and is located near the actual class boundary are the reference patterns of the correct class (of the given sample) and those of some competing classes adjusted so as to increase the possibility of classifying the sample correctly. Moreover, all of them have in common a heuristic window function for restricting the adjustment to the class boundary region.

Recall the simplified MCE/GPD adjustment rule for the distance classifier, i.e., (44). One can easily see that in the circumstances of handling static patterns, this rule is quite similar to an improved version of LVQ, which is characterized as follows [69]. If and only if a given sample is misclassified and is located in a small region (window) near the actual class boundary, then 1) the closest reference (to the sample) of the correct class is pushed closer to the sample and 2) the closest reference of the best (most probable) competing class is pulled farther from the sample; otherwise no adjustment is incurred. Clearly, the adjustment mechanism of this LVQ training is realized in (44). In particular, it is worth noting that the window set in LVQ corresponds to a derivative function of the MCE/GPD's smooth classification error count loss. Consequently, this correspondence proves that in substance, LVQ uses the design objective of classification error count.

In LVQ, the adjustment was generally performed only when a sample was misclassified. In contrast with this, (44), incorporating (37), performs the adjustment when a sample is correctly classified but the corresponding misclassification measure is small (close to zero), i.e., when the certainty of the corresponding classification decision is small. This is a natural result of the MCE/GPD formalization using the smooth decision process. Consequently, this "learning-in-correct" contributes to increasing design robustness [62]. The effect of "learning-in-correct" is also reported in [1], [4], [57], and [97].

The close relation between MCE/GPD and LVQ enables us to use many application results of LVQ as circumstantial evidence of the effectiveness of MCE/GPD [57], [61], [70]. The inadequacy of LVQ, which was reported in the literature, e.g., [48], may be due to a lack of consistency with regard to the design objective (loss) formulation between a given task and the LVQ training. In most cases, LVQ has been used for reducing the misclassification of static patterns, such as the elemental acoustic feature vectors of the dynamic speech pattern to be classified originally. Solving this inconsistency was one of the motivations for the development of MCE/GPD.

B. Relation with Maximization of Mutual Information

As befits the Bayes decision theory, we have considered classification error count loss in this paper. On the other hand, various other objective functions have also been extensively studied for the sake of increasing classification accuracy. Among them, the use of mutual information has attracted much research interest in speech pattern classification [7].

This approach aims to select the state of Λ so as to increase as much as possible the mutual information between class C_j and a sample x_1^T $(\in C_k)$, which is defined in the following:

$$I_k(x_1^T;\Lambda) = \ln \frac{p_\Lambda(x_1^T|C_k)}{\sum_j^M p_\Lambda(x_1^T|C_j)P(C_j)}. \tag{50}$$

Let us consider the effect of maximizing the mutual information in the GPD (instead of MCE/GPD) framework. For convenience, we use a negative value of mutual information. Thus, the goal of GPD design is to minimize this negative measure. The negative mutual information is rewritten as

$$\begin{aligned}
-I_k(x_1^T;\Lambda) &= \ln \frac{\sum_j^M p_\Lambda(x_1^T|C_j)P(C_j)}{p_\Lambda(x_1^T|C_k)} \\
&= \ln \left\{ P(C_k) + \frac{\sum_{j,j\neq k}^M p_\Lambda(x_1^T|C_j)P(C_j)}{p_\Lambda(x_1^T|C_k)} \right\} \\
&\geq \ln \frac{\sum_{j,j\neq k}^M p_\Lambda(x_1^T|C_j)P(C_j)}{p_\Lambda(x_1^T|C_k)} \\
&= -\ln\, p_\Lambda(x_1^T|C_k) \\
&\quad + \ln \left\{ \sum_{j,j\neq k}^M P(C_j)e^{\ln p_\Lambda(x_1^T|C_j)} \right\}.
\end{aligned} \tag{51}$$

Here, defining the logarithmic likelihood $\ln\, p_\Lambda(x_1^T|C_k)$ as the discriminant function, one can treat the bottom line expression of (51) as a kind of misclassification measure

$$d_k(x_1^T;\Lambda) = -g_k(x_1^T;\Lambda) + \ln \left\{ \sum_{j,j\neq k}^M P(C_j)e^{g_j(x_1^T;\Lambda)} \right\}. \tag{52}$$

Then, the inequality

$$-I_k(x_1^T;\Lambda) \geq d_k(x_1^T;\Lambda) \tag{53}$$

holds true, and therefore maximizing the mutual information leads at least to minimizing the misclassification measure (52). Consequently, it turns out that a classifier design based on maximizing mutual information has the same effect as a GPD design that uses the misclassification measure (52) and the linear loss function. Note here that the loss used is not a smoothed error count but a simple linear function of the misclassification measure. Obviously, this method is not guaranteed to be consistent with the minimum classification error condition unless a $0-1$ nonlinear function is imposed.

C. Relation with MSE Loss

Most ANN classifiers have used the squared error between the teaching signal and the classifier output (the value of the discriminant function) as the loss; the design employed therein aims to minimize the expected loss or the average empirical loss, computed by using this individual squared error loss. Similar to the mutual information maximization method, let us consider the nature of this squared error loss minimization method in the GPD framework. First, the squared error loss is represented as

$$\ell_k(x_1^T;\Lambda) = \frac{1}{2}\sum_{j}^{M}\left\{g_j(x_1^T;\Lambda) - \varepsilon_j\right\}^2 \tag{54}$$

where $\{\varepsilon_j\}$ is a teaching signal which is usually set, for a design sample x_1^T $(\in C_k)$, to

$$\varepsilon_j = \begin{cases} 1, & (j=k) \\ 0, & (\text{otherwise}). \end{cases} \tag{55}$$

The loss is thus rewritten as

$$\begin{aligned}\ell_k(x_1^T;\Lambda) &= -g_k(x_1^T;\Lambda) + \tfrac{1}{2}\left\{(g_k(x_1^T;\Lambda))\right\}^2 \\ &\quad + \tfrac{1}{2} + \tfrac{1}{2}\sum_{j,j\neq k}^{M}\left\{(g_j(x_1^T;\Lambda))\right\}^2 \\ &> -g_k(x_1^T;\Lambda) + \tfrac{1}{2}\sum_{j,j\neq k}^{M}\left\{(g_j(x_1^T;\Lambda))\right\}^2.\end{aligned} \tag{56}$$

Then, we can treat the bottom line expression of (56) as a kind of misclassification measure

$$d_k(x_1^T;\Lambda) = -g_k(x_1^T;\Lambda) + \frac{1}{2}\sum_{j,j\neq k}^{M}\left\{(g_j(x_1^T;\Lambda))\right\}^2. \tag{57}$$

It turns out that the reduction of the squared error loss results in the reduction of this misclassification measure. Hence, we can conclude that the MSE leads at least to an optimal situation, in the GPD sense, which is based on the linear loss using the misclassification measure (57). Note here again that the loss used is not a smoothed classification error count but a simple linear function of the misclassification measure. Obviously, it is not guaranteed that the resulting status of this method is consistent with that of the minimum classification error condition.

Equation (57) is difficult to use as the misclassification measure. Since the second term (of the right-hand side) of (57), i.e., the average discriminant function of the competing classes, is not normalized by the number of possible classes (M), the design result using this measure is fundamentally affected by the number of classes. Furthermore, since the discriminant function of C_k is compared with the squared values of the other competing class discriminant functions, the scalar decision based on (57) does not emulate the classification appropriately.

V. Derivatives of GPD

A. Overview

The key concept of GPD formalization is to emulate the overall process of a task at hand in a tractable functional form. Actually, the implementation of MCE/GPD presented in Section III properly realized this concept for the task of classifying a dynamic speech pattern. However, in the example task, the pattern to be classified was *a priori* extracted from its surrounding input signal and was represented in the preset form of an acoustic feature vector sequence. The task defined therein is one simplified and limited case among many. Clearly, the design concept of GPD should be applied directly to more complicated and realistic task situations. For example, the concept of GPD development should be applied to either a connected word recognition task or an open-vocabulary speech recognition task. It should also be applied to the entire process of recognition, which includes the spotting (detection) of target speech sounds, the design of the feature extraction parameters, and the design of the language model. In this light GPD has actually been quite extensively studied, and its original formalization for classification has been dramatically extended to a new family of discriminative design methods which are more suitable for handling complex real-world tasks. In this section, we shall summarize several important members of the GPD family, such as segmental GPD, minimum spotting error learning (MSPE) [63], discriminative utterance verification [89], discriminative feature extraction (DFE), e.g., [15], [58], and also introduce application examples of GPD to speaker recognition [66].

B. Segmental-GPD for Continuous Speech Recognition

Most definitions used in the simple example task of classifying spoken syllables, in Sections II and III, can be applied to a more realistic task of classifying continuous speech utterances. The most important issue in this advanced application is still to embed the entire process of classification in an appropriate GPD-based functional form. For example, for a connected word recognition task, a discriminant function should be defined so as to directly measure the degree to which a connected word sample belongs to its corresponding class. Due to resource restrictions, most continuous speech recognizers employ subword models, such as phonemes, as the basic unit of acoustic modeling, and represent words and connected words by concatenating the subword models. The classification process in this task consequently includes the mechanism of dividing a long input of continuous speech into short subword segments. Also, due to the need for tackling the statistical variation of speech sounds, most present continuous speech recognizers use HMM-based acoustic models. Efforts of applying GPD to continuous speech recognition must appropriately take into account these two requirements, i.e. the segmentation mechanism and the probabilistic modeling. Indeed, the development of segmental GPD was motivated by these concerns [24], [53].

Assume the use of a modular connected word recognizer that consists of a feature extraction module and a classification module. Then consider $x_1^T = (\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_T)$ to be speech input to the classifier, where $\boldsymbol{z}_t$ is an acoustic feature vector (observation) at time index t. Also let $W = (w_1, w_2, \ldots, w_S)$ be a word sequence that usually constitutes a sentence. Generally, an HMM classifier for x_1^T is defined by using a first-order S-state Markov chain governed by the following manifolds:

1) a state transition probability matrix $A = [a_{\iota\kappa}]$, where $a_{\iota\kappa}$ is the probability of making a transition from state ι to state κ;
2) an initial state probability $\pi_\iota = P(q_0 = \iota)$, which specifies the state of the system q_0 at time index $t = 0$;
3) an observation emission probability according to a distribution $b_{q_t}(\boldsymbol{z}_t) = P(\boldsymbol{z}_t|q_t), q_t = 1, 2, \ldots, S$, which is usually defined with a mixtured Gaussian distribution.

Accordingly, assuming that Λ is a set of the above probability parameters, i.e., the state transition probabilities, the initial state probabilities, and the observation emission probabilities, the discriminant function for x_1^T is given as the following likelihood measure:

$$g_{W_r}(x_1^T; \Lambda) = \log\ P(x_1^T, \boldsymbol{q}_{W_r}, W_r|\Lambda) \tag{58}$$

where

$$\begin{aligned} W_r =& \arg\max_{W \neq W_1, \ldots, W_{r-1}} P(x_1^T, \boldsymbol{q}_{W_r}, W_r|\Lambda) \\ =& r\text{th best word sequence} \\ \boldsymbol{q}_{W_r} =& \text{best state sequence according to } W_r \end{aligned} \tag{59}$$

and $P(x_1^T, \boldsymbol{q}_{W_r}, W_r|\Lambda)$ is the joint state-word sequence likelihood.

The goal of training is to find an optimal Λ that leads to accurate classification of word sequences. Given $x_1^T \in W_k$ for training, segmental GPD defines the misclassification measure as

$$\begin{aligned} d_{W_r}(x_1^T; \Lambda) =& -g_{W_k}(x_1^T; \Lambda) \\ &+ \log \left\{ \frac{1}{r_m} \sum_{r=1}^{r_m} e^{g_{W_r}(x_1^T; \Lambda) \cdot \eta} \right\}^{1/\eta} \end{aligned} \tag{60}$$

where r_m is the total number of the competing word sequences, different from W_k, that will be taken into consideration in training. The training procedure of segmental GPD is then completed by embedding this misclassification measure in the smooth error count loss, as in Section III. Note here that by setting r_m to a small number in (60), in other words, by focusing only on a limited number of likely rival classes, one can make the adjustment more simple.

Clearly, the Markov modeling satisfies the above two requirements for continuous speech recognition: it performs the segmentation process with its state transition mechanism, and it also provides a probabilistic representation framework. A point to note here is that an implementation with probabilistic models should maintain the probabilistic constraint of the model parameters, i.e., the sum of the probabilistic parameters, such as the state transition probability, over all the possible cases should be equal to one. Segmental-GPD thus uses parameter transformation as

$$a_{ij} \rightarrow \tilde{a}_{ij} \quad \text{where } a_{ij} = \frac{e^{\tilde{a}_{ij}}}{\sum_m e^{\tilde{a}_{im}}} \tag{61}$$

in the parameter adaptation.

The power of segmental GPD has been demonstrated in several experimental tasks. For example, [24] reported quite successful recognition results in the nine-class American English E-ryhme task, used in Section III, i.e., an HMM recognizer with ten-state, five-component mixtured Gaussian(/state) models scored 99% over the training data and 88% over the testing data. For the connected word case, [25] and [26] reported results in the TI connected-digit recognition task. In particular, in [26] the best results reported so far on the database (a string error rate of 0.72% and a word error rate of 0.24% on testing data) were successfully achieved by a segmental GPD-trained, context-dependent subword model system.

The same design concept employed in segmental GPD, i.e., optimizing to increase recognition accuracy for concatenated word inputs, has been tested in several slightly different ways and has further demonstrated its high utility [72], [73], [87].

C. Minimum Error Training for Open-Vocabulary Speech Recognition

1) Open-Vocabulary Speech Recognition: Spontaneous conversation utterances usually contain speech segments that are not directly relevant to the tasks at hand, such as false-starts, interjections, and repairs, and they also suffer from various acoustic problems such as increased coarticulation. Many large-vocabulary continuous-speech recognizers have been successfully developed for dictation-style (recognition of read speech) tasks, but the recognition of such casual conversational utterances is still an important research issue [108]. Actually, it is costly to fully model the highly variable acoustic phenomena of large-vocabulary conversational utterances. Therefore, there has been increasingly more interest recently in a practical alternative, i.e., an open-vocabulary paradigm where only selected keywords are to be recognized. Fig. 4 illustrates the mechanism of open-vocabulary speech recognition.

There are two main approaches to open-vocabulary recognition [63], [90]. The first is keyword spotting based

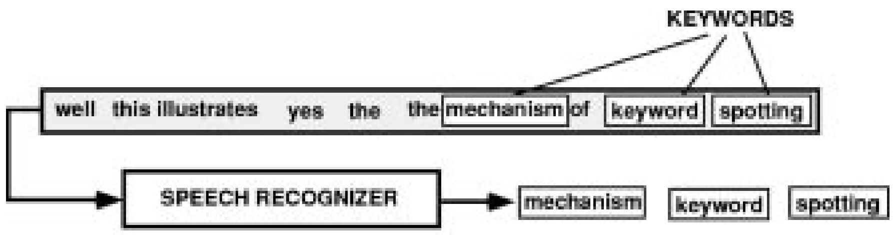

KEYWORDS
well this illustrates yes the the mechanism of keyword spotting
SPEECH RECOGNIZER
mechanism
keyword
spotting

Fig. 4. Concept of open-vocabulary speech recognition.

on a threshold comparison. A system prepares keyword models, computes the similarity between a segment of input utterance and each keyword model, and decides whether the segment contains the keyword or not by comparing a similarity value and a preset threshold. The second is to use a continuous-speech recognizer having a "filler" model. In this case, the recognizer attempts to segment and classify an input utterance in the same manner as conventional speech recognizers for isolated or concatenated spoken words do, but here a nonkeyword segment is classified as the class corresponding to the filler model. Note also that here the filler model can fundamentally cover an infinite number of nonkeyword segments, and accordingly, a continuous-speech recognizer with this filler model can run as an open-vocabulary system. There are clear differences between the two approaches, especially in terms of segmentation strategy and implementation approach. However, one can also note that the approaches still have a close link to each other because the similarity comparison between the filler class and keyword classes works as a kind of thresholding operation. Evaluation of these approaches is one of the important on-going research issues in the speech recognition field.

2) Minimum Spotting Error Learning: Depending upon the decision strategy used for spotting, one can clearly note that spotting performance relies heavily upon the adequacy of the model and the threshold. Conventionally, the model has been designed using ML training, as is done for standard continuous speech recognition, and the threshold has been simply determined in a trial-and-error fashion. Obviously, such an unintegrated design method does not guarantee optimal spotting accuracy. One natural solution to this problem is to design both the model and the threshold jointly in the sense of minimizing spotting errors. In this light, GPD was reformulated as MSPE for keyword spotting in [63].

Since spotting can be done in keyword-by-keyword mode, or in other words independently for each keyword class, in the algorithm described here we will consider a simple task with only one keyword class and thus a spotter (spotting system) that contains a single keyword model λ (such as reference patterns or HMM's) and its threshold h. The goal of MSPE design is to optimize $\Lambda = \{\lambda, h\}$. For clarity of description, we assume λ to be a reference-pattern-based keyword model, i.e., a sequence of reference patterns, each defined in the acoustic feature vector space. A spotting decision is fundamentally done at every time index. Therefore, a discriminant function $g_t(x_1^T, S_t; \lambda)$ is

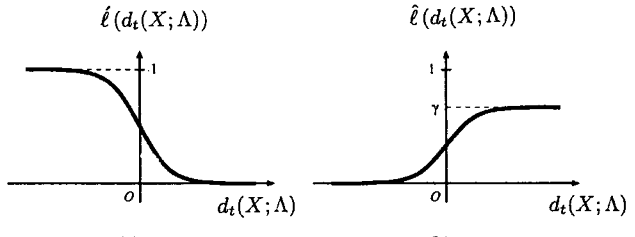


Fig. 5. An example of loss functions for keyword spotting: (a) for false-detection and (b) for false-alarm (after [52]).

defined as a function that measures a distance between a selected segment S_t of input utterance x_1^T and the model λ, and the spotting decision rule is formalized as: if the discriminant function meets

$$g_t\left(x_1^T, S_t; \lambda\right) < h \tag{62}$$

then the spotter judges at "t" that a keyword exists in the segment S_t; no keyword is spotted otherwise. Importantly, this type of decision could produce in principle two types of spotting errors: false-detection (the spotter decides that S_t does not include the keyword when S_t actually includes.) and false-alarm (the spotter decides that S_t includes the keyword even when S_t does not include.).

Similar to original GPD, the MSPE formalization aims to embed the above-cited spotting decision process in an optimizable functional form and to provide a concrete algorithm of optimization so that one can consequently reduce the spotting errors. The spotting decision process is then emulated as spotting measure $d_t(X;\Lambda)$ which is defined as

$$d_t(X;\Lambda) = h - \ln\left\{\frac{1}{|I_t|}\sum_{\varsigma\in I_t}\exp\left(-\xi g_\varsigma(X;\Lambda)\right)\right\}^{-1/\xi} \tag{63}$$

where I_t is a short segment which is set for increasing the reliability and stability of the decision around the time position, t,$|I_t|$ is the size of I_t (the number of acoustic feature frames in I_t), and ξ is a positive constant. A positive value of $d_t\ (X;\Lambda)$ implies that at least one keyword exists in I_t and a negative value implies that no keyword exists in I_t. Decision results are evaluated by using a loss function that is defined by using two types of smoothed $0-1$ functions, illustrated in Fig. 5, as

$$\ell_t(X;\Lambda) = \begin{cases} \acute{\ell}(d_t(X;\Lambda)), \\ \qquad \text{if } I_t \text{ includes a training keyword} \\ \gamma\hat{\ell}(d_t(X;\Lambda)), \\ \qquad \text{if } I_t \text{ includes no training keyword.} \end{cases} \tag{64}$$

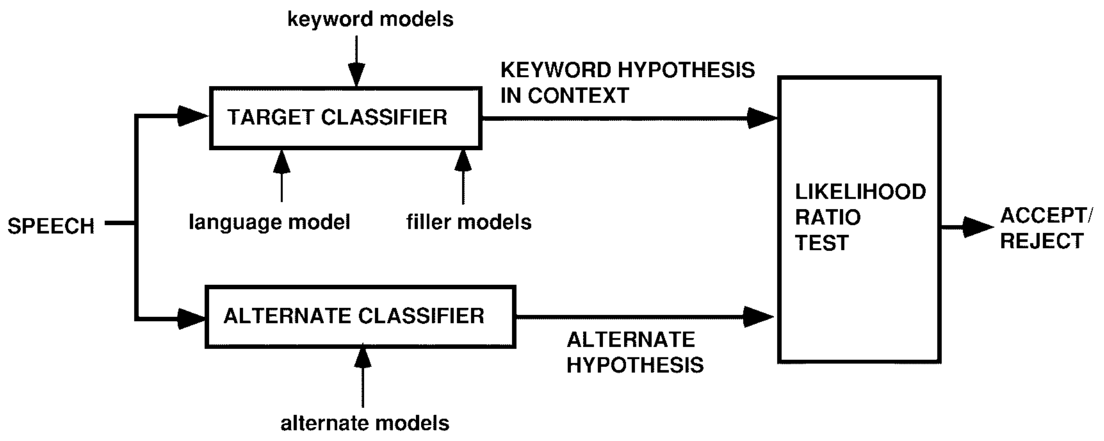


Fig. 6. A speech recognizer with the capability for verifying the word hypotheses produced by a continuous speech recognizer (after [74] with some simplification).

$\ell_t(\cdot)$ approximates: 1) one for one false-detection [the left side of $\acute{\ell}$, Fig. 5(a)]; 2) γ for one false-alarm [the right side of $\gamma\hat{\ell}$, Fig. 5(b)]; and 3) zero for correct spotting (the right side of $\acute{\ell}$ and the left side of $\gamma\hat{\ell}$), where γ is a parameter controlling the characteristics of the spotting decision. By setting γ to one, all of the errors can be treated evenly. By letting γ be less than one, the system can be tuned toward reduction of false-detection errors, which are often more troublesome than false-alarm errors in many application situations.

The training procedure of MSPE is accordingly obtained by applying the adjustment rule (24) of the probabilistic descent theorem to the above-defined functions, i.e., the loss, the spotting measure, and the discriminant function. In [63], readers can find its promising aspects, demonstrated in a task of spotting Japanese consonants.

Even in the open-vocabulary situation, a system is often required to spot a sequence (or set) of keywords instead of only one single keyword. In order to meet this requirement, MSPE was reformulated as the minimum error classification of keyword-sequences method (MECK). See [64] for details.

3) Discriminative Utterance Verification: One can easily notice that the keyword spotting procedure is considered a hypothesis testing problem, based especially on Neyman–Pearson testing and a likelihood ratio test (LRT). The design concept of GPD was also used to increase the LRT-based keyword spotting capability of a continuous speech recognizer employing a filler model.

Fig. 6 illustrates a recognizer used for the above-cited spotting purpose. The system consists of two subclassifiers: 1) the target classifier and 2) the alternate classifier. The target classifier labels an input as a sequence of the hypothesized keyword and the out-of-vocabulary filler segments. The alternate classifier then generates an alternate hypotheses for the LRT test at the segment where the keyword is hypothesized by the target classifier. An LRT tests the hypothesis that an input x_1^T is generated by the model λ_C corresponding to the keyword W_C versus x_1^T, having been generated by an imposter model λ_I corresponding to the alternate hypothesis W_I, according to the likelihood ratio $p(x_1^T|\lambda_C)/p(x_1^T|\lambda_I)$. Accordingly, if the ratio exceeds a preset threshold, the system accepts the hypothesized keyword; otherwise it rejects the hypothesis.

Obviously, the quality of hypothesis testing relies on the adequacy of the estimated likelihood values. However, similar to most cases of the ML-based speech recognition, it is quite difficult to achieve accurate estimates due to the fact that both the probability density functions and their parameterization forms are often unknown. To alleviate this problem, GPD was used as a design algorithm for directly minimizing the false-detection errors and the false-alarm errors [89]. Fig. 7 illustrates the GPD-based training scheme of the keyword hypothesis verification, i.e., discriminative utterance verification (DUV). The training first defines a distance based on the likelihood ratio as

$$d_C(x_1^T;\Lambda) = \log\, p(x_1^T|\lambda_I) - \log\, p(x_1^T|\lambda_C). \tag{65}$$

One should notice here that a positive value of the distance implies hypothesis rejection and a negative value implies hypothesis acceptance. Thus, a loss is defined so as to embed the two types of errors, i.e., the false-detection and the false-alarm, directly as

$$\begin{aligned}\ell(x_1^T;\Lambda) =& \ell^*(d_C(x_1^T;\Lambda))1(x_1^T \in W_C)\\ &+ \ell^*(-d_C(x_1^T;\Lambda))1(x_1^T \in W_I)\end{aligned} \tag{66}$$

where ℓ^* is a smooth monotonic $0-1$ function. Clearly, a training target here is to minimize the loss over possible design samples in the same manner as the other GPD training cases. Refer to [89] for details.

D. DFE

1) Fundamentals: In the previous sections, we have considered only the GPD applications to the postend decision modules, such as the classification (or identification) module, the spotting decision module, and the verification module. Actually, several more advanced applications of GPD have been reported where the GPD training was applied to both the acoustic modeling and the language modeling [45], [96], and the training scope was extended to the front-end feature extraction stage [12], [13], [15]. In

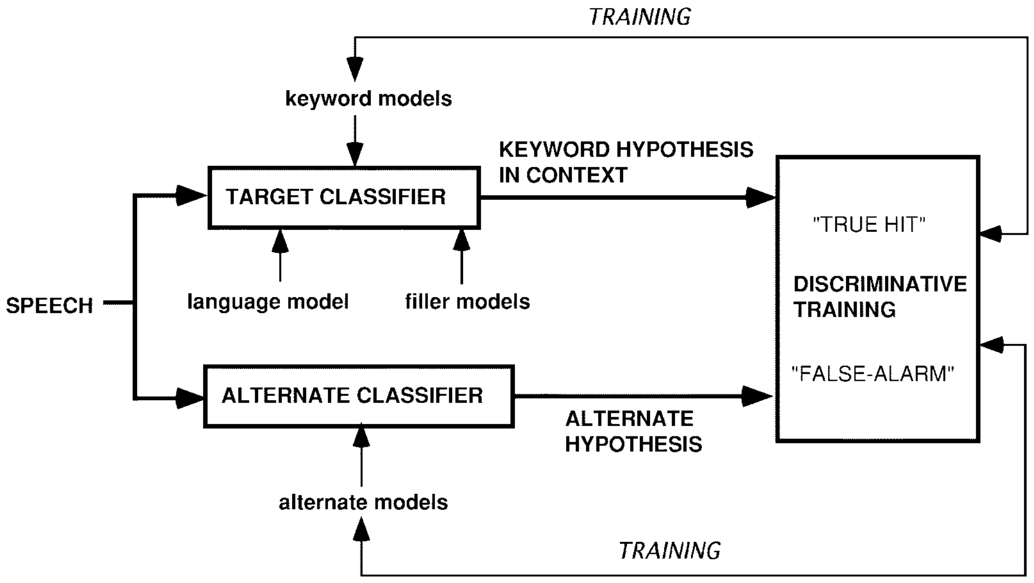


Fig. 7. Block diagram illustrating a GPD-based discriminative training for a speech recognizer verifying hypothesized vocabulary words (after [74] with some simplification).

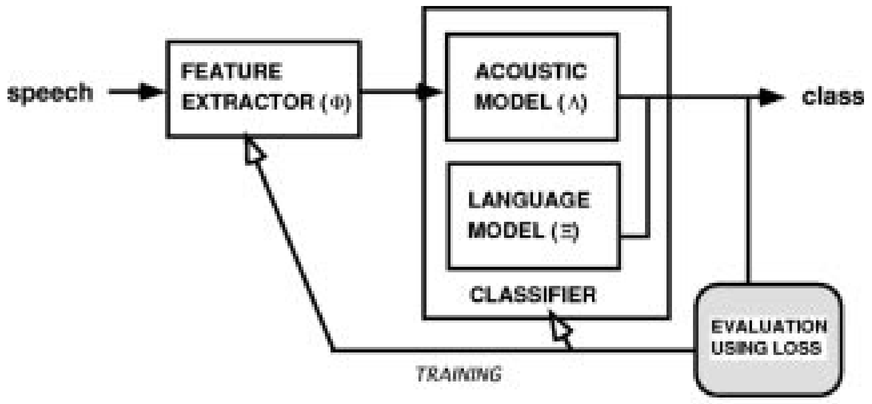


Fig. 8. Training strategy of discriminative feature extraction.

particular, in recent years, GPD-based overall design of the feature extraction and classification modules has attracted the attention of speech researchers who recognize the importance of feature space design (on which the quality of the post-end classification decision indispensably relies), and who also seek solutions to unsatisfactory aspects of current feature extractor design techniques. DFE, which is one of the most straightforward embodiments of GPD's concept of global optimization, is becoming a novel paradigm for design algorithm study [14], [22], [23], [27], [30], [41], [82], [86].

The formalization of DFE is provided by simply replacing the classification rule (3) with the following recognition rule for an input instantiation $u_1^{T_0}$ in the GPD paradigm:

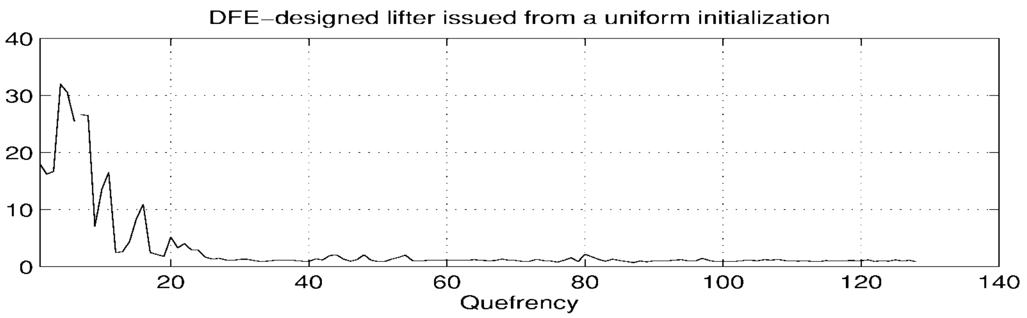
$$C\left(u_1^{T_0}\right) = C_i, \quad \text{iff } i = \arg\max_j \, g_j\left(\mathcal{T}_\Phi\left(u_1^{T_0}\right); \Lambda\right) \tag{67}$$

where $\mathcal{T}_\Phi()$ is a feature extraction operation with the designable parameters Φ. One should notice that the feature extraction process is embedded in the discriminant function, and that by assuming the feature extraction process to be differentiable in Φ, both the feature extractor parameters Φ and the classifier parameters Λ now become the targets of the GPD optimization. Λ is trained in the very same manner as that of GPD training for classification, and Φ is trained by using the chain rule of differential calculus that is additionally used for back-propagating the derivative of the loss to the feature extraction module. Fig. 8 illustrates the DFE training strategy. One should clearly notice here that the DFE design jointly optimizes the overall recognizer with the single objective of minimizing recognition errors.

In (67) we used the speech wave sample $u_1^{T_0}$. However, one can use any reasonable form of input which may be interpreted as an intermediate feature representation of the original utterance input, such as a sequence of FFT-computed power spectrum vectors. In such cases, the DFE training scope is slightly limited but the training can be performed without any degradation of formalization rigor.

In the DFE framework, any reasonable differentiable (in Φ) function, e.g., linear transformation [14], MLP-based

(a)



(b)

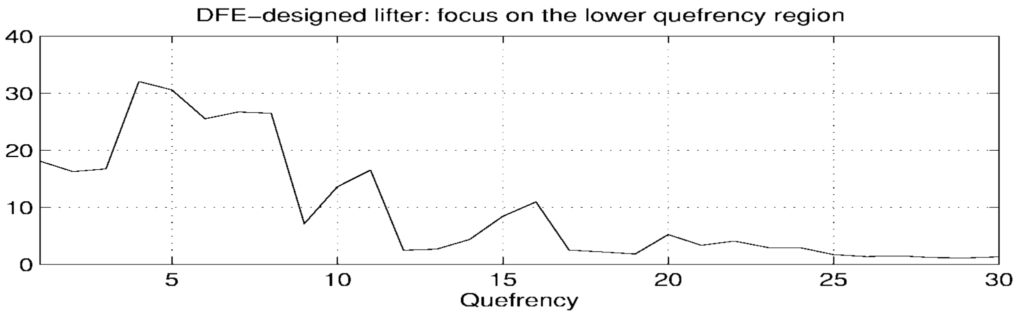


Fig. 9. A typical DPE-trained lifter shape: (a) the entire view issued from the uniform initialization that set the lifter values at a constant before DFE training and (b) the same lifter with a focus on the lower quefrency region (after [11]).

nonlinear transformation, or filtering [82], can be applied to the design of the feature extraction module. In the rest of this section we particularly focus on one example of a direct DFE application, in which a linear transformation called liftering is used, and on two [slightly advanced] topics related to DFE training.

2) An Example Implementation for Cepstrum-Based Speech Recognition: Among many implementation possibilities, [15] studies DFE applied to the design of a cepstrum-based speech recognizer. The cepstrum is one of the most widely used parameter selections for the acoustic feature vector. It has been shown that its low-quefrency components contain salient information for speech recognition, and various lifter shapes have been investigated with the aim of more accurate recognition. However, since there were no methods to design the lifter shape under the criterion of minimizing recognition errors, the shape has usually been determined in an trial-and-error fashion. Obviously, such conventional lifter shapes are not guaranteed to be optimal. An expectation of DFE here is therefore that DFE training can lead to more accurate recognition through the realization of an optimal design for both the lifter feature extractor and the postend classifier.

Fig. 9 shows a typical DFE-trained lifter shape, which was obtained in the Japanese five-vowel pattern-recognition task [15]. One can find that this lifter de-emphasizes: 1) the high-quefrency region that corresponds to pitch harmonics and minute spectral structure and 2) the lower quefrency region (0–2 frequency region) that is dominated by the bias and slant of the overall spectrum while enhancing the region of 3–20 quefrency that corresponds primarily to the formant structure. The observed shape suggests that the DFE training successfully extracted the salient features for vowel pattern recognition. The training actually achieved a clear error reduction, from 14.5% (best by a baseline system) to 11.3% (DFE-trained system), over unknown testing data using the very same size of recognizer.

3) Discriminative Metric Design: Feature extraction can be viewed as a process that forms a metric for measuring the class membership of an input pattern. In general, however, the definition of class identity can be different from class to class. Thus, the feature representation framework, or metric, does not have to be common to all of the classes, i.e. each class could have its own metric that is suitable for representing its identity as effectively as possible. From this point of view, DFE, whose formalization originally assumed a feature extractor common to all sample classes, is not quite sufficient. Motivated by this point, DFE was extended to the discriminative metric design method (DMD) by using the subspace method (SM) paradigm [103].

A recognition decision rule of the DMD framework is as follows:

$$C\left(u_1^{T_0}\right) = C_i, \quad \text{iff } i = \arg\max_j g_j\left(\mathcal{T}_\Phi^j\left(u_1^{T_0}\right); \Lambda\right) \tag{68}$$

where $\mathcal{T}_\Phi^j(\,)$ is a class-specific feature extractor for C_j. Note that for each class the feature extractor is embedded in its corresponding discriminant function, i.e., $g_j(\cdot)$. Similar to the DFE formalization, DMD has many implementation possibilities. Among them, [103] in particular studied implementation using a quadratic discriminant function, which is shown to be closely related to MCE/GPD training for Gaussian kernel functions and accordingly to recent MCE/GPD applications in HMM speech recognizers.

Readers are referred to [103] and [105] for details.

4) Minimum Error Learning SM: Recognition-oriented feature design has been most extensively studied in the SM paradigm, especially for problems in character and image recognition [81]. SM originated from the similarity method where the recognition (classification) decision was made by measuring the angle between a pattern (vector) to be recognized and a class model for every class. In particular, a recognizer based on this method is defined by (lower dimensional) class subspaces, each assumed to represent the salient features of its corresponding class. There is no distinction between the feature extraction process and the classification process in this system framework. Given an input sample, the recognizer computes the orthogonal projection of the input onto each class subspace and then classifies the pattern to the class giving the maximum projection value.

The performance of SM-based recognizers relies on the quality of the class subspaces. The most fundamental algorithms for designing the subspaces were the class-featuring information compression (CLAFIC) [102] and the multiple similarity method (MSM) [46], where each class subspace was designed by running the Karhunen–Loéve Transformation or principal component analysis over the design data of its corresponding class. Obviously, this class-by-class design does not directly guarantee recognition error reduction; the recognition error of design samples are not reflected in the subspace design. These methods were later improved, aiming at increasing recognition accuracy. The CLAFIC was extended to the learning subspace method (LSM) (e.g., [81]); the MSM was reformed as the compound similarity method [46]. In these new versions, subspaces are trained iteratively (adaptively) depending upon the recognition result of each design sample; i.e., when an input design pattern is misrecognized, the subspace of the true class and that of the most likely but incorrect class are adjusted so that projection onto the true class subspace increases and projection onto the competing incorrect class subspace decreases. This discriminative iteration actually contributed to improvements in accuracy. However, the training mechanism had only an intuitive validity, and its mathematical optimality in terms of error minimization has long remained unclear.

To alleviate the above problem, [104] studied a DFE formalization for the SM framework and proposed the minimum error learning subspace (MELS) method that guarantees the optimal subspace design in the MCE/GPD sense. Detailed discussions, including the training convergence proof, are included in [104] and [106].

E. Speaker Recognition Using GPD

Speaker recognition is usually classified into two major categories, i.e., speaker identification, which is the process of identifying an unknown speaker from a known population, and speaker verification, which is the process of verifying the identity of a claimed speaker from a known population. Given a test utterance x_1^T, which is assumed to be represented as a sequence of feature vectors, the likelihood of observing x_1^T being generated by speaker k is denoted as $g_k(x_1^T;\Lambda)$, where Λ is a set of trainable recognizer parameters. Then, the decision operation of speaker identification is formalized as

$$\hat{k} = \arg\max_k g_k(x_1^T;\Lambda) \tag{69}$$

with $\hat{k}$ being the identified speaker, attaining the highest likelihood score among all competing speakers; the decision operation of speaker verification is formalized as: if the likelihood meets

$$g_k(x_1^T;\Lambda) > h_k \tag{70}$$

then the recognizer accepts the claimed speaker identity k; the recognizer rejects it otherwise. One can easily notice here that the formalization for speaker identification is equivalent to that for speech recognition and also that the formalization for speaker verification is essentially the same as that for keyword spotting. One may also notice that the recognizer training here incurs the same difficulty in the likelihood estimation as does the conventional training of continuous speech recognizers and keyword spotters. To alleviate the problem, in [66] GPD training was successfully applied to the design of both an HMM-based speaker identification system and an HMM-based speaker verification system. The derivation of GPD-based training is fundamentally the same as that for speech recognition. Note that whereas GPD training was applied to threshold training in MECK [63], the threshold h_k was determined experimentally in [66]. Refer to [66] for details of the implementation and experimental results.

For speaker verification, [66] also proposed the use of a GPD-trained normalized likelihood score. The decision rule in this case is formalized as: if the likelihood meets

$$\log \frac{g_k(x_1^T;\Lambda)}{g_{k'}(x_1^T;\Lambda)} > h_k \tag{71}$$

then the recognizer accepts the claimed speaker identity k; the recognizer rejects it otherwise, where k is the claimed speaker and k' is the antispeaker (the set of speakers other than k). Clearly, the likelihood ratio of (71) is quite similar to the misclassification measure of GPD, and one can naturally expect that the discriminative power of the normalized score is increased by GPD training. In [66], a remarkable improvement due to GPD training was actually demonstrated.

To see other GPD applications to the task of speaker recognition, readers are referred to recent literature such as [31], [68], [94], and [98].

VI. Concluding Remarks

A. Recent Progress of DFA-Based Speech Recognition

In this paper, we have reviewed a recent approach to pattern recognizer design, based on the GPD method. Since our main purpose is to introduce this recent design approach comprehensively, we have focused our discussions on issues rather closely related to the GPD family of methods.

This type of close focus may give readers a somewhat biased understanding, however. Thus, for the purpose of being informative and to avoid misunderstanding, we provide a brief review on the current situation of discriminative training methods, other than GPD, in the speech recognition field.

As cited in Section I, the development of GPD was motivated by the insufficiency of DFA-based speech pattern recognition in the years of around 1990 and the classical formulation of PDM. However, in parallel with the progress of the family of GPD methods, several other DFA-based methods, such as hybrid HMM/ANN methods (e.g., [77]), have also been greatly extended in this decade.

We showed, in Section IV, that the MSE loss is not directly tied to the minimization of classification errors. On the other hand, it has been shown that a discriminant function based on this minimization method can be an estimate of the corresponding class *a posteriori* probability function in the statistical sense (e.g., see [6] and [36]). Actually, several hybrid HMM/ANN speech recognizers, in which the ANN works as an *a posteriori* probability estimator, have successfully utilized the discriminative power of MSE-based training in advanced frameworks suited for making classification decisions for the entire speech input instead of its elemental acoustic feature vectors (e.g., [17] and [77]).

Concerning MSE-based training, it seems that there is still some gap between the theoretical criticisms being made and some practical successes; actually, it has also been pointed out that fundamentally, the MSE-based estimation of *a posteriori* probabilities is not accurate in the class boundary region that is usually represented by small values in the sample distribution function [29], but which is important for classification. This point may be a good future research issue.

Efforts to integrate classification decision results, one for every acoustic feature vector, into an overall classification result for an entire speech input have also been made for the framework of time-delay neural network (TDNN) [101]. That is, TDNN has been successfully extended to multiple-state TDNN (MSTDNN), in which a loss function is defined over the entire input and explicitly reflects the classification decision at the level of the final category outputs of the problem [37], [38]. In the MSTDNN framework, most parts of the network classifier are adjusted under a single training objective, defined for the entire speech input, i.e., the MSE objective in [37] and the maximization of mutual information (MMI) objective in [38].

Moreover, a similar effort has been made for another type of ANN/HMM hybrid [11], in which the MMI objective is used for training. Clearly, the global design scope of these advanced methods, which attempts to reflect the classification decision of the entire speech input in the training step, is in the same spirit as that of the GPD formalism, though the MMI criterion used therein is different from the MCE training target.

It may be worthwhile to mention the fact that the MMI-based discriminative training has attracted research interest in recent years, though its mathematical rigor is not sufficiently guaranteed (e.g., see Section IV of this paper and [77]). Among the many accessible reports, [88] specially evaluates MMI-based training for recurrent networks, and [80] and [100] do the same for HMM classifiers.

In this decade, there has been remarkable progress not only in research on DFA-based design algorithms (e.g., the one based on GPD methods), but also on speech recognition technology in general. A survey of this advancement would be beyond the scope of this paper, and we refer reader to recent reviews such as [65], [77], and [108].

B. Advantages of GPD Formalization

The most important point of the GPD concept is to embed the entire process of a given recognition task into a smooth functional form so that one can optimize all of the adjustable system parameters in a manner consistent with the design objective of minimizing recognition errors.

There is the natural intuition that discriminative design methods such as GPD can achieve a higher recognition accuracy using fewer trainable system parameters than methods based on the ML principle: the discriminative design method focuses on classification results near the class borders. In fact, we demonstrated this efficiency in terms of the number of trainable system parameters in many experimental results in literature, e.g., [73], a GPD-trained prototype, reference pattern-based classifier achieved higher recognition scores than an ML-trained hidden Markov network having about twenty times more trainable parameters than that of the GPD-trained system. Unfortunately, this “efficiency” advantage has often been criticized: the high discriminative power is considered to lead to a less robust design result, i.e., an overlearning of the given design samples. However, such criticism is simplistic and misses an essential point. The overlearning phenomenon is commonly observed in any system with a surplus of adjustable parameters; on the other hand, it is rarely observed in discriminative designs using fewer parameters. Any overlearning in a discriminative design has its basis in the directness by which such designs are linked to a given classification task and therefore demonstrates the power of the approach.

The significance of GPD is that it has both mathematical rigor and a great degree of practicality. GPD was shown to provide attractive solutions to three of the four major DFA issues: 1) the design objective; 2) optimization method; and 3) design consistency with unknown samples. In addition, the high utility of GPD has been demonstrated in various experiments, as were introduced in the paper. The remaining DFA issue, i.e., the selection of the discriminant function form, has not been fully studied to date. Basically, this point should be investigated in a task-by-task basis, and GPD, which gives a sound mathematical framework for the other design issues, can be quite useful in this future study.

C. Future Issues in GPD-Based Pattern Recognition

As introduced in this paper, GPD provides useful solutions to many of the existing problems in speech pattern recognition. However, there are still issues needing further investigation. A point of utmost importance is the discovery (for any given task) of a desirable form for the discriminant function, as cited in the last paragraph of the previous subsection. Solving this problem will dramatically advance speech recognition technology, but it is obviously difficult and needs significant research effort. Another important point is to find a reasonable way of controlling the smoothness of the functions, such as the smooth classification error count loss, defined in the GPD training. This point is closely linked to the issue of training sensitivity to unknown samples, and study on the point may necessitate advanced analyses from the viewpoint of statistics. The following points are rather minor (compared with the issues of the discriminant function form selection and smoothness control) but clearly important for the effective implementation of GPD methods.

1) DFA-based training suffers essentially from a scaling problem, that is, in a large-scale task such as large-vocabulary continuous speech recognition, extensive computation is involved in evaluating the interclass competition over the tremendous number of possible classes (of connected words). Obviously, the misclassification measure of GPD possesses the same problem. In addition to the simplification based on the use of the L_∞ norm function, further simplification, such as the use of N-best hypotheses [21], [53], may be needed to reduce the adjustment computation in the training stage.
2) The gradient search optimization used in GPD is usually slower than the common expectation-maximization (EM) method, especially in a large-scale task. Incorporating a faster optimization mechanism in the GPD formalism will clearly be useful.
3) The success of the gradient-based adjustment relies on a good selection of several controllable parameters such as the learning factor. The selection often controls the balance of interaction among the modules (such as the feature extractor and the classifier) as well as the speed of training convergence. Since the selection is usually performed experimentally due to a lack of theory, a more systematic and theoretically grounded selection method is needed.

The most important concept of the GPD formalization is, in short, to formalize the overall procedure of the task at hand into an optimizable design process. As the original form of GPD for classification problems has been greatly extended so as to cope with various aspects of pattern recognition, such as the open vocabulary problem and speaker verification, we could further extend the family to other types of tasks as well. The simple but significant concept of GPD, i.e., formalizing the overall procedure of the task at hand into an optimizable design process, can be quite useful in general fields such as system design and information processing.

Appendix

A. Probabilistic Descent Theorem for Probability-Based Discriminant Functions

In the case of using a probability function as the discriminant function, the adjustment convergence of (20) no longer holds true; the probabilistic descent theorem assumed the classifier parameter Λ to be arbitrary scalar variables and/or arbitrary vector variables and did not anticipate that each class parameter vector $\lambda_j = (\lambda_j[1], \ldots, \lambda_j[D])^T$ had to meet the following probability constraint:

$$\sum_d \lambda_j[d] = 1 \quad \text{and} \quad \lambda_j[d] \geq 0 \text{ (for } \forall d) \tag{72}$$

where we assume that λ_j is of D-dimension. The HMM state transition probability and the mixture coefficients of the mixtured Gaussian HMM are typical examples of this type of constrained parameter vector.

One solution to this problem is to use the transformation of (61). In addition, [54] and [56] provides a more general solution, i.e., the following, constrained probabilistic descent theorem for applying the GPD-based adjustment to the constrained parameters.

Theorem 2 (Constrained Probabilistic Descent Theorem): Assume that a design sample $x_1^T(t) (\in C_k)$ is given and a classifier parameter set λ_j for C_j includes the parameter vector set $\{\rho_j\}$ that satisfies the constraint of (72). Then, the adjustment using $(\delta\Lambda(x_1^T(t), C_k, \Lambda))_\Phi$, which is obtained by projecting $\delta\Lambda(x_1^T(t), C_k, \Lambda)$ of (20) to the ρ_j's subspace that is spanned according to the constraint, reduces $\delta L(\Lambda)$ in the sense of expectation; here $(\,)_\Phi$ is the orthogonal projection of the parenthesized parameter vector onto the (72)-based subspace of the parameter vector space.

Proof: Considering that the gradient computation is independently done for every dimensional element of the parameter vector, we can prove the theorem by showing that the adjustment in terms of one parameter vector ρ_j, which satisfies the constraint, reduces $\delta L(\Lambda)$ in the expectation sense. In fact, the following inequality clearly proves that the adjustment can reduce $\delta L(\Lambda)$ in the expectation sense

$$\begin{aligned} E[\delta L(\Lambda)] &= E[(\delta\Lambda(x_n, C_k, \Lambda))_\Phi \cdot \nabla L(\Lambda)] \\ &= -\epsilon\{\boldsymbol{U} E[\nabla \ell_k(x_n; \Lambda)]\} \cdot (\nabla L(\Lambda))_\Phi \\ &= -\epsilon\{\boldsymbol{U} \nabla L(\Lambda)\} \cdot (\nabla L(\Lambda))_\Phi \\ &= -\epsilon \boldsymbol{U} ||\nabla L(\Lambda)||^2 cos\vartheta \leq 0 \end{aligned} \tag{73}$$

where ϑ is the angle between $\nabla L(\Lambda)$ and $(\nabla L(\Lambda))_\Phi$, and $\vartheta \leq \pi/2$ always holds true. ■

Now it is clear that the characteristics of the original theorem, such as the stochastic approximation-based convergence to a locally optimal situation, hold true.

Let us point out that even in the case of using the probability-based discriminant function, all the parameters do not need to observe the constrained probabilistic descent adjustment. For example, the mean vector of the component Gaussian distribution of the mixtured Gaussian HMM system can be updated according to (20). Moreover, the covariance matrix of the component distribution can be updated in the same way. The adjustment of the diagonal covariance matrix, which is widely used in speech pattern recognition, is actually easy. The adjustment of the full covariance matrix is somewhat complicated and difficult: it should satisfy the positive-definiteness constraint of the matrix. However, it can be done fundamentally based on (20). A GPD implementation for this case is presented in detail in [103].

B. Selection of Discriminant Function Forms

In this paper, we have especially used the distance classifier for explanations. Thus, the form of the discriminant function used has been distance. We have also mentioned briefly the case of the probability function form of the discriminant function. Obviously, the distance is closely related to the probability function, and thus it is unreasonable to consider these two forms to be separate. In order to embody GPD effectively, we should comprehensively understand the nature of the discriminant functions selected. In light of this, we summarize here the nature of several important discriminant function forms (or the structure of classifiers).

Discussions focusing on the fixed-dimensional sample space cannot necessarily be applied directly to the classification of dynamic patterns. However, it should be of some help to summarize various structures of the static pattern classifiers.

As shown in the literature, such as [33], it is known that the multilayer perceptron (MLP), having a sigmoidal output function at its hidden nodes, possesses quite a large potential for function approximation. Reference [52] introduced the GPD implementation for this type of nonlinear network classifier and demonstrated the high efficiency and effectiveness of its training.

There are several other types of feedforward networks that are similar to MLP in terms of structure but that compute different measures of distance. Among them, let us focus on a three-layer RBF network. Interestingly, it is shown that this type of network achieves universal function approximation under mild conditions, even though linear (e.g., see [40]). Thus, similar to the above MLP case [52], an RBF network trained with GPD would thus greatly contribute to achieving the minimum classification error probability condition.

If the three-layer RBF network uses a continuous probabilistic basis function and, furthermore, the upper connection coefficients satisfy the probability constraint (72), the network works as a likelihood network that uses probability or likelihood as the discriminant function [55]. In particular, a likelihood network using the Gaussian-form probability function as the discriminant function, i.e., a mixtured Gaussian likelihood network, would be quite useful due to its simple computation. Due to the probability constraint, a likelihood network would not have the universal approximation capability of the original RBF network in the sense of [40]. However, it was shown that the mixtured Gaussian density function could approximate an arbitrary continuous probability density function in the sense of minimizing the L_p norm error between the two corresponding density functions [95]. These results would therefore enable us to expect highly effective applications of GPD to this Gaussian-basis likelihood network.

The mixtured Gaussian likelihood network is obviously equivalent to the one state case of the mixtured Gaussian-component, continuous HMM. The discussions of the above paragraphs thus suggest that the GPD-trained mixtured Gaussian-component HMM classifier would have a large potential for achieving the minimum error probability condition of dynamic pattern classification.

Similar to the Mahalanobis distance, several likelihood-based distance measures can be defined based on the Gaussian distribution probability density function. The squared Euclidean distance is the most limited but simplest version among these likelihood-based distances. Let us replace the probability computation with a computation using one of these distances in a Gaussian-basis likelihood network. We assume here that the lower connection weights correspond to the mean vectors of the Gaussian-basis function. Then, the resulting network is obviously equivalent to a distance network using the distance between an input vector and a reference vector as the discriminant function [55]. One may note here that this distance network can also be seen as a generalized version of a distance classifier. Fundamentally, in a distance network, the output nodes are fully connected to the hidden nodes. On the other hand, as described in Section II-D, a distance classifier usually assigns one reference vector to one of all of the possible classes exclusively. The distance network does not take such *a priori* assignment of the reference vectors; instead, this network attempts to share the reference vectors (inner models) among the classes. The same concept of sharing is also implemented for a continuous HMM [44].

In practical applications of RBF networks and distance networks for classification, the network classifiers usually use a much smaller number of hidden nodes than given design samples. That is, any clustering concept for the given samples underlies the setting of the hidden nodes. Let us consider the situation in which as we increase the number of hidden nodes, we decrease the size of the cluster corresponding to each hidden node. In the extreme case in which each hidden-node cluster corresponds to one sample, the RBF network performs a generalized Parzen approximation and the distance network works as a generalized k-nearest neighbor classifier. The generalization here relies on flexibility in determining the upper layer connection weights. The Parzen estimate of probability distribution is an unbiased and consistent estimate of the true distribution function. Moreover, the error probability of the k-nearest neighbor method is bounded to be below twice the Bayes error probability. These well known characteristics suggest that the distance network is also worth further study.

Fundamentally, one should select classifier structures, discussed above, that determine the discriminant function forms, based on ease of hardware implementation, computational efficiency, and the nature of a given feature representation.

C. Various Definitions of Discriminant Functions, Misclassification Measures, and Losses

There is a large variety in the defining functions used in GPD, such as the discriminant functions and the misclassification measures. Assuming that the discriminant function represents the degree to which an input sample belongs to a class, we present several examples of definitions which have not been used in the previous sections. Note that we assume a dynamic pattern x_1^T $(\in C_k)$ to be an input.

- Discriminant function

$$g_j(x_1^T;\Lambda) = \ln\, p_\Lambda(x_1^T|C_j) \tag{74}$$

- Misclassification measure

$$d_k(x_1^T;\Lambda) = -1 + \frac{\left[\dfrac{1}{M-1}\left\{\displaystyle\sum_{j,j\neq k} g_j(x_1^T;\Lambda)\right\}^{\mu}\right]^{1/\mu}}{g_k(x_1^T;\Lambda)} \tag{75}$$

$$d_k(x_1^T;\Lambda) = 1 - \frac{g_k(x_1^T;\Lambda)}{\left[\dfrac{1}{M-1}\left\{\displaystyle\sum_{j,j\neq k} g_j(x_1^T;\Lambda)\right\}^{\mu}\right]^{1/\mu}} \tag{76}$$

$$d_k(x_1^T;\Lambda) = -g_k(x_1^T;\Lambda) + \ln\left\{\frac{1}{M-1}\sum_{j,j\neq k} e^{g_j(x_1^T;\Lambda)\mu}\right\}^{1/\mu}. \tag{77}$$

The form of (77) was given in [24].

- Loss

$$\tilde{\ell}_k(x_1^T;\Lambda) = \frac{\displaystyle\sum_{j,j\neq k}^{M}\{g_j(x_1^T;\Lambda)\}^{\gamma}\delta_{jk}}{\displaystyle\sum_{j,j\neq k}^{M}\{g_j(x_1^T;\Lambda)\}^{\gamma}}\,\ell_k(x_1^T;\Lambda) \tag{78}$$

where $\ell_k(x_1^T;\Lambda)$ is such a usual loss as (37), and δ_{jk} is a symbolic distance between C_j and C_k, which is often determined by parsing in speech recognition. Equation (78) was proposed in [71]. Using this loss, one can attempt to pursue the minimum of a specific risk to a given task instead of the general misclassification account.

Acknowledgment

The authors would like to express their thanks to their many colleagues at the Bell Laboratories and the ATR Laboratories for their significant contributions over the years to the development of the GPD family. In particular, they thank A. Biem, W. Chou, E. McDermott, H. Watanabe, and E. Woudenberg, whose contributions made possible the authors' development work. They would also like to give special thanks to E. Woudenberg for his patience and assistance during the preparation of this paper. Finally, the authors acknowledge the careful review work of their two reviewers. Their detailed comments greatly helped to substantially improve the quality of this paper.

References

[1] S. Amari, "A theory of adaptive pattern classifiers," *IEEE Trans. Electron. Comput.*, vol. EC-16, pp. 299–307, Mar. 1967.
[2] ——, *Information Theory II—Geometrical Theory of Information*, (in Japanese). Tokyo: Kyoritsu, 1968.
[3] ——, *Neural Network Model and Connectionism* (in Japanese). Tokyo: Tokyo Univ. Press, 1989.
[4] A. Ando and K. Ozeki, "A clustering algorithm to minimize recognition error function," *IEICE Trans. A*, vol. J74-A, no. 3, pp. 360–367, Mar. 1991.
[5] H. Asou, *Neural Network Information Processing—Introduction to Connectionism, or Toward Soft Symbolic Representation* (in Japanese). Tokyo: Sangyo Tosho, 1988.
[6] H. Asou and N. Otsu, "Nonlinear data analysis and multilayer perceptrons," in *Proc. IEEE IJCNN*, vol. 2, 1989, pp. 411–415.
[7] L. Bahl, P. Brown, P. de Souza, and R. Mercer, "Maximum mutual information estimation of hidden Markov model parameters for speech recognition," in *Proc. IEEE ICASSP'86*, vol. 1, pp. 49–52.
[8] ——, "A new algorithm for the estimation of hidden Markov model parameters," in *Proc. IEEE. ICASSP'98*, vol. 1, pp. 493–496.
[9] L. E. Baum and T. Petrie, "Statistical inference for probabilistic functions of finite state Markov chains," *Ann. Math. Stat.*, vol. 37, pp. 1554–1563, 1966.
[10] L. E. Baum and G. R. Sell, "Growth functions for transformations on manifolds," *Pacific J. Math.*, vol. 27, no. 2, pp. 211–227, 1968.
[11] Y. Bengio, R. De Mori, G. Flammia, and R. Kompe, "Global optimization of a neural network-hidden Markov model hybrid," *IEEE Trans. Neural Networks*, vol. 3, no. 2, pp. 252–259, Mar. 1992.
[12] A. Biem and S. Katagiri, "Feature extraction based on minimum classification error/generalized probabilistic descent method," in *Proc. IEEE ICASSP'93*, vol. 2, pp. 275–278.
[13] A. Biem, S. Katagiri, and B.-H. Juang, "Discriminative feature extraction for speech recognition," in *Proc. 1993 IEEE Workshop Neural Networks for Signal Processing III*, pp. 392–401.
[14] A. Biem, E. McDermott, and S. Katagiri, "Discriminative feature extraction to filter bank design," in *Proc. 1996 IEEE Workshop Neural Networks for Signal Processing VI*, pp. 273–282.
[15] A. Biem, S. Katagiri, and B.-H. Juang, "Pattern recognition using discriminative feature extraction," *IEEE Trans. Signal Processing*, vol. 45, pp. 500–504, Feb. 1997.
[16] H. Bourlard and C. Wellekens, "Links between Markov models and multilayer perceptrons," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 12, pp. 1167–1178, Dec. 1990.
[17] H. Bourlard and N. Morgan, *Connectionist Speech Recognition—A Hybrid Approach*. Norwell, MA: Kluwer, 1994.
[18] J. Bridle, "Alpha-nets: A recurrent 'neural' network architecture with a hidden Markov model interpretation," *Speech Commun.*, vol. 9, pp. 83–92, 1990.
[19] P.-C. Chang and B.-H. Juang, "Discriminative template training for dynamic programming speech recognition," in *Proc. IEEE ICASSP'92*, vol. 1, pp. 493–496.
[20] ——, "Discriminative training of dynamic programming based speech recognizers," *IEEE Trans. Speech Audio Processing*, vol. 1, pp. 135–143, Feb. 1993.
[21] J.-K. Chen and F. K. Soong, "An N-best candidates-based discriminative training for speech recognition applications," *IEEE Trans. Speech Audio Processing*, vol. 2, pp. 206–216, Jan. 1994.
[22] R. Chengalvarayan and L. Deng, "Use of generalized dynamic feature parameters for speech recognition," *IEEE Trans. Speech Audio Processing*, vol. 5, pp. 232–242, Mar. 1997.
[23] ——, "HMM-based speech recognition using state-dependent, discriminatively derived transforms on Mel-warped DFT features," *IEEE Trans. Speech Audio Processing*, vol. 5, pp. 243–256, Mar. 1997.

[24] W. Chou, B.-H. Juang, and C.-H. Lee, "Segmental GPD training of HMM based speech recognition," in *Proc. IEEE ICASSP'92*, vol. 1, pp. 473-476.

[25] ——, "Minimum error rate training based on N-best string models," in *Proc. IEEE ICASSP'93*, vol. 2, pp. 652–655.

[26] W. Chou, C.-H. Lee, and B.-H. Juang, "Minimum error rate training of inter-word context dependent acoustic model units in speech recognition," in *Proc. ICSLP'94*, pp. 439–442.

[27] W. Chou, M. G. Rahim, and E. Buhrke, "Signal conditioned minimum error rate training," in *Proc. EUROSPEECH'95*, pp. 495–498.

[28] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Roy. Statist. Soc.*, vol. 39, no. 1, pp. 1–38, 1977.

[29] R. Duda, and P. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.

[30] S. Euler, "Integrated optimization of feature transformation for speech recognition," in *Proc. EUROSPEECH'95*, pp. 109–112.

[31] K. R. Farrell, R. J. Mammone, and K. T. Assaleh, "Speaker recognition using neural networks and conventional classifiers," *IEEE Trans. Speech Audio Processing*, vol. 2, no. 1, pt. 2, pp. 194–205, Jan. 1994.

[32] K. Fukunaga, *Introduction to Statistical Pattern Recognition*. New York: Academic, 1972.

[33] K. Funahashi, "On the approximate realization of continuous mappings by neural networks," *Neural Networks*, vol. 2, no. 3, pp. 183–191, 1989.

[34] Y.-Q. Gao, T.-Y. Huang, and D.-W. Chen, "HMM-based warping in neural networks," in *Proc. IEEE ICASSP'90*, vol. 1, pp. 501–504.

[35] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," *IEEE Trans. Parallel Anal. Machine Intell.*, vol. PAMI-6, pp. 721–741, June 1984.

[36] H. Gish, "A probabilistic approach to the understanding and training of neural network classifiers," *Proc. IEEE ICASSP'90*, vol. 3, pp. 1361–1364.

[37] P. Haffner, M. Franzini, and A. Waibel, "Integrating time alignment and neural networks for high performance continuous speech recognition," in *Proc. IEEE ICASSP'91*, pp. 105–108.

[38] P. Haffner, "A new probabilistic framework for connectionist time alignment," in *Proc. ICSLP'94*, pp. 1559–1562.

[39] J. Hampshire II and A. Waibel, "A novel objective function for improved phoneme recognition using time-delay neural networks," *IEEE Trans. Neural Networks*, vol. 1, pp. 216–228, Mar. 1990.

[40] E. Hartman, J. Keeler, and J. Kowalski, "Layered neural networks with Gaussian hidden units as universal approximations," *Neural Computation*, vol. 2, pp. 210–215, 1990.

[41] J. Hernando, J. Ayarte, and E. Monte, "Optimization of speech parameter weighting for CDHMM word recognition," in *Proc. EUROSPEECH'95*, pp. 105–108.

[42] J. Hopfield and D. Tank, "Neural computation of decisions in optimization problems," in *Biological Cybernetics*, vol. 52. New York: Springer-Verlag, pp. 141–152, 1985.

[43] D. Howell, "The multi-layer perceptron as a discriminating post processor for hidden Markov networks," in *Proc. 7th FASE Symp.*, 1988, pp. 1389–1396.

[44] X. Huang, Y. Ariki, and M. Jack, *Hidden Markov Models For Speech Recognition*. Edinburg, U.K.: Edinburg Univ. Press, 1990.

[45] X. Huang, M. Belin, F. Alleva, and M. Hwang, "Unified stochastic engine (USE) for speech recognition," in *Proc. IEEE ICASSP'93*, vol. 2, pp. 636–639.

[46] T. Iijima, *Pattern Recognition Theory,* (in Japanese). Tokyo: Morikita, 1989.

[47] H. Iwamida, S. Katagiri, E. McDermott, and Y. Tohkura, "A hybrid speech recognition system using HMM's with an LVQ-trained codebook," *ASJ, J. Acoustical Soc. Japan (E)*, vol. 11, no. 5, pp. 277–286, Sept. 1990.

[48] H. Iwamida, S. Katagiri, and E. McDermott, "Re-evaluation of LVQ-HMM hybrid algorithm," *ASJ, J. Acoustical Soc. Japan (E)*, vol. 14, no. 4, pp. 267–274, July 1993.

[49] F. Jelinek, "Self-organized language modeling for speech recognition," IBM T. J. Watson Research Center, Yorktown Heights, NY, Tech. Rep., 1985.

[50] B.-H. Juang and L. Rabiner, "The segmental K-means algorithm for estimating parameters of hidden Markov models," *IEEE Trans. Audio Speech Signal Processing*, vol. 38, pp. 1639–1641, Sept. 1990.

[51] B.-H. Juang and S. Katagiri, "Discriminative training," *ASJ, J. Acoustical Soc. Japan (E)*, vol. 13, no. 6, pp. 333–339, Nov. 1992.

[52] ——, "Discriminative learning for minimum error classification," *IEEE Trans. Signal Processing.*, vol. 40, pp. 3043–3054, Dec. 1992.

[53] B.-H. Juang, W. Chou, and C.-H. Lee, "Minimum classification error rate methods for speech recognition," *IEEE Trans. Speech Audio Processing.*, vol. 5, pp. 257–265, Mar. 1997.

[54] S. Katagiri, C.-H. Lee, and B.-H. Juang, "A generalized probabilistic descent method," in *Proc. ASJ Autumn Conf.*, 1990, pp. 141–142.

[55] ——, "Discriminative multi-layer feed-forward networks," in *Proc. IEEE Workshop Neural Networks for Signal Processing*, 1991, pp. 11–20.

[56] ——, "New discriminative training algorithms based on the generalized probabilistic descent method," in *Proc. IEEE Workshop Neural Networks for Signal Processing*, 1991, pp. 299–308.

[57] S. Katagiri and C.-H. Lee, "A new hybrid algorithm for speech recognition based on HMM segmentation and learning vector quantization," *IEEE Trans. Speech Audio Processing*, vol. 1, pp. 421–430, Apr. 1993.

[58] S. Katagiri, B.-H. Juang, and A. Biem, "Discriminative feature extraction," in *Artificial Neural Networks for Speech and Vision*, R. Mammone, Ed. London, U.K.: Chapman and Hall, 1994, pp. 278–293.

[59] S. Katagiri, "A unified approach to pattern recognition," in *Proc. ISANN'94*, pp. 561–570.

[60] D. Kimber, M. Bush, and G. Tajchman, "Speaker-independent vowel classification using hidden Markov models and LVQ2," in *Proc. IEEE ICASSP'90*, vol. 1, pp. 497–500.

[61] T. Kohonen, "The self-organizing map," *Proc. IEEE*, vol. 78, pp. 1464–1480, Sept. 1990.

[62] T. Komori and S. Katagiri, "GPD training of dynamic programming-based speech recognizers," *ASJ, J. Acoustical Soc. Japan (E)*, vol. 13, no. 6, pp. 341–349, Nov. 1992.

[63] ——, "A minimum error approach to spotting-based pattern recognition," *IEICE Trans. Inform. Syst.*, vol. E78-D, no. 8, pp. 1032–1043, Aug. 1995.

[64] ——, "A novel spotting-based approach to continuous speech recognition: Minimum error classification of keyword-sequences," *ASJ, J. Acoustical Soc. Japan (E)*, vol. 16, no. 3, pp. 147–157, May 1995.

[65] C.-H. Lee, F. K. Soong, and K. K. Paliwal, Eds., *Automatic Speech and Speaker Recognition*. Norwell, MA: Kluwer, 1996.

[66] C.-S. Liu, C.-H. Lee, W. Chou, B.-H. Juang, and A. Rosenberg, "A study on minimum error discriminative training for speaker recognition," *J. Acoustical Soc. Amer.*, vol. 97, no. 1, pp. 637–648, Jan. 1995.

[67] S. Makino, M. Endo, T. Sone, and K. Kido, "Recognition of phonemes in continuous speech using a modified LVQ2 method," *ASJ, J. Acoustical Soc. Japan (E)*, vol. 13, no. 6, pp. 351–360, Nov. 1992.

[68] T. Matsui and S. Furui, "A study of speaker adaptation based on minimum classification error training," in *Proc. EUROSPEECH'95*, pp. 81–84.

[69] E. McDermott, "LVQ3 for phoneme recognition," in *Proc. ASJ Spring Conf.*, 1990, pp. 151–152.

[70] E. McDermott and S. Katagiri, "LVQ-based shift-tolerant phoneme recognition," *IEEE Trans. Signal Processing,* vol. 39, pp. 1398–1411, June 1991.

[71] ——, "Prototype-based MCE/GPD training for word spotting and connected word recognition," in *Proc. IEEE ICASSP'93*, vol. 2, pp. 291–294.

[72] ——, "Prototype-based MCE/GPD training for various speech units," *Comput. Speech Language*, vol. 8, pp. 351–368, 1994.

[73] ——, "String-level MCE for continuous phoneme recognition," in *Proc. EUROSPEECH'97*, vol. 1, pp. 123–126.

[74] M. Miyatake, H. Sawai, Y. Minami, and K. Shikano, "Integrated training for spotting Japanese phonemes using large phonemic time-delay neural networks," in *Proc. IEEE ICASSP'90*, vol. 1, pp. 449–452.

[75] S. Mizuta and K. Nakajima, "A discriminative training method for continuous mixture density HMM's and its implementation

to recognize noisy speech," *ASJ, J. Acoustical Soc. Japan (E)*, vol. 13, no. 6, pp. 389–393, Nov. 1992.

[76] H. Mizutani, "Discriminative learning for minimum error classification with reject options," (in Japanese), IEICE, Tokyo, Tech. Rep. PRMU97-245, 1998.

[77] N. Morgan and H. A. Bourlard, "Neural networks for statistical recognition of continuous speech," *Proc. IEEE*, vol. 83, pp. 742–770, May 1995.

[78] L. Niles and H. Silverman, "Combining hidden Markov model and neural network classifier," in *Proc. IEEE ICASSP'90*, vol. 1, pp. 417–420.

[79] N. Nilsson, *The Mathematical Foundations of Learning Machines*. San Mateo, CA: Morgan Kaufmann, 1990.

[80] Y. Normandin, R. Cardin, and R. De Mori, "High-performance connected digit recognition using maximum mutual information estimation," *IEEE Trans. Speech Audio Processing*, vol. 2, pp. 299–311, Feb. 1994.

[81] E. Oja, *Subspace Methods of Pattern Recognition*. Letchworth: Research Studies Press, 1983.

[82] K. K. Paliwal, M. Bacchiani, and Y. Sagisaka, "Minimum classification error training algorithm for feature extractor and pattern classifier in speech recognition," in *Proc. EUROSPEECH'95*, pp. 541–544.

[83] Y.-H. Pao, *Adaptive Pattern Recognition and Neural Networks*. Reading, MA: Addison-Wesley, 1989.

[84] J. Park and I. Sandberg, "Universal approximation using radial-basis-function networks," *Neural Computation*, vol. 3, pp. 246–257, 1991.

[85] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition.* Englewood Cliffs, NJ: Prentice-Hall, 1993.

[86] M. G. Rahim and C.-H. Lee, "Simultaneous ANN feature and HMM recognizer design using string-based minimum classification error (MCE) training," in *Proc. ICSLP'96*, pp. 1824–1827.

[87] D. Rainton and S. Sagayama, "Minimum error classification training of HMM's—Implementation details and experimental results," *ASJ, J. Acoustical Soc. Japan (E)*, vol. 13, no. 6, pp. 379–387, Nov. 1992.

[88] A. J. Robinson, "An application of recurrent nets to phone probability estimation," *IEEE Trans. Neural Networks*, vol. 5, pp. 298–305, Mar. 1994.

[89] R. C. Rose, B.-H. Juang, and C.-H. Lee, "A training procedure for verifying string hypothesis in continuous speech recognition," in *Proc. ICASSP'95* pp. 281–284.

[90] R. C. Rose, "Word spotting from continuous speech utterances," in *Automatic Speech and Speaker Recognition: Advanced Topics*, C.-H. Lee, F. K. Soong, and K. K. Paliwal, Eds. Norwell, MA: Kluwer, 1996, pp. 303–329.

[91] D. E. Rumelhart, J. L. McClelland, and the PDP Research Group, *Parallel Distributed Processing*. Cambridge, MA: MIT, 1986.

[92] H. Sakoe, R. Isotani, K. Yoshida, K. Iso, and T. Watanabe, "Speaker-independent word recognition using dynamic programming neural networks," in *Proc. IEEE ICASSP'89*, vol. 1, pp. 29–32.

[93] A. Sato and K. Yamada, "A learning method for definite canonicalization based on minimum classification error," (in Japanese), Tokyo, IEICE, Tech. Rep. PRMU97-244, 1998.

[94] A. Setlur and T. Jacobs, "Results of a speaker verification service trial using HMM models," in *Proc. EUROSPEECH'95*, pp. 639–642.

[95] H. Sorenson and D. Alspach, "Recursive Baysian estimation using Gaussian sums," *Automatica*, vol. 7, pp. 465–479, 1971.

[96] K.-Y. Su, T.-H. Chiang, and Y.-C. Lin, "A unified framework to incorporate speech and language information in spoken language processing," in *Proc. ICASSP'92*, vol. 1, pp. 185–188.

[97] K.-Y. Su and C.-H. Lee, "Speech recognition using weighted HMM and subspace projection approaches," in *IEEE Trans. Speech Audio Processing*, vol. 2, pp. 69–79, Jan. 1994.

[98] M. Sugiyama and K. Kurinami, "Minimal classification error optimization for a speaker mapping neural network," in *Proc. IEEE Workshop Neural Networks for Signal Processing II*, 1992, pp. 233–242.

[99] K. Unnikrishnan, J. Hopfield, and D. Tank, "Connected-digit speaker-dependent speech recognition using a neural network with time-delayed connections," *IEEE Trans. Signal Processing*, vol. 39, pp. 698–713, Mar. 1991.

[100] V. Valtchev, J. J. Odell, P. C. Woodland, and S. J. Young, "Lattice-based discriminative training for large vocabulary speech recognition," in *Proc. IEEE ICASSP'96*, pp. 605–608.

[101] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Trans. Audio Speech Signal Processing*, vol. 37, pp. 328–339, Mar. 1989.

[102] S. Watanabe, P. F. Lambert, C. A. Kulikowski, J. L. Buxton, and R. Walker, "Evaluation and selection of variables in pattern recognition," in *Computer and Information Sciences II*, J. T. Tou, Ed. New York: Academic, 1967, pp. 91–122.

[103] H. Watanabe, T. Yamaguchi, and S. Katagiri, "A novel approach to pattern recognition based on discriminative metric design," in *Proc. IEEE Workshop Neural Networks for Signal Processing V*, 1995, pp. 48–57.

[104] H. Watanabe and S. Katagiri, "Discriminative subspace method for minimum error pattern recognition," in *Proc. IEEE Workshop Neural Networks for Signal Processing V*, 1995, pp. 77–86.

[105] H. Watanabe, T. Yamaguchi, and S. Katagiri, "Discriminative metric design for robust pattern recognition," *IEEE Trans. Signal Processing*, vol. 45, pp. 2655–2662, Nov. 1997.

[106] H. Watanabe and S. Katagiri, "Subspace method for minimum error pattern recognition," *IEICE Trans. Inform. Syst.*, vol. E80-D, no. 12, pp. 1195–1204, Dec. 1997.

[107] H. Watanabe, Y. Matsumoto, and S. Katagiri, "Minimum detection error training for acoustic signal monitoring," in *Proc. IEEE ICASSP'98*, vol. 2, pp. 1193–1196.

[108] S. Young, "A review of large-vocabulary continuous-speech recognition," *IEEE Signal Processing Mag.*, vol. 13, pp. 45–57, May 1996.

[109] G. Yu, W. Russel, R. Schwartz, and J. Makhoul, "Discriminant continuous speech recognition," in *Proc. IEEE ICASSP'90*, vol. 2, pp. 685–688.

Shigeru Katagiri (Senior Member, IEEE) was born in Japan on November 3, 1953. He received the B.E. degree in electrical engineering and the M.E. and Dr.Eng. degrees in information engineering from Tohoku University, Sendai, Japan, in 1977, 1979, and 1982, respectively.

From 1982 to 1986 he worked at the Electrical Communication Laboratories, Nippon Telegraph and Telephone Public Corporation, Tokyo, Japan, where he was engaged in speech recognition research. Since 1986 he has been with the Advanced Telecommunications Research Institute International, Kyoto, Japan, where he is currently Head of the ATR Human Information Processing Research Laboratories. During 1989–1990 he was a Visiting Researcher with the Speech Research Department, AT&T Bell Laboratories. In 1998 he joined the faculty of the Graduate School of Informatics, Kyoto University, Kyoto, Japan, as a Visiting Professor. His current research interests include studies on discriminative training, the learning capability of artificial neural networks, and speech recognition.

Dr. Katagiri was a co-recipient of the 22nd Sato Paper Award of the Acoustical Society of Japan (ASJ), the 27th Sato Paper Award of the ASJ, and the IEEE Signal Processing Society 1993 Senior Award in 1982, 1987, and 1994, respectively. He is a Senior Member of the IEEE Signal Processing Society (SPS) and a member of the Acoustical Society of America, the Acoustical Society of Japan, the Institute of Electronics, Information and Communication Engineers (IEICE) of Japan, and the Japanese Society for Artificial Intelligence. He has served as Associate Editor of the IEEE TRANSACTIONS ON SIGNAL PROCESSING (1994–1997), as Secretary of the Technical Committee on Neural Networks for Signal Processing of the IEEE Signal Processing Society (1997–1998), and as the Vice-Chair of the Tokyo Chapter of the IEEE Signal Processing Society. He has also served as an Associate Editor of the *Transactions of IEICE D-II*. As Program Chair he organized the 1996 IEEE Signal Processing Society Workshop on Neural Networks for Signal Processing (NNSP'96).

Biing-Hwang Juang (Fellow, IEEE) received the B.Sc. degree in electrical engineering from National Taiwan University, Taipei, in 1973 and the M.Sc. and Ph.D. degrees in electrical and computer engineering from the University of California, Santa Barbara, in 1979 and 1981, respectively.

In 1978 he did research on vocal tract modeling at Speech Communications Research Laboratory (SCRL). He then joined Signal Technology, Inc. in 1979 as a Research Scientist, working on signal and speech related topics. Since 1982, he has been with Bell Labs where he is engaged in a wide range of communication-related research activities ranging from speech coding and speech recognition to multimedia communications. He is currently Head of the Acoustics & Speech Research Department in the Multimedia Communications Research Laboratory. He holds a number of patents in the area of speech communication and communication services, and he is co-author of the book *Fundamentals of Speech Recognition* (Prentice-Hall, 1993).

Dr. Juang was an Editor of IEEE TRANSACTIONS ON ACOUSTICS, SPEECH, AND SIGNAL PROCESSING (1986–1988), the IEEE TRANSACTIONS ON NEURAL NETWORKS (1992–1993), and the *Journal of Speech Communication* (1992–1994). He has served on the Digital Signal Processing and the Speech Technical Committees as well as the Conference Board of the IEEE Signal Processing Society, and he was Chairman of the Technical Committee on Neural Networks for Signal Processing (1991–1993). He is currently Editor-in-Chief of the IEEE TRANSACTIONS ON SPEECH AND AUDIO PROCESSING and a member of the Publication Board of the PROCEEDINGS OF THE IEEE. He also serves on several international advisory boards outside the United States. He received the IEEE Signal Processing Society's 1993 Senior Paper Award, 1994 Senior Paper Award, and the 1974 Best *Signal Processing Magazine* Paper Award. He received the Bell Labs President's Gold Award in 1997 for leading the effort in automatic speech recognition.

Chin-Hui Lee (Fellow, IEEE) received the B.S. degree in electrical engineering from National Taiwan University, Taipei, in 1973, the M.S. degree in engineering and applied science from Yale University, New Haven, CT, in 1977, and the Ph.D. degree in electrical engineering with a minor in statistics from University of Washington, Seattle, in 1981.

In 1981 he joined Verbex Corporation, Bedford, MA, where he was involved in research work on connected word recognition. In 1984 he became affiliated with Digital Sound Corporation, Santa Barbara, CA, where he was engaged in research and product development in speech coding, speech synthesis, speech recognition, and signal processing for the development of the DSC-2000 Voice Server. Since 1986 he has been with Bell Laboratories, Murray Hill, NJ, where he is now a Distinguished Member of Technical Staff and Head of the newly established Dialogue Systems Research Department. His current research includes multimedia signal processing, speech and speaker recognition, speech and language modeling, adaptive and discriminative learning, spoken dialogue processing, biometric authentication, and information retrieval. His research scope is reflected in a recent book entitled *Automatic Speech and Speaker Recognition: Advanced Topics* (Norwell, MA: Kluwer, 1996). He has published over 170 papers and 16 patents on the subject of automatic speech and speaker recognition.

Dr. Lee is a member of the IEEE Signal Processing Society, the IEEE Communication Society, and the European Speech Communication Association. He is also a Lifetime Member of the Computational Linguistic Society in Taiwan. From 1991 to 1995 he was an Associate Editor for IEEE TRANSACTIONS ON SIGNAL PROCESSING and IEEE TRANSACTIONS ON SPEECH AND AUDIO PROCESSING. He serves as the Chairman of the IEEE Signal Processing Society Speech Processing Technical Committee. He received the IEEE Signal Processing Society Senior Award in 1994 and the IEEE Signal Processing Society Best Paper Award in 1997.