Deterministically Annealed Design of Hidden Markov Model Speech Recognizers

Ajit V. Rao and Kenneth Rose, *Member, IEEE*

***Abstract*—Many conventional speech recognition systems are based on the use of hidden Markov models (HMM) within the context of discriminant-based pattern classification. While the speech recognition objective is a low rate of misclassification, HMM design has been traditionally approached via maximum likelihood (ML) modeling which is, in general, mismatched with the minimum error objective and hence suboptimal. Direct minimization of the error rate is difficult because of the complex nature of the cost surface, and has only been addressed recently by discriminative design methods such as generalized probabilistic descent (GPD). While existing discriminative methods offer significant benefits, they commonly rely on local optimization via gradient descent whose performance suffers from the prevalence of shallow local minima. As an alternative, we propose the deterministic annealing (DA) design method that directly minimizes the error rate while avoiding many poor local minima of the cost. DA is derived from fundamental principles of statistical physics and information theory. In DA, the HMM classifier's decision is randomized and its expected error rate is minimized subject to a constraint on the level of randomness which is measured by the Shannon entropy. The entropy constraint is gradually relaxed, leading in the limit of zero entropy to the design of regular nonrandom HMM classifiers. An efficient forward–backward algorithm is proposed for the DA method. Experiments on synthetic data and on a simplified recognizer for isolated English letters demonstrate that the DA design method can improve recognition error rates over both ML and GPD methods.**

***Index Terms*—Deterministic annealing, discriminative training, hidden Markov model, isolated word recognition, minimum classification error.**

I. Introduction

In THE late 1960s, Baum and his colleagues presented a series of papers (including [4], [5]) investigating the mathematical structure and practical usefulness of Hidden Markov models (HMMs). In the years that followed, it became generally known that HMMs can be usefully employed for speech recognition. This important realization is due to the pioneering work of several researchers, notably Jelinek [17], Baker [3], Ferguson [13] and Rabiner [29], [30]. Since then, the HMM-based classifier has steadily replaced template matching as the main paradigm for speech recognition.

In HMM-based speech recognition, the input speech is divided into segments, each of which is classified to an element of a finite length *dictionary* of *speech units*. Speech units may be words (in isolated word recognition) or subword phones (in continuous speech recognition). Classification is performed on each segment via competition between HMMs that represent speech units in the dictionary. In isolated word recognition, the segments correspond to words, and the task of dividing the input speech into segments is usually performed prior to, and independent of, the classification task. In continuous speech recognition, segments correspond to subword units and the tasks of segmentation and classification are performed jointly.

For superior performance, an HMM speech recognizer must be trained on a large speech database. Since recognition is performed by competition between HMMs, ideally, the training procedure should *jointly* optimize all competing HMMs to minimize the training error rate, which is defined as the fraction of the training set that is misclassified. In isolated word recognition, the error rate corresponds to the fraction of training words that are misclassified. In continuous speech recognition, the error rate may be measured as the fraction of training sentences that contain any recognition errors.

One important design difficulty is the complex nature of the error rate cost surface. This surface, which represents the classifier error as a function of the HMM parameters, is piecewise constant and riddled with shallow local minima. In principle, a design procedure seeks the system parameters that globally minimize this cost surface, but standard optimization methods such as gradient descent will normally fail to produce the optimal solution. The prevailing approach to speech recognizer design circumvents this difficulty by discarding the minimum classification error (MCE) objective, and adopting instead the potentially mismatched maximum likelihood (ML) objective. The choice of ML leads to a smoother cost surface and also facilitates independent optimization of each HMM via the efficient Baum–Welch algorithm [4], [5].

The common justification for the employment of the ML criterion is its asymptotic equivalence to the MCE criterion, which is valid if infinite training data is available and if the HMM structure is a precise model for speech production. Unfortunately, in practical speech recognition problems, neither of the above assumptions hold. Thus, there is an (at least theoretical) advantage to direct optimization of the MCE rather than the ML criterion.

The shortcomings of ML have been recognized by several researchers (e.g., [1], [2], [6], [18], [7]) who provided strong

Manuscript received April 16, 1999; revised May 1, 2000. This work was supported in part by the National Science Foundation under Grant NCR-9314335, the University of California MICRO program, Cisco Systems, Inc., Conexant Systems, Inc., Dialogic Corp., Fujitsu Laboratories of America Inc., General Electric Company, Hughes Network Systems, Lernout and Hauspie Speech Products, Lockheed Martin, Lucent Technologies, Panasonic Technologies, Inc., Qualcomm Inc., and Texas Instruments Inc. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Rafid A. Sukkar.

A. Rao is with Microsoft Corporation, Santa Barbara, CA (e-mail: ajitrao@microsoft.com).

The authors are with the Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA 93106 USA (e-mail: rose@ece.ucsb.edu).

Publisher Item Identifier S 1063-6676(01)00846-X.

reasons for discarding it in favor of *discriminative design methods* [31, Sec 5.6] that jointly optimize all the HMMs in a classifier. One promising discriminative design method, Generalized Probabilistic Descent (GPD), was proposed and extended by Juang, Katagiri and others in a series of papers that are reviewed in [21]. Not surprisingly, GPD and other discriminative approaches are applicable to general (not necessarily speech) structured pattern classifiers. The central idea in GPD is to approximate the piecewise constant cost surface by a smooth and differentiable function. Once the cost is smoothed, gradient methods may be used to optimize the classifier's parameter set and find a local minimum on the (smoothed) surface. Discriminative methods have been applied to the design of speech recognizers based on both template matching [7], [24] and HMMs [9], [18].

In the context of HMMs, it was shown in [9] and [18] that GPD provides a significant improvement in recognition accuracy over ML design. Our starting point, however, is with the observation that while cost surface smoothing allows the use of gradient descent optimization, the smoothed cost surface is nevertheless highly complex with numerous shallow local minimum traps. Consequently, GPD may often converge to a poor local minimum and yield suboptimal recognition performance. The experimental results in this paper validate this observation.

From the preceding arguments, it is clear that direct minimization of the classifier error rate requires the use of a nonconvex optimization method that can avoid shallow local minimum traps on the cost surface. In this paper, we present a powerful optimization method that builds on the technique of deterministic annealing (DA). DA is derived from fundamental principles in statistical physics and information theory. It has already been successfully employed to solve a number of difficult optimization problems in source coding and pattern recognition (for a tutorial see [37]). The DA formulation for clustering and related problems was first proposed in [38]–[40]. Later, DA was extended to allow the inclusion and imposition of structural constraints [25]. This extension has considerably broadened the scope of its applications. Recently, DA methods have been developed to design pattern classifiers [25], regression functions [36], [34] and a new class of source coding systems [35]. DA has previously been proposed for HMM design, albeit with ML as the design objective [26].

The design of discriminative HMM-based speech recognizers requires an important and nontrivial extension of the DA approach. Our method is general and can be applied to design both isolated word and continuous speech recognizers that use discrete, continuous observation, or tied-mixture HMMs. This paper's focus is on the introduction of the basic derivation of DA for speech recognition, and preliminary demonstration of its usefulness and potential. We therefore restrict the formulation and experiments here to the simpler case of isolated word recognition with discrete observation HMM systems. The formulation is supplemented with the derivation of a low complexity forward–backward (FB) implementation, which may be viewed as a generalization of the Baum–Welch re-estimation algorithm.

The results section reports on HMM design experiments in time-series classification and in isolated word recognition. The results compare DA with ML and GPD, and demonstrate that DA offers consistent accuracy improvements over the competing methods.

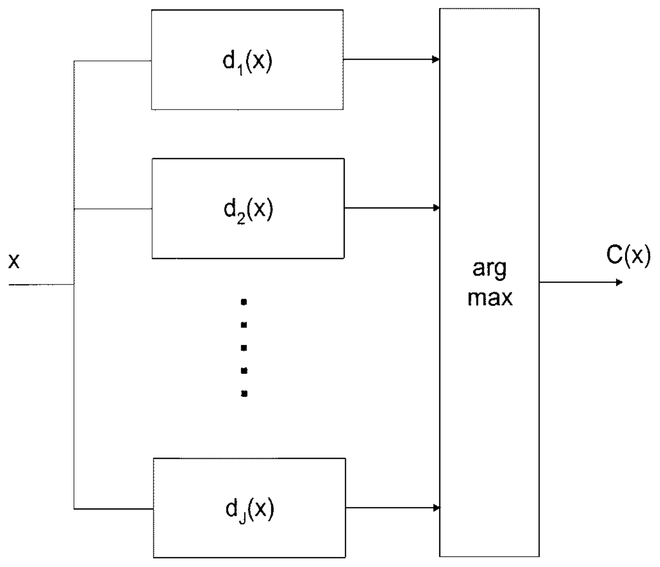


Fig. 1. An HMM-based speech recognition system viewed as a maximum discriminant classifier.

The paper is organized as follows: In the next section, we review HMM-based isolated word recognizers, present its central design issues and motivate the need for a powerful optimization technique. In Section III, we briefly review DA and derive an explicit DA algorithm for the HMM design problem, including the forward–backward implementation. The section ends with comments on extensions of the basic DA formulation to continuous speech recognizers and continuous observation HMMs. Section IV summarizes the experimental results, and Section V offers comments on the computational complexity of the DA method. In the Appendix, we establish and briefly discuss connections between DA and other discriminative design methods.

II. HMM-Based Isolated Word Recognition

The input speech is divided into fixed-length frames and a short-term feature vector (such as a vector of cepstral coefficients) is extracted per frame. Next, an end-point identification algorithm is used to divide consecutive speech frames into *segments*. Each segment is then mapped to a word in the dictionary by an HMM-based pattern recognizer. Grammatical constraints and/or language models may further be used to improve the recognition accuracy.

The pattern classification step is performed by a set of HMMs, one per word in the dictionary. The classification procedure consists, in fact, of competition between the HMMs (Fig. 1). Each HMM, H_j, computes a *class discriminant* $d_j(\mathbf{x})$ given $\mathbf{x}$, the feature vector representing a speech segment. The segment is ultimately labeled with the index of the "winner" which is the dictionary entry corresponding to the HMM with the highest class discriminant.

A. HMM Classifier Design

Obtaining good recognition performance depends to a large degree on careful training of the classifier's HMM parameters, which determine the class discriminants. During training, the system accuracy is measured by the classification error rate, i.e., the fraction of training words incorrectly recognized by the classifier. We assume that the training data has been divided into segments that correspond to isolated words and labeled with the correct dictionary entries. The classifier design problem is, therefore, to adjust the HMM parameters to minimize the error rate measured over the labeled training data. We emphasize, however, the generality of the basic approach which is not restricted to the above simplifying assumptions.

Even this simple classifier design problem is extremely difficult to solve because of the piecewise constant nature of the cost surface which prevents the use of gradient-based optimization. As mentioned in the introduction, the common approach to circumvent this difficulty is to discard the MCE criterion and adopt, instead, the potentially mismatched ML criterion. The training data is divided into subsets of identically labeled data (segments that correspond to the same dictionary entry) and an HMM is designed for each subset via maximum likelihood estimation of the HMM parameters.

B. HMM Design for Minimum Classification Error

The starting point for this work is the realization that maximum likelihood is potentially mismatched to the desired objective of HMM design. Speech recognition is fundamentally a pattern classification problem whose ultimate objective is not to accurately model utterances of particular words but, rather, to distinguish between them while making as few errors as possible. The classifier's performance is, therefore, measured most appropriately by its error rate and, if possible, this cost should be directly minimized during design via a joint optimization of all HMMs.

It is important to note that a classifier system that is designed by an ML technique is a close relative of the Bayesian classifier which is optimal in the sense of MCE. However such optimality depends on the availability of the precise probability distributions, and on the (improbable) assumption that the HMM probabilistic model is in complete agreement with the speech source. Even if the model structure were correct, in reality, one only has access to reasonably short training sets that do not allow reliable estimation of the probability distribution parameters. Consequently, the performance of ML may differ significantly from that of MCE methods. This fact has been observed by several researchers and we briefly review below some of the contributions that are particularly relevant to this work. The results section herein will also provide ample evidence to support this observation.

As an aside, it should perhaps be noted that the mismatch between the true objective and the commonly used design criterion has been pointed out for the related problems of general pattern classifier design and regression function design. This realization has led to new approaches [19], [36], [25], [34] that directly optimize the true objective and demonstrate significant improvements in performance.

In the specific context of HMM-based classifier design, the inadequacy of the ML approach has attracted much attention. One important example is the work in [1] which demonstrates the suboptimality of ML and proposes a maximum mutual information (MMI) criterion as an alternative. In MMI design, the HMM parameters are optimized such that the resulting statistical model maintains the highest possible mutual information between the feature vectors and the classes when measured over the training set. Another proposal is the method of corrective training [2] which is related to, and improves on, MMI. Corrective training applies heuristic measures to train the HMM parameters to generate high likelihood scores for the correct classes *and* simultaneously generate low likelihood scores for the incorrect classes. Experiments indicate that improvements over standard ML are obtained by MMI and, to a larger degree, by corrective training. We note that both corrective training and MMI attempt to reduce error rates indirectly by increasing class separability. Neither method attempts a direct minimization of the error rate cost function.

Another interesting approach to solve the joint optimization problem is the recursive estimation and maximization of a-posteriori probabilities (REMAP) algorithm [6]. Although REMAP was proposed in the context of the design of hybrid neural-network/HMM recognition systems, it can in principle be applied to the design of standard HMM recognizers. REMAPs design objective is to maximize the a posteriori probabilities of the correct class, given the feature vector. REMAP is claimed to have better optimality properties than ML and can be applied to design any statistical classifier. However, it relies heavily on the assumption of the validity of the statistical model (in this case, the HMM model).

From the perspective of the approach we take in this paper, a particularly promising alternative is the discriminative learning method based on generalized probabilistic descent (GPD) [19]. GPD is motivated by the understanding that a major difficulty in minimizing the error rate arises from the piecewise-constant nature of the cost surface, which implies that derivatives with respect to the design variables vanish almost everywhere. More specifically, since the training set is of finite length, an infinitesimal change in the value of the design parameters will not change the classification of any training utterance and hence will not cause any change in the error rate cost. Clearly, it is not possible to directly apply gradient based optimization. To overcome this difficulty, GPD replaces the piecewise-constant cost with a smooth and differentiable approximation to it. The smoothed cost may be minimized by a gradient descent method. The result is a local minimum which, hopefully, approximates well the performance of the globally optimal solution. The advantage of GPD is that it does not make any assumptions on the validity of the statistical model (HMM) but, instead, directly adjusts the classifier parameters to minimize the true cost. Experiments in [9] and [18] show that discriminative design provides significant improvements over ML design for HMM-based classifier systems.

However, an important and fundamental drawback of GPD is that, even after smoothing, there are numerous shallow local minima that riddle the cost surface. Consequently, gradient-based algorithms that seek a local minimum on the cost surface may easily be trapped in shallow local optima, and may produce a substantially suboptimal classifier. It is, therefore, our premise here that a powerful optimization technique will provide the means for realizing the full potential benefits of a direct minimization of the classification error.

In summary, discriminative methods such as GPD are an important step toward optimality as they target the right cost objective-minimum classification error. There are, however, substantial additional gains to be recouped by using a powerful optimization algorithm that offers the capability to avoid shallow local minima.

III. Deterministic Annealing

A. Problem Formulation

An HMM classifier is to be designed given a *labeled training set*,

$$\mathcal{T} \equiv \{(\mathbf{x}_1, c_1), (\mathbf{x}_2, c_2), \cdots, (\mathbf{x}_N, c_N)\} \tag{1}$$

where *training pattern* $\mathbf{x}_i$ is known to be an utterance of speech unit c_i, which is an entry in the given dictionary of speech unit labels $\mathcal{C} \equiv \{1, 2, \cdots, J\}$. The pattern $\mathbf{x}_i$ is in fact a sequence of observation feature vectors extracted from a segment of l_i speech frames, $\mathbf{x}_i = (\mathbf{x}_i(1), \mathbf{x}_i(2), \cdots, \mathbf{x}_i(l_i))$.

The exact nature of the observation feature vector $\mathbf{x}_i(t)$ is application dependent. In many practical implementations, $\mathbf{x}_i(t)$ consists of cepstral coefficients or linear prediction coefficients and their derivatives. Since these features take value in a continuous space ($\mathbf{x}_i(t) \in \mathcal{R}^n$), such classification is normally performed by continuous observation HMMs. In a number of applications, however, the high computational complexity involved in modeling continuous observations is not acceptable, and the classification is implemented with discrete observation HMMs. Here, the feature vector is extracted from the speech frame and then vector-quantized to an entry in a pre-designed codebook of K *prototype vectors*. The sequence of l_i quantization indexes obtained by this process is the discrete observation vector or training pattern, $\mathbf{x}_i$. The derivation in this paper assumes the discrete observation case, $\mathbf{x}_i(t) \in \mathcal{K} \equiv \{1, 2, \cdots, K\}$. However, the method is general and extendible to the case of continuous observations [15].

The HMM recognition system for discrete observations consists of a set of HMMs, $\{H_j, \; j = 1, 2, \cdots, J\}$, which correspond to the J words in the dictionary. The model H_j has S_j states and is fully specified by the parameter set $\Lambda_j \equiv (A_j, B_j, \Pi_j)$ where, following the standard notation, A_j is the $(S_j \times S_j)$ state transition probability matrix, B_j is the $(S_j \times K)$ emission probability matrix and Π_j is the $(S_j \times 1)$ initial state probability vector.

We consider HMM classifier systems that use the common "best path" discriminant approach. Note, however, that this assumption is not required, and the design method can be modified to the case where the discriminant is obtained by appropriate averaging of the likelihood over all paths in the HMM.

The best path classifier works as follows: Given a training pattern, $\mathbf{x}_i$, for each HMM, H_j, and for each sequence (length l_i) of states, $\mathbf{s} \equiv (s(1), s(2), \cdots, s(l_i))$ in the trellis of H_j, we determine the quantity ("path score")

$$l(\mathbf{x}_i, \mathbf{s}, H_j) = \frac{1}{l_i}\left\{\log \Pi_j[s(1)] + \sum_{t=1}^{l_i-1} \log A_j[s(t), s(t+1)] + \sum_{t=1}^{l_i} \log B_j[s(t), \mathbf{x}_i(t)]\right\}, \tag{2}$$

which is the normalized log of the joint probability of observation $\mathbf{x}_i$ and the state sequence, $\mathbf{s}$ given the parameters of H_j. We use the conventional notation: $Q[\cdot, \cdot]$ denotes an element of matrix Q; and $q[\cdot]$ denotes an element of the vector q. We note that although normalization of the likelihood (by the length of the observation) does not change the problem definition and is not commonly used, we find it useful for the DA algorithm description.

Next, we maximize the path score over all paths in the trellis of H_j and determine the score of model H_j:

$$d_j(\mathbf{x}_i) = \max_{\mathbf{s} \in \mathcal{S}_{l_i}(H_j)} l(\mathbf{x}_i, \mathbf{s}, H_j) \tag{3}$$

where $\mathcal{S}_l(H_j)$ is the set of all state sequences of length l in the trellis of H_j. The quantity $d_j(\mathbf{x}_i)$ approximates the likelihood of model H_j given the observation, $\mathbf{x}_i$. Interpreting $d_j(\cdot)$ as the discriminant for class j, we adopt the traditional discriminant-based classification rule (Fig. 1):

$$C(\mathbf{x}_i) = \arg\max_j d_j(\mathbf{x}_i). \tag{4}$$

The classification procedure can be viewed as a competition between paths. The HMM containing the path with the highest score is declared "winner" and the classifier assigns pattern $\mathbf{x}_i$ to the corresponding class, or word in the dictionary. A known advantage of the "best path" discriminant classifier is that the search for the winning path can be reduced to a sequential optimization problem that can be solved via an efficient dynamic programming algorithm.

The HMM-based classifier should, in principle, be optimized by adjusting the HMM parameters $\{\Lambda_j\}$ to minimize the empirical misclassification rate measured over the training set:

$$\min_{\{\Lambda_j\}}\left\{P_e = 1 - \frac{1}{N}\sum_{i=1}^{N} \delta(C(\mathbf{x}_i), c_i)\right\}. \tag{5}$$

Here δ is the Kronecker delta function:

$$\delta(u, v) = \begin{cases} 1, & \text{if } u = v \\ 0, & \text{otherwise.} \end{cases}$$

The introduction section of this paper provides a detailed discussion of the difficulties in solving the above optimization problem. These difficulties are due to the piecewise constant nature of the cost function, P_e, and the abundance of shallow local minima.

B. Randomized Classification Rule

We adopt the DA formulation whose fundamental principles are: a) Introduce randomness in the classification rule during the design process; b) Minimize the expected misclassification rate of the random classifier while controlling the level of randomness via a constraint on the Shannon entropy; and c) Gradually relax the entropy constraint so that the effective cost converges to the misclassification cost at the limit of zero entropy (nonrandom classification).

Thus, we replace the original (nonrandom) best path classification rule with a randomized classification rule. While the nonrandom rule assigns a pattern $\mathbf{x}_i$ to a unique winning state sequence, the randomized rule associates each pattern, $\mathbf{x}_i$, with every state sequence, $\mathbf{s}$, in the trellis of every model, H_j, with probability $P[\mathbf{s}, j|\mathbf{x}_i]$. Naturally, these conditional probabilities are normalized functions such that $\sum_j \sum_{\mathbf{s}\in\mathcal{S}_{l_i}(H_j)} P[\mathbf{s}, j|\mathbf{x}_i] = 1$.

The probabilities, $P[\mathbf{s}, j|\mathbf{x}_i]$, are in fact, the representation of the *randomized classification rule* and should not be confused with the probabilities characterizing the HMM model itself. $P[\mathbf{s}, j|\mathbf{x}_i]$ is the probability that the classifier will select $\mathbf{s}$ as the winning path and, consequentially, H_j as the winning HMM. We propose to derive the classification probabilities from basic principles. We first note that the nonrandom classifier takes in pattern $\mathbf{x}_i$ and finds the state sequence $\mathbf{s}_i$ with the highest score among all state sequences in all HMMs, in order to determine the class. We may trivially formulate this operation via the criterion function

$$D_e = \frac{1}{N}\sum_i l(\mathbf{x}_i, \mathbf{s}_i, H(\mathbf{s}_i)) \tag{6}$$

where $\mathbf{s}_i \in \bigcup_j \mathcal{S}_{l_i}(H_j)$ and $H(\mathbf{s}_i)$ is the HMM to which $\mathbf{s}_i$ belongs. Clearly, this function is maximized by applying to each $\mathbf{x}_i$ the best-path classification rule:

$$\mathbf{s}_i = \arg \max_{\mathbf{s}\in\bigcup_j \mathcal{S}_{l_i}(H_j)} l(\mathbf{x}_i, \mathbf{s}, H(\mathbf{s})). \tag{7}$$

We next define the optimal random classifier as the distribution that maximizes:

$$\langle D_e\rangle = \frac{1}{N}\sum_i \sum_j \sum_{\mathbf{s}\in\mathcal{S}_{l_i}(H_j)} P[\mathbf{s}, j|\mathbf{x}_i] l(\mathbf{x}_i, \mathbf{s}, H_j) \tag{8}$$

which is the immediate probabilistic generalization of D_e in (6). Note that if we simply maximize $\langle D_e\rangle$ over all distributions, $P[\mathbf{s}, j|\mathbf{x}_i]$, we will still obtain the rule of (7) that assigns, with probability 1, pattern $\mathbf{x}_i$ to the path with the highest likelihood score. While the best-path rule will ultimately be used once the design process is complete, it is important to realize that, during design, it is advantageous to maintain randomness in the classifier decision as it makes the design more robust to shallow local minima. Toward this end, we propose instead to maximize $\langle D_e\rangle$ subject to a constraint on the level of randomness in the classification rule, which we measure by the (conditional) Shannon entropy,

$$H = -\frac{1}{N}\sum_i \sum_j \sum_{\mathbf{s}\in\mathcal{S}_{l_i}(H_j)} P[\mathbf{s}, j|\mathbf{x}_i] \log P[\mathbf{s}, j|\mathbf{x}_i]. \tag{9}$$

More specifically, we maximize the Lagrangian

$$\gamma\langle D_e\rangle + H + \sum_i \lambda_i \sum_{j,\mathbf{s}} P[\mathbf{s}, j|\mathbf{x}_i],$$

where the last term is used to impose that the distribution is normalized to 1. By straightforward differentiation and constraint imposition we obtain that the optimal probability distribution is the Gibbs distribution,

$$P[\mathbf{s}, j|\mathbf{x}_i] = \frac{e^{\gamma l(\mathbf{x}_i, \mathbf{s}, H_j)}}{\sum_{j'} \sum_{\mathbf{s}'\in\mathcal{S}_{l_i}(H_{j'})} e^{\gamma l(\mathbf{x}_i, \mathbf{s}', H_{j'})}}. \tag{10}$$

The level of Shannon entropy corresponding to this Gibbs distribution is determined by the positive *scale parameter* γ. For $\gamma = 0$, the distribution over paths is uniform. For finite, positive values of γ, the Gibbs distribution indicates that we assign higher probabilities of winning to state sequences with higher likelihood scores. In the limiting case of $\gamma \to \infty$, the random classification rule reverts to the nonrandom "best path" classifier, which assigns a nonzero probability of winning only to the path with the highest likelihood score as in (7).

We re-emphasize that the random classifier paradigm is adopted only during design. Ultimately, the DA algorithm will produce a regular, nonrandom HMM classifier which is based on the best-path discriminant.

C. The Effective Cost Function and the Statistical Physics Analogy

So far we have derived a framework for randomizing the classifier, which captures the "best path" classification rule in the limiting (zero entropy) case. We now apply this framework to actually minimize the error rate of the classifier. The average misclassification rate of the random classifier is given by:

$$\langle P_e\rangle = 1 - \frac{1}{N}\sum_{i=1}^{N} P[c_i|\mathbf{x}_i] \tag{11}$$

which is a straightforward randomization of (5). The quantity $P[c_i|\mathbf{x}_i]$ is the probability that the correct class c_i will be selected as winner, and can be computed by summation over paths:

$$P[c_i|\mathbf{x}_i] = \sum_{\mathbf{s}\in\mathcal{S}_{l_i}(H_{c_i})} P[\mathbf{s}, c_i|\mathbf{x}_i]. \tag{12}$$

Thus, the design problem for the random classifier can be stated as follows: Find the optimal values of the model parameters $\{\Lambda_j\}$ and γ, which determine $P[\mathbf{s}, c_i|\mathbf{x}_i]$ so as to minimize the misclassification probability (11).

Note that direct minimization of (11) would lead to a nonrandom ($\gamma \to \infty$) distribution. This may be deduced by analyzing the gradient of the error rate with respect to the scale parameter, γ:

$$\frac{\partial\langle P_e\rangle}{\partial\gamma} = \frac{1}{N}\sum_{i=1}^{N} P[c_i|\mathbf{x}_i]\{E[l(\mathbf{x}_i, \mathbf{s}, H_{c_i})] - E[l(\mathbf{x}_i, \mathbf{s}, H_{c_i})|\mathbf{s}\in\mathcal{S}_{l_i}(H_{c_i})]\}, \tag{13}$$

where $E[\cdot]$ denotes expectation with respect to the Gibbs distribution of (10). We assume that the HMM parameter set $\{\Lambda_j\}$ is "reasonable," that is, the expected path discriminant for paths in the trellis of the "correct" HMM, $E[l(\mathbf{x}_i, \mathbf{s}, H_{c_i})|\mathbf{s} \in \mathcal{S}_{l_i}(H_{c_i})]$, is on the average (over the training set) greater than the expected path discriminant over all paths in all HMMs, $E[l(\mathbf{x}_i, \mathbf{s}, H_{c_i})]$. Hence, (13) is negative and drives γ to infinity to produce a nonrandom distribution. Of course, such a nonrandom, best-path classifier is the ultimate goal of the design procedure. However, as mentioned earlier, we wish to enforce this "nonrandomness" gradually during the optimization, to avoid shallow local minimum traps.

We, therefore, pose the problem of minimizing $\langle P_e\rangle$ while maintaining a level of randomness in the classifier through a constraint on the entropy: $H = H_0$. This constrained optimization problem is, equivalently, the minimization of the unconstrained Lagrangian cost function,

$$\min_{\{\Lambda_j\},\gamma} \{F \equiv \langle P_e\rangle - TH\} \tag{14}$$

where T is the Lagrange multiplier which is referred to as the "temperature" to allude to an interesting analogy to statistical physics: The Lagrangian minimization of (14) is analogous to the classical definition of thermal equilibrium in statistical physics. The quantity, F, is the Helmholtz free energy of a thermodynamic system (strictly speaking it is the Helmholtz thermodynamic potential) with average energy $\langle P_e\rangle$, entropy H, and temperature T. A fundamental principle of statistical mechanics states that the free energy is minimized when the thermodynamic system reaches thermal equilibrium at given temperature T. From the viewpoint of our optimization problem, we are ultimately interested in thermal equilibrium at $T = 0$ which corresponds to direct minimization of $\langle P_e\rangle$, our ultimate objective.

D. Annealing

The statistical physics analogy suggests that, in order to minimize $\langle P_e\rangle$, it is beneficial to implement an annealing process, that is, gradually lower the temperature while maintaining the system at thermal equilibrium. This process, of course, gradually reduces the entropy, or randomness, of the system. We start at a high level of T, where the sole objective is entropy maximization, which is achievable by the uniform distribution. We then gradually reduce T while tracking the minimum of F. At $T = 0$, the optimization of F seeks the desired solution which is the minimum of $\langle P_e\rangle$ with respect to $\{\Lambda_j\}$ and γ. For practical reasons, it is efficient to end the annealing procedure with a "quenching" step—when T falls below a threshold we increase γ in gradual steps to a very high value. When γ is sufficiently high, the classifier reduces to the nonrandom "best-path" classifier.

The algorithm can be summarized as follows:

1) Set parameters: initial temperature T_i, final temperature, T_f, minimum entropy, $H_{\min}$, annealing schedule function $\alpha(\cdot)$ and quenching schedule function, $q(\cdot)$.
2) Set $T = T_i$, $\gamma = \gamma_{init}$.
3) $\min_{\{\Lambda_j\},\gamma}\{F \equiv \langle P_e\rangle - TH\}$ (We use gradient descent in our simulations)
4) Lower temperature: $T \leftarrow \alpha(T)$.
5) If $T > T_f$ goto step 3; else goto step 6;
6) Quenching: Increase γ according to $\gamma \leftarrow q(\gamma)$, $\min_{\{\Lambda_j\}} \langle P_e\rangle$
7) If $H > H_{\min}$ goto step 6; else end design.

In our experiments, we used the following simple exponential annealing and quenching schedules: $\alpha(T) = 0.9T$, and $q(\gamma) = 1.2\gamma$. An analytical treatment of the question of annealing schedules has not been attempted as yet.

The annealing process yields a sequence of solutions at decreasing levels of entropy and $\langle P_e\rangle$ leading to a "best-path" classifier in the limit. The optimization of the Lagrangian, F, at each temperature is accomplished by a series of gradient descent steps based on the gradients:

$$\frac{\partial F}{\partial \Lambda_j} = \frac{\gamma}{N}\sum_i \sum_{\mathbf{s}\in\mathcal{S}_{l_i}(H_j)} \frac{\partial l(\mathbf{x}_i, \mathbf{s}, H_j)}{\partial \Lambda_j} \cdot P[\mathbf{s}, j|\mathbf{x}_i]\{f(\mathbf{x}_i, \mathbf{s}, H_j) - \langle f(\mathbf{x}_i, \mathbf{s}, H_j)\rangle\} \tag{15}$$

and

$$\frac{\partial F}{\partial \gamma} = \frac{1}{N}\sum_i \sum_j \sum_{\mathbf{s}\in\mathcal{S}_{l_i}(H_j)} l(\mathbf{x}_i, \mathbf{s}, H_j) \cdot P[\mathbf{s}, j|\mathbf{x}_i]\{f(\mathbf{x}_i, \mathbf{s}, H_j) - \langle f(\mathbf{x}_i, \mathbf{s}, H_j)\rangle\}. \tag{16}$$

Here, $f(\mathbf{x}_i, \mathbf{s}, H_j) = T\gamma l(\mathbf{x}_i, \mathbf{s}, H_j) - \delta(j, c_i)$ where $\delta(\cdot, \cdot)$ is the Kronecker delta function. The operation $\langle h(\cdot)\rangle$ represents the expectation of function $h(\cdot)$ taken over all state sequences in the trellises of all HMMs. Hence,

$$\langle f(\mathbf{x}_i, \mathbf{s}, H_j)\rangle = \sum_j \sum_{\mathbf{s}\in\mathcal{S}_{l_i}(H_j)} P[\mathbf{s}, j|\mathbf{x}_i] f(\mathbf{x}_i, \mathbf{s}, H_j). \tag{17}$$

E. Forward–Backward Algorithm

An important aspect of the proposed method is the discovery of an efficient forward–backward (FB) algorithm to determine the gradients in (15) and (16). Note that the summation in both gradient expressions is taken over all state sequences in the trellis of the HMMs. Since the number of state sequences grows exponentially with the number of states in the HMM, it is impractical to compute each gradient by simply summing up the contributions of individual state sequences.

The basis of the FB algorithm is a simple mathematical manipulation, which reveals that the gradients can be efficiently evaluated via a forward–backward calculation that drastically reduces the number of computations. The resulting computational and memory complexity is proportional to the square of the number of states in the HMM. Thus, the complexity scales similarly to that of maximum likelihood.

The FB algorithm for computing the gradient parameters is implemented in the following manner: For each training pattern, $\mathbf{x}_i$, and each class, j, we perform an initialization, a forward pass, and a backward pass as shown in Table I. In the forward pass, the forward variables, $r_{ij}(t, m)$ and $u_{ij}(t, m)$ are computed for all the states, $m = 1, 2, \cdots, S_j$ in a sequential manner for $t = 1, 2, \cdots, l_i$. In the backward pass, the backward variables, $s_{ij}(t, m)$ and $v_{ij}(t, m)$ are computed for these states in a reverse-time sequential manner, $t = l_i, \cdots, 2, 1$. Next, the "transition variables," $\omega_{ij}(t, m, n)$ and $\xi_{ij}(t, m, n)$ are computed for each transition from state m to state n for $t = 1, 2, \cdots, l_i$.

For the purpose of interpreting the forward and backward variables, it is useful to first define two quantities that we refer to as "partial discriminants":

$$l_1^{(t)}(\mathbf{x}_i, \mathbf{s}, H_j) = \frac{1}{l_i}\left\{\log \Pi_j[s(1)] + \sum_{t'=1}^{t-1} \log A_j[s(t'), s(t'+1)] + \sum_{t'=1}^{t} \log B_j[s(t'), \mathbf{x}_i(t')]\right\}, \tag{18}$$

TABLE I

COMPUTATION OF FORWARD, BACKWARD AND TRANSITION VARIABLES IN THE DETERMINISTIC ANNEALING ALGORITHM

Initialization

```
Scale parameter: $\gamma_i = \gamma / l_i$
Forward variables: $r_{ij}(1,m) = (\Pi_j[m]B_j[m,\mathbf{x}_i(1)])^{\gamma_i}$
                   $u_{ij}(1,m) = r_{ij}(1,m)\log(\Pi_j[m]B_j[m,\mathbf{x}_i(1)])$
Backward variables: $s_{ij}(l_i,m) = B_j[m,\mathbf{x}_i(l_i)]^{\gamma_i}$
                    $v_{ij}(l_i,m) = s_{ij}(l_i,m)\log B_j[m,\mathbf{x}_i(l_i)]$
$(i = 1..N,\ j = 1..J,\ m = 1..S_j)$
```

Forward pass

```
$Repeat for\ t = 2,3,..l_i.\{$
    $Repeat for\ m = 1,2,..S_j.\{$
        $\alpha_n = A_j[n,m]B_j[m,\mathbf{x}_i(t)] \quad n = 1,2,..S_j$
        $r_{ij}(t,m) = \sum_{n=1}^{S_j} r_{ij}(t-1,n){\alpha_n}^{\gamma_i}$
        $u_{ij}(t,m) = \sum_{n=1}^{S_j} {\alpha_n}^{\gamma_i}(u_{ij}(t-1,n) + r_{ij}(t-1,n)\log\alpha_n)$
    $\}$
$\}$
```

Backward pass

```
$Repeat for\ t = l_i - 1,..2,1.\{$
    $Repeat for\ m = 1,2,..S_j.\{$
        $\beta_n = A_j[m,n]B_j[m,\mathbf{x}_i(t)] \quad n = 1,2,..S_j$
        $s_{ij}(t,m) = \sum_{n=1}^{S_j} s_{ij}(t+1,n){\beta_n}^{\gamma_i}.$
        $v_{ij}(t,m) = \sum_{n=1}^{S_j} {\beta_n}^{\gamma_i}(v_{ij}(t+1,n) + s_{ij}(t+1,n)\log\beta_n).$
    $\}$
$\}$
```

Computation of transition variables

$$\omega_{ij}(t,m,n) = A_j[m,n]^{\gamma_i} r_{ij}(t,m)s_{ij}(t+1,n),$$
$$\xi_{ij}(t,m,n) = \frac{A_j[m,n]^{\gamma_i}}{l_i}(u_{ij}(t,m)s_{ij}(t+1,n) + r_{ij}(t,m)v_{ij}(t+1,n)) + \omega_{ij}(t,m,n)\log A_j[m,n]$$

and

$$l_2^{(t)}(\mathbf{x}_i,\, \mathbf{s},\, H_j) = \frac{1}{l_i}\left\{\sum_{t'=t}^{l_i-1}\log A_j[s(t'),\, s(t'+1)] + \sum_{t'=t}^{l_i}\log B_j[s(t'),\, \mathbf{x}_i(t')]\right\}. \quad (19)$$

The partial discriminant $l_1^{(t)}(\mathbf{x}_i,\, \mathbf{s},\, H_j)$ represents the contribution of the first t frames to the discriminant of state sequence $\mathbf{s}$ in H_j for observation $\mathbf{x}_i$. Similarly, the second partial discriminant $l_2^{(t)}(\mathbf{x}_i,\, \mathbf{s},\, H_j)$ represents the contribution from frame t in $\mathbf{x}_i$ until the end of the observation.

To explain the FB variables, it is convenient to begin with descriptions of $r_{ij}(\,)$, $s_{ij}(\,)$, and $\omega_{ij}(\,)$. Following this, we proceed to describe the remaining variables, namely, $u_{ij}(\,)$, $v_{ij}(\,)$, and $\xi_{ij}(\,)$. Mathematically,

$$r_{ij}(t,\, m) = \sum_{\mathbf{s}\in\mathcal{U}(t,\, j,\, m)} e^{\gamma l_1^{(t)}(\mathbf{x}_i,\, \mathbf{s},\, H_j)}. \quad (20)$$

The summation in the above equation is performed over all paths in the set $\mathcal{U}(t,\, j,\, m)$ which represents all sequences in the trellis of H_j which pass through state m at time t, i.e. $\mathcal{U}(t,\, j,\, m) = \{\mathbf{s}\colon \mathbf{s} \in \mathcal{S}_{l_i}(H_j),\, \mathbf{s}(t) = m\}$.

The backward variable corresponding to $r_{ij}(t,\, m)$ is

$$s_{ij}(t,\, m) = \sum_{\mathbf{s}\in\mathcal{U}(t,\, j,\, m)} e^{\gamma l_2^{(t)}(\mathbf{x}_i,\, \mathbf{s},\, H_j)}. \quad (21)$$

Both $r_{ij}(\,)$ and $s_{ij}(\,)$ have simple interpretations in terms of path probabilities: The quantity $r_{ij}(t,\, m)$ represents the *un-normalized* partial probability given the first t frames of observation $\mathbf{x}_i$, that the winning state sequence for $\mathbf{x}_i$ will pass through state m in H_j at time t. By "un-normalized," we mean that the variable represents the numerator of a Gibbs distribution, and that the denominator (the normalization constant) can be computed to ensure that the appropriate probabilities add up to unity at each time instant t and for each training vector, $\mathbf{x}_i$. A similar interpretation can be offered for $s_{ij}(t,\, m)$—this quantity is the un-normalized partial probability given the last $l_i - t + 1$ frames in $\mathbf{x}_i$, that the winning state sequence for $\mathbf{x}_i$, will pass through state m in H_j at time t.

The quantities $r_{ij}(\,)$ and $s_{ij}(\,)$ have the important property that they can be computed iteratively: $r_{ij}(\,)$ using forward iterations and $s_{ij}(\,)$ using backward iterations. (These iterative equations are given in Table I). From $r_{ij}(\,)$ and $s_{ij}(\,)$, we compute transition variables, $\omega_{ij}(t, m, n)$, which are defined as:

$$\omega_{ij}(t, m, n) = \sum_{\mathbf{s}\in\mathcal{V}(t, j, m, n)} e^{\gamma l(\mathbf{x}_i, \mathbf{s}, H_j)}. \tag{22}$$

The above summation is evaluated over paths in the set $\mathcal{V}(t, j, m, n)$, which represents all state sequences in H_j which traverse through state m at time t and state n at time $t+1$, i.e. $\mathcal{V}(t, j, m, n) = \{\mathbf{s}\colon \mathbf{s} \in \mathcal{S}_{l_i}(H_j), \mathbf{s}(t) = m, \mathbf{s}(t+1) = n\}$ The quantity $\omega_{ij}(t, m, n)$ represents the un-normalized probability that the winning state sequence for observation $\mathbf{x}_i$ passes through state m at t and state n at $t+1$ (both states in H_j).

We now define $u_{ij}(\,)$ and $v_{ij}(\,)$:

$$u_{ij}(t, m) = \sum_{\mathbf{s}\in\mathcal{U}(t, j, m)} l_1^{(t)}(\mathbf{x}_i, \mathbf{s}, H_j) e^{\gamma l_1^{(t)}(\mathbf{x}_i, \mathbf{s}, H_j)} \tag{23}$$

and

$$v_{ij}(t, m) = \sum_{\mathbf{s}\in\mathcal{U}(t, j, m)} l_2^{(t)}(\mathbf{x}_i, \mathbf{s}, H_j) e^{\gamma l_2^{(t)}(\mathbf{x}_i, \mathbf{s}, H_j)}. \tag{24}$$

The quantity $u_{ij}(t, m)$ may be interpreted as the "un-normalized" conditional expectation of $l_1^{(t)}(\mathbf{x}_i, \mathbf{s}, H_j)$ given the first t frames in $\mathbf{x}_i$ and given that the the winning state sequence for training vector, $\mathbf{x}_i$, will pass through state m in H_j at time t. Similarly, $v_{ij}(t, m)$, may be interpreted as the ("un-normalized") conditional expectation of $l_2^{(t)}(\mathbf{x}_i, \mathbf{s}, H_j)$ given the last $l_i - t + 1$ observations in $\mathbf{x}_i$ and given that the the winning state sequence for training vector, $\mathbf{x}_i$, will pass through state m in HMM H_j at time t.

Finally, we define

$$\xi_{ij}(t, m, n) = \sum_{\mathbf{s}\in\mathcal{V}(t, j, m, n)} l(\mathbf{x}_i, \mathbf{s}, H_j) e^{\gamma l(\mathbf{x}_i, \mathbf{s}, H_j)}. \tag{25}$$

The quantity $\xi_{ij}(t, m, n)$ represents the conditional expectation of the discriminant (albeit un-normalized) given that the winning state sequence for $\mathbf{x}_i$ will pass through state m at time t and state n at time $t+1$ (both states in H_j).

The variables u_{ij} and v_{ij} are computed in an iterative manner in the FB algorithm. The $\xi_{ij}(\,)$ variables are computed from $u_{ij}(\,)$, $v_{ij}(\,)$, $r_{ij}(\,)$ and $s_{ij}(\,)$.

Interestingly, $r_{ij}(t, m)$ and $s_{ij}(t, m)$ reduce to the standard FB variables in Baum–Welch re-estimation for the degenerate special case of $\gamma = 1$ (ignoring the normalization by l_i).

To obtain the gradient parameters from the FB variables, the following quantities are defined:

$$\Xi_i = \sum_{j=1}^{J} \sum_{m=1}^{S_i} \sum_{n=1}^{S_i} \xi_{ij}(1, m, n), \tag{26}$$

$$\Omega_i = \sum_{j=1}^{J} \sum_{m=1}^{S_i} \sum_{n=1}^{S_i} \omega_{ij}(1, m, n), \tag{27}$$

and

$$D_{ij} = T\gamma \frac{\Xi_i}{\Omega_i} + \left(\delta(c_i, j) - \frac{\sum_{m=1}^{S_i}\sum_{n=1}^{S_i} \omega_{ic_i}(1, m, n)}{\Omega_i} \right). \tag{28}$$

The gradient parameters are given by:

$$\frac{\partial F}{\partial A_j[m,n]} = \frac{\gamma}{NA_j[m,n]} \sum_{i=1}^{N} \frac{\sum_{t=1}^{l_i-1} \omega_{ij}(t,m,n)}{l_i\Omega_i} \cdot \left\{ T\gamma \frac{\sum_{t=1}^{l_i-1} \xi_{ij}(t,m,n)}{\sum_{t=1}^{l_i-1} \omega_{ij}(t,m,n)} - D_{ij} \right\}, \tag{29}$$

$$\frac{\partial F}{\partial B_j[m,k]} = \frac{\gamma}{NB_j[m,k]} \sum_{i=1}^{N} \frac{\sum_{t;\mathbf{x}_i(t)=k} \sum_{n=1}^{S_i} \omega_{ij}(t,m,n)}{l_i\Omega_i} \cdot \left\{ T\gamma \frac{\sum_{t;\mathbf{x}_i(t)=k} \sum_{n=1}^{S_i} \xi_{ij}(t,m,n)}{\sum_{t=1;\mathbf{x}_i(t)=k} \sum_{n=1}^{S_i} \omega_{ij}(t,m,n)} - D_{ij} \right\}, \tag{30}$$

and

$$\frac{\partial F}{\partial \Pi_j[m]} = \frac{\gamma}{N\Pi_j[m]} \sum_{i=1}^{N} \frac{\sum_{n=1}^{S_i} \omega_{ij}(1,m,n)}{l_i\Omega_i} \cdot \left\{ T\gamma \frac{\sum_{n=1}^{S_i} \xi_{ij}(1,m,n)}{\sum_{n=1}^{S_i} \omega_{ij}(1,m,n)} - D_{ij} \right\}. \tag{31}$$

We note that the scale parameter, γ, can also be optimized via a gradient descent, with the gradient computed through the FB algorithm. However, in our experiments, we found it simpler to use a line search scheme [28] for γ optimization.

Generally, the value of γ increases as the temperature is reduced, At high temperatures, the emphasis is on maximizing the entropy. The optimization of γ accomplishes this objective by settling at a small value. As the temperature is increased, the line search prefers larger values of γ. The chosen value represents a trade-off between the necessity to keep entropy high (small γ) and the objective of reducing classifier errors (large γ). When annealing ends at zero temperature, γ increases to a high value in order to minimize $\langle P_e \rangle$. The final quenching step drives γ to infinity leading to the nonrandom classification rule.

F. Gradient Descent for Free Energy Minimization

To fully exploit the optimization capabilities of the DA approach, it is important to perform the free energy minimization steps with an effective and efficient implementation of gradient descent. Several variations on the general idea of gradient descent are possible (see [16] for a discussion on this topic). In our experiments, we obtained good results by using descent steps that were implemented in the log probability domain. The equation for each iteration of this method is given by

$$A_j[m, n] \leftarrow g_a(j, m) A_j[m, n] e^{-g_s(\partial F / \partial \log A_j[m, n])} \quad (32)$$

$$B_j[m, k] \leftarrow g_b(j, m) B_j[m, k] e^{-g_s(\partial F / \partial \log B_j[m, k])} \quad (33)$$

$$\Pi_j[m] \leftarrow g_p(j) \Pi_j[m] e^{-g_s(\partial F / \partial \log \Pi_j[m])}. \quad (34)$$

In the above equations, g_s is a scale parameter for the gradient, and $g_a(j, m)$, $g_b(j, m)$, and $g_p(j)$, are normalizing constants, which ensure that the relevant probabilities add up to unity after the gradient descent. The gradient with respect to any parameter, Λ_j in the log domain is given by

$$\frac{\partial F}{\partial \log \Lambda_j} = \Lambda_j \frac{\partial F}{\partial \Lambda_j}. \quad (35)$$

G. Extensions of the DA Derivation

This subsection briefly summarizes possible extensions, some of which have already been addressed to a limited degree, and others which still await investigation.

1) Continuous Observation HMMs: The basic DA approach that we have presented in this paper can be extended to design more complex HMM recognizers. The extension of DA to design classifiers based on continuous density HMMs or semi-continuous (tied-mixture) HMMs is possible. The latter case requires some nontrivial but largely understood extensions of the basic approach.

The underlying principle of the DA approach is the same as in the discrete observation case. However, the main difference is that while we optimize the state dependent discrete emission probabilities (represented by the B matrix) in discrete observation systems, in the continuous case we optimize means and co-variances of state-dependent multimodal density functions. For tied-mixture HMMs, optimization is over a codebook of densities (Gaussians) and a matrix of mixing weights. Preliminary experiments in this direction indicate that the DA extension to design continuous observation HMMs [15] outperforms ML and achieves good recognition results. Extensions to tied-mixture HMMs are currently under development and investigation.

2) Connected Word and Continuous Speech Recognition: We envisage another important and nontrivial extension of the DA approach to design connected word and continuous speech recognition systems. Although no experimental work has been performed as yet, it appears safe to predict that the theory developed here for isolated word recognition is extendible to tackle the connected word and continuous speech problems. In both connected word and continuous speech recognition, the objective is to minimize the error rate in classifying an observation composed of a concatenated string of speech units. A speech unit refers to either a word (connected word recognition) or a subword unit such as a phone (continuous speech recognition). In both cases, recognition is performed as follows: HMMs are designed for individual speech units and segments of the input utterance are each classified via assignment to an HMM model that represents a speech unit. The observation is thus mapped to an output string of speech units. An important aspect of the problem is that the task of dividing the observation into segments must be performed *in conjunction with*, rather than *independent of*, the task of classifying the segments.

In principle, recognition is performed by first implicitly generating models for each allowed segment string by concatenating HMM models of subword units that the string is composed of. Next, the output string model with the highest likelihood ("path score") is determined through a search procedure. Since the objective is to correctly identify the entire string of words that composes an observation, the design objective is stated as the minimization of the string error rate (the fraction of training strings that are assigned to the wrong string models).

We envisage the following extension of the original DA formulation to solve this problem: Given each observation, recognition is viewed as a competition between allowed word strings. Each class j in the DA formulation is interpreted as a "word string" instead of a word (the usual isolated word interpretation). The DA optimization is derived in the usual way, with misclassification error as the design objective.

One concern in the above formulation is the large number of allowed word strings (classes) that will result with the use of even a few subword models. However, the complexity of the problem can be drastically reduced by using fully connected string models (all subword model sequences are allowed) and extending the FB algorithm.

IV. Experimental Results

In this section, we report the results of our experiments comparing the proposed DA method for HMM design with two conventional design approaches—standard maximum likelihood (ML) and Generalized Probabilistic Descent (GPD). We first briefly comment on our implementation of each competing design method. Next, we present the results of the experiments, which are divided into two categories: The first set of experiments demonstrates performance on synthetic time series data, and the second is a set of experiments in the recognition of English letters spoken in isolation.

A. Comments on Implementation

1) Maximum Likelihood: The ML method was implemented in the conventional way using the Baum–Welch algorithm. First, the observations were uniformly segmented into states, and the state dependent emission probabilities were initialized from the segmental histogram of observations in each state. The transition probabilities were initialized randomly, using a uniform distribution over the allowed interval (0 to 1). The prior probabilities were fixed to be consistent with the L-R nature of the HMMs. Fifty iterations of the maximum likelihood algorithm were run. During optimization, the values of all probabilities were constrained to lie in the interval $(10^{-6}, 1 - 10^{-6})$.

2) Deterministic Annealing: The DA method was implemented according to the description in Section III, with the following specifications: initial value of the scale parameter, $\gamma_{init} = 0.1$, initial temperature, $T_i = 1.0$, final temperature, $T_f = 10^{-6}$, annealing schedule, $\alpha(T) = 0.9T$, quenching schedule, $q(\gamma) = 1.2\gamma$, and minimum entropy, $H_{\min} = 10^{-6}$. The detailed information of parameter choices is provided here not because of its importance but to facilitate exact reproduction of the experiments. Prior to performing DA iterations, the HMMs were each initialized to "flat" models—all transition probabilities were chosen to be uniform and all emission probability distributions were chosen to be identical to the state-independent histogram of observations for the corresponding class. The value of the initial temperature T_i was simply "high enough" based on our experimental observation that the HMM models did not significantly change from the flat initial models, if the temperature, T was greater than 1.0.

The gradient descent steps were performed in the log domain, as described in Section III-F. For efficient implementation, the scale parameter, γ was fixed at each temperature and re-optimized via a line search procedure after the completion of the optimization steps for the HMM parameters $\{H_j\}$ at this temperature. Each gradient descent step was performed simultaneously on all HMM parameters. The scale parameter for the gradient (g_s defined in Section III-F) was chosen using a simple line search procedure. Following each gradient descent step, all gradients were recomputed using the FB algorithm. Gradient descent steps were continued until the fractional improvement in the free energy was found to be less than a threshold of 4.0×10^{-5}.[1] At this point, γ was optimized using line search, and the temperature was lowered (annealing) before more gradient descent steps were performed.

3) Generalized Probabilistic Descent: The GPD method was implemented as a special case of DA—the temperature was set to zero, the value of γ was fixed by the level of smoothing desired, and free energy minimization steps were performed via gradient descent in the log domain, similar to DA.[2]

We recognize that ours is one of many possible implementations of GPD. We chose this implementation to facilitate a direct comparison of the design principles behind GPD and DA. In the course of our experiments with GPD, we observed that the performance of GPD depends significantly on both the initialization for the HMM parameters (as expected) and the value chosen for γ (the smoothness parameter).

To improve the GPD results, we allowed 20 different values for the parameter γ and two different initializations for the HMM parameters. The allowed values of γ were chosen according to the geometric series, $1.0, 2.0, 4.0, 8.0, \cdots, 524\,288.0$ which allows for a large variation in its value, and which effectively is a "post"-optimization of γ. Two initial sets of ML-designed models were each obtained from different random initializations of the transition probabilities. For each initialization, the same emission probabilities were chosen (state-specific histograms following uniform segmentation). ML was implemented as fifty steps of the Baum–Welch algorithm. In the literature, ML-designed models have often been used as initialization for GPD.

Each GPD model was thus obtained by running 40 different GPD experiments—using 20 values of γ and two initial sets of HMMs—and choosing the best of the 40 solutions in terms of performance on the training set.

[1]When the free energy changes from $F_{\rm old}$ to $F_{\rm new}$ in one gradient descent step, the fractional improvement in free energy is defined as $F_{\rm old} - F_{\rm new}/F_{\rm old}$.

[2]As mentioned in Section III, when $T = 0$ and γ is fixed, the free energy of DA is equivalent to the GPD cost function.

TABLE II
A COMPARISON OF TRAINING AND TEST ERROR RATES OBTAINED FOR HMM CLASSIFIERS DESIGNED USING ML, GPD AND DA METHODS FOR 8 DIFFERENT SYNTHETIC DATASETS. THE TRAINING SETS (TR) ARE OF SIZE 2000 AND THE TEST SETS (TE) ARE OF SIZE 10 000. EACH DATASET CONSISTS OF DATA FROM $K = 2, 3$, OR 4 CLASSES. THE LAST COLUMN SHOWS THE ERROR OF THE OPTIMAL (BAYES) CLASSIFIER

Method		ML		GPD		DA		Bayes
Dataset	K	TR	TE	TR	TE	TR	TE	
1	2	17.4	19.0	17.4	19.0	6.5	8.5	7.2
2	2	31.6	31.8	31.5	31.2	21.7	25.7	24.3
3	2	26.5	27.6	25.8	27.3	18.7	23.2	21.9
4	3	28.7	29.4	28.4	29.2	20.9	24.6	23.7
5	3	27.0	27.7	26.4	27.3	21.0	21.4	20.0
6	3	32.5	32.3	31.7	31.7	27.3	30.8	26.8
7	3	24.9	22.7	24.8	22.7	17.4	19.4	19.6
8	4	42.3	43.0	42.0	42.9	31.7	36.6	33.5

B. Synthetic Data Sets

We first experimented with the classification of synthetic time series data. The experiments were performed over eight different data sets, each consisting of 2000 labeled time sequences of discrete observations. Each data set was generated by a system of J HMMs ($J = 2, 3$ or 4 depending on the data set). Each data-generating HMM had three, four or five states. The HMMs used in the classifier were, however, constrained to have only three states. The introduction of some modeling mismatch is appropriate since, in real-world situations, we do not have access to the true model that generated the data. The number of states that we allow in each HMM during classification is typically determined in an ad-hoc manner. The three methods, ML, GPD and DA were applied to the training sets. The error rate was measured on the training data and on an independent test set of 10 000 time sequences. The classification error results are tabulated in Table II. Clearly, the proposed DA approach improves the classification performance consistently and significantly. Improvements over ML and GPD were obtained over both training and test data for all data sets. In summary, the DA design method was found to reduce the error rate measured over the training set by a factor of 1.3 to 2.5 compared to the ML and GPD methods. The test set improvement factors were in the range of 1.05 to 1.5. Note that in most cases, the test set error rate achieved by the DA approach was very close to the error rate of the optimal Bayes classifier, which is also provided in the Table for reference.

C. Spoken Letter Recognition

The recognition of spoken English letters appears as an important subproblem within several applications [20], [23] including automatic car navigation, automated directory assistance and voice activated call forwarding.

Speech recognizers based on HMMs [33], [20], dynamic time warping (DTW) [32], neural networks [12] and knowledge-based classifiers [10], [23] have been proposed to tackle

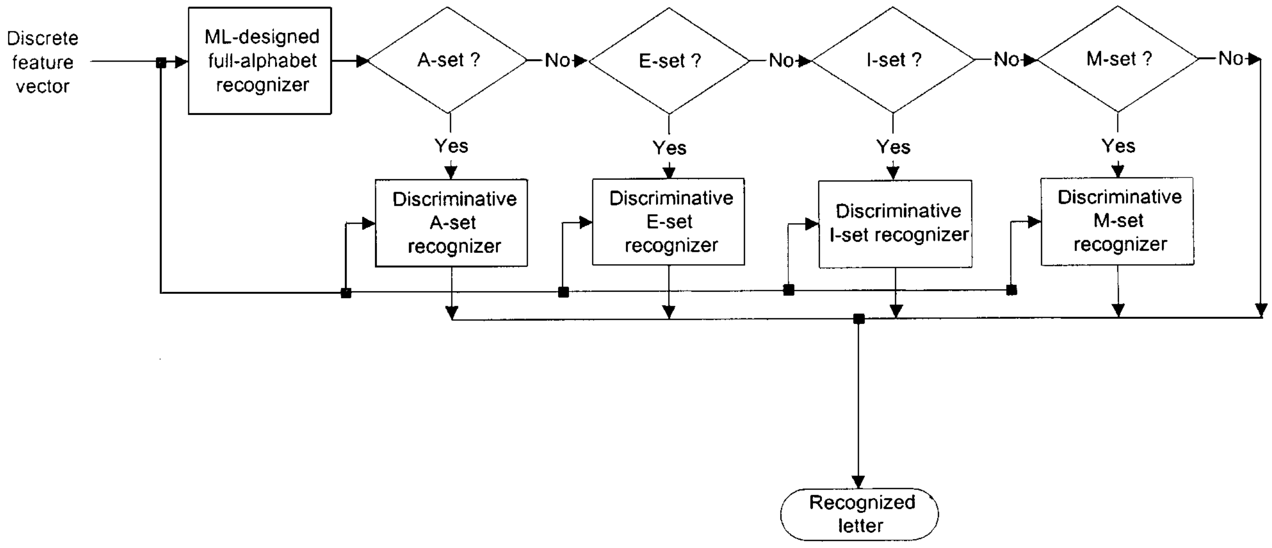

Discrete feature vector
ML-designed full-alphabet recognizer
A-set ?
No
E-set ?
No
I-set ?
No
M-set ?
No
Yes
Yes
Yes
Yes
Discriminative A-set recognizer
Discriminative E-set recognizer
Discriminative I-set recognizer
Discriminative M-set recognizer
Recognized letter

Fig. 2. Block diagram of a multipass full alphabet recognition system.

this important problem. In [33] Rabiner and Wilpon present a comparison of different approaches on benchmark data sets.

The task of recognizing spoken English letters is known to be challenging due to the high confusability of the alphabet. Four subsets of English letters are highly confusable: The E-set $\{b, c, d, e, g, p, t, v, z\}$, the A-set, $\{a, k, j\}$, the I-set $\{i, r, y\}$, and the M-set $\{m, n\}$. Most notorious for its confusability is the E-set. In real-world situations, the difficulty is further aggravated by the presence of background noise and, possibly, by the presence of channel distortion.

1) Speech Database and Feature Extraction: The speech used in our experiments is drawn from the ISOLET database [11]. ISOLET consists of utterances by 150 native English speakers (75 male and 75 female, ages 14 to 70 years), where each speaker utters, twice, the 26 English letters in isolation. The data is divided into five sets (ISOLET1 to ISOLET5), each containing utterances by 30 speakers. The ISOLET speech was recorded with a Sennheiser HMD 224 microphone that was low-pass filtered at 7.6 kHz and sampled at 16 kHz.

In our experiment, we used this database to design and test speech recognizers in the "multispeaker" mode—one set of utterances of all the letters by the 60 speakers in ISOLET1 and ISOLET2 was used as the training set and the other set of utterances by the same speakers were used as the test set.

The sampled speech signal was divided into 20 ms frames, where consecutive frames overlap by 10 ms. A set of features were extracted from each speech file by using a public domain software for end-point detection and feature extraction.[3] The end-point detector was employed to eliminate silent frames at the beginning and the end of the speech file. A 28-dimensional feature vector consisting of 14 Mel-scaled FFT cepstral coefficients (MFCC) [27] and their first-order time derivatives (ΔMFCC coefficients) was extracted in each frame. The MFCC coefficients are believed to be relatively robust to noise and can be easily computed via an FFT. The feature vector in each frame was quantized using a codebook of "prototypes," thus resulting in a sequence of discrete features for each isolated word. In our experiments, we used either 16 or 32 prototypes. We refer to the observations quantized to 16 prototypes as Data16 and those quantized to 32 prototypes as Data32. The codebook of prototypes was designed from features extracted from all speech frames in the training set using a vector quantizer design method based on successive splitting of prototypes [14, Ch. 11].

We recognize that the front-end feature extraction process does not reflect the current state-of-the-art, and improvement of this part of the system may lead to significantly better classification rates for all competing methods, but this is secondary and orthogonal to our focus in this paper, which is on the fundamental HMM training technique.

2) Recognizer Design: The objective of our experiment is to design context-independent whole-word models for the letters using each design method and to compare the error performance. Each word model consists of a 5-state left-to-right HMM.

First, we designed an HMM classifier for the entire alphabet using the ML approach. As observed before, the main problem in using standard ML appears to be the confusion between members within the E, A, I, and M sets. A straightforward remedy would be to use a discriminative design approach such as GPD or DA to design a recognizer for the entire alphabet. However the complexity of discriminative methods is proportional to the square of the number of classes, and is further aggravated by the necessity to use a very large training set to obtain good performance outside the training set.

To reduce the complexity of the experiments we instead considered a two-pass classifier that can efficiently exploit the power of the discriminative design methods. The structure of the proposed system is motivated by the observation that the main difficulty in alphabet recognition lies within the confusable sets and, hence, almost all the advantages of discriminative design can be reaped if it can be effectively used to distinguish between letters within each of the confusable sets.

A block diagram of the two-pass recognition system is shown in Fig. 2. The system works as follows: First, the discrete fea-

[3]We used J. He's speech recognition research tool which can be obtained via anonymous ftp from ftp.informatik.uni-ulm.de/pub/NI/jialong/spchtool.zip.

TABLE III
A COMPARISON OF ERROR RATES OBTAINED IN CLEAN SPEECH CONDITIONS FOR EACH CONFUSABLE SET (E-SET, A-SET, I-SET, AND M-SET). TRAINING (TR) AND TEST (TE) ERRORS ARE SHOWN FOR (a) DATA 16 AND (b) DATA32

Dataset	Set	ML		GPD		DA	
		TR	TE	TR	TE	TR	TE
Data16	E	49.4	60.2	47.6	58.9	25.2	42.8
	A	12.2	18.3	6.7	20.0	3.9	9.4
	I	10.6	12.8	4.4	8.9	0.6	7.2
	M	15.0	26.7	12.5	21.7	7.5	15.0

(a)

Dataset	Set	ML		GPD		DA	
		TR	TE	TR	TE	TR	TE
Data32	E	32.6	45.9	31.1	46.3	20.4	45.4
	A	10.6	20.6	3.3	17.8	0.6	11.1
	I	1.7	5.0	0.6	4.7	0.6	4.7
	M	10.0	21.7	7.5	23.3	2.5	15.9

(b)

TABLE IV
A COMPARISON OF ERROR RATES OBTAINED BY DESIGNING MULTISTAGE HMM RECOGNIZERS FOR THE ENGLISH ALPHABET. TRAINING (TR) AND TEST (TE) ERROR ARE SHOWN FOR CLEAN, WHITE NOISE AND CAR NOISE BACKGROUND CONDITIONS FOR BOTH DATA16 AND DATA32

Background	Dataset	ML		GPD		DA	
		TR	TE	TR	TE	TR	TE
Clean	Data16	32.5	40.5	31.5	40.1	23.7	34.4
	Data32	20.3	29.9	18.7	30.0	15.0	29.5
White Noise	Data16	34.4	40.4	34.2	40.5	28.9	38.1
	Data32	27.1	32.7	23.7	31.9	18.6	30.8
Car Noise	Data16	34.5	40.5	31.7	39.4	25.0	36.7
	Data32	25.2	30.6	21.7	30.4	15.9	28.1

tures extracted from the speech signal are presented to a standard ML-designed HMM classifier for the entire alphabet. If the input is mapped by the classifier to a letter outside all the confusable sets, the winning letter is declared as the output of the classifier (only one pass is used). If instead, the input is mapped to a letter inside any of the confusable sets, then a second classifier is invoked. The second classifier is one that has been previously optimized by a discriminative design method (either GPD or DA) to distinguish effectively between tokens within the appropriate confusable set. If a second pass is used, the class that wins in this pass is declared as the output class. Clearly, the second pass that uses either DA or GPD trained HMMs can improve on the performance of the first pass.

The results of our experiments comparing the design methods are presented in Tables III and IV. To obtain the results in Table III, we designed and tested classifiers for each confusable set using only tokens from that confusable set. The design was performed independently for each confusable set, on Data16 and Data32 using ML, GPD and DA. The objective of this comparison is to demonstrate the type of improvement achievable in each confusable set. The results indicate that the DA method improves significantly over both ML and GPD design methods on both training and test sets. The improvements are more substantial for Data16 than for Data32, especially over the test set.

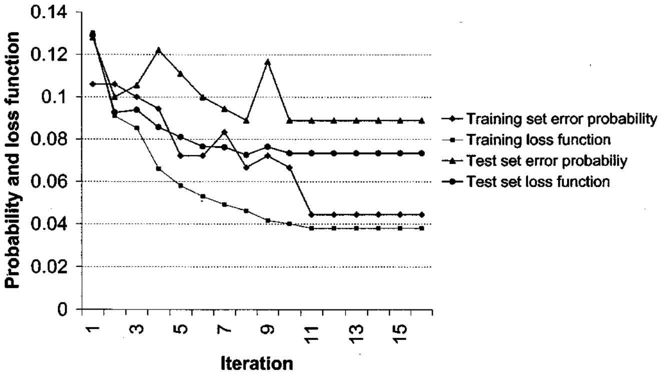


Fig. 3. Evolution of the loss function and the misclassification probability on the training and test sets during GPD design.

We have also performed some preliminary experiments on the robustness of the recognizer to background noise. We first generated two noisy databases from the ISOLET database by adding synthetic white noise and recorded car noise (at a -15 dB energy level relative to the speech signal) to the clean speech files. (Of course, the utterances themselves were recorded in a noise-free environment and the speaker's reaction to the presence of acoustic noise is neglected). We next designed a multipass recognizer for the clean condition as well as for each noisy condition. The recognizer for each of the noisy conditions was designed from a training set of speech files with the appropriate noise synthetically added. The results, in terms of classifier errors obtained for each method over the entire alphabet, are tabulated in Table IV, which also shows the error rates obtained for car and white noise against the those obtained for clean speech. The recognizer designated as "ML" represents a one-pass system that uses only an ML-designed full-alphabet recognizer (the first stage of the system shown in Fig. 2). The "GPD" recognizer compared in this table was obtained by using the two-pass recognizer of Fig. 2, wherein, "ML" was used in the first pass and the second pass classifiers were each designed using the GPD approach. The "DA" recognizer was also a two-pass recognizer with "ML" used in the first pass, but its second pass classifiers were each designed using the proposed DA approach. The "ML" "GPD" and "DA" recognizers were designed independently on Data16 and Data32, and for clean, white noise and car noise conditions. The results indicate that significant performance gains can be obtained by using DA instead of ML and GPD methods in both clean and noisy speech conditions.

D. Graphical Illustration of GPD and DA

Figs. 3 and 4 are graphic illustrations of the typical evolution of the error rates during GPD and DA design of HMM classifiers. Both figures correspond to the recognition of I-set letters in the clean background for the Data16 set. This particular case is chosen only as an illustration. Fig. 3 shows the evolution of the loss function ($\langle P_e \rangle$) and the true error rate (P_e) during a GPD

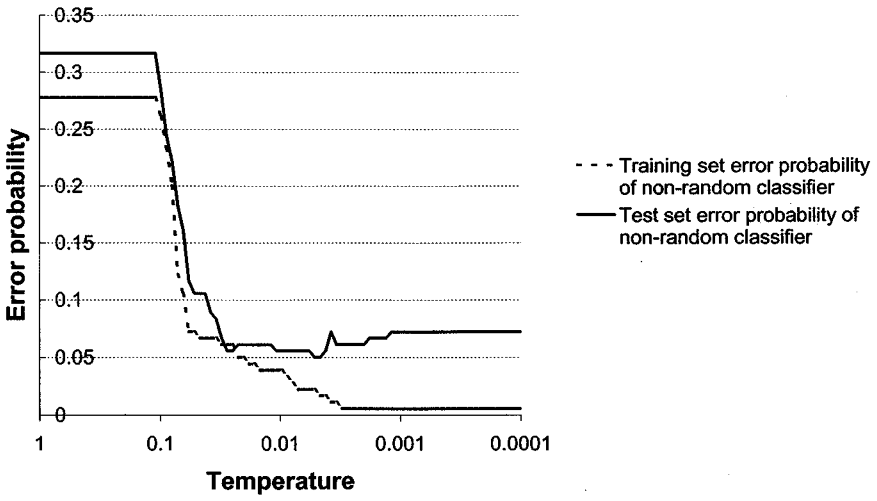


Fig. 4. Evolution of the training and test set error probabilities during DA design.

run. The initial solution which is an ML-designed model corresponds to iteration index, 1. From Fig. 3, it is clear that $\langle P_e \rangle$ measured on the training set decreases monotonically in each successive iteration of the algorithm. The training and test set error rates also improve, although not monotonically. The optimization is terminated when the fractional improvement in $\langle P_e \rangle$ is smaller than a threshold. In this example, the algorithm terminated in iteration 16. The final GPD error rate was 4.4% on the training set and 8.9% on the test set.

Fig. 4 illustrates the evolution of the error rate in DA, for the same dataset. The initial temperature was set to $T_i = 1$. As annealing begins, the error rate continues to remain at the initial (high) value until the temperature is reduced to $T = 0.1$. At this point, there is a sudden change in system parameters and the training error rate decreases to 7%. We refer to this temperature, as the first "critical temperature" of the system. Critical temperatures, which are data-dependent, are associated with significant drops in the free energy of the system, and are typical characteristics of DA optimization. The change in system parameters at the critical temperatures is often viewed as the equivalent of "phase transitions" in physical annealing [38]. Following the first phase transition, the training error rate briefly appears to stabilize at 7%, before a gradual drop ensues from $T = 0.03$ until $T = 0.003$, when the error settles at it's final value of 0.6%. During the rest of the annealing and during quenching, there was no change in the error rate on this dataset. The test set error rate, which is also shown in Fig. 4, decreases to 10% during the first phase transition and finally settles at 7.2%. DA performed significantly better than GPD on this dataset.

V. A Note on Computational Complexity

The improvements in recognition performance of DA over GPD and ML are obtained at the cost of additional computational complexity. We emphasize that the additional complexity is entailed only for the design process. The DA method has no effect on the computational complexity of the classifier itself. Table V shows a comparison of the average computation times in seconds (total elapsed real time from beginning to end of simulation) required for each design algorithm. The comparison is performed over the speech recognition examples of different sizes (2, 3, and 9 classes) and over different data sets (Data16 and Data32). The 2-class case represents the M-set, the 3-class case represents an average over the "I" and the "A" sets and the 9-class case represents the E-set. In the case of GPD, we account for time required to run the design method with multiple initializations as needed to achieve good quality of results. On average, the computational complexity of DA is 86 times that of ML and 3.7 times that of GPD.

TABLE V
A Comparison of Average Execution Time (in Seconds) for ML, GD and DA Designs. Comparisons are Performed Over Speech Recognition Problems of Different Sizes (Two, Three, or Nine Classes). The 2 Class Averages were Obtained from the M-Set Classifier Design, The 3-Class from the I and A Sets, and the 9-Class from the E-Set Classifier Design

Dataset	Classes	ML	GPD	DA
Data16	2	30	574	2309
	3	74	1514	6928
	9	582	22280	58345
Data32	2	28	404	1872
	3	74	1354	6171
	9	589	25426	63863

All simulations were performed on a Dell Pentium II PC with a 350 MHz clock and 128 Mbytes of RAM, under the Linux operating system. The software was written in C and compiled using the GNU gcc compiler.

VI. Conclusions

A novel training algorithm for designing HMM-based speech recognizers was presented. The proposed training method directly minimizes the recognizer's misclassification error rate unlike the commonly used maximum likelihood approach. At the heart of this method is the deterministic annealing optimization strategy. The new method can be used to design isolated word and continuous speech recognition systems based on both continuous and discrete observation HMMs. The effectiveness of the novel design method was tested on synthetic data as well as on the problem of training a simple recognizer of isolated English letters. Our tests compared the error rates obtained by using DA against those obtained by using the popular maximum likelihood approach and the recently proposed generalized probabilistic descent method. Significant improvements in error rates were obtained by using DA. The performance improvements were obtained at the cost of a manageable increase in design complexity. A detailed investigation into the effectiveness of applying DA to continuous speech recognition is currently in progress.

Appendix
Connections with Other Discriminative Design Methods

In this appendix, we discuss some interesting mathematical relationships between DA optimization and the mathematical formulations of other discriminative design methods. As a first step in establishing these relationships, we consider links between the probabilities that define the random classifier in DA ($P[\mathbf{s}, j|\mathbf{x}_i]$) and the standard generation probabilities of observations based on the stochastic hidden Markov model.

The HMM defines the joint probability $\tilde{p}[\mathbf{s}, \mathbf{x}_i|j]$ that state sequence $\mathbf{s}$ and observation $\mathbf{x}_i$ are generated if the word corresponding to class j is uttered.[4] This joint probability is related in a simple manner to the normalized log likelihood parameter, $l(\mathbf{x}_i, \mathbf{s}, H_j)$, which we interpreted simply as a path score in the DA formulation. Specifically,

$$\tilde{p}[\mathbf{s}, \mathbf{x}_i|j] = e^{l_i l(\mathbf{x}_i, \mathbf{s}, H_j)}. \tag{36}$$

Assuming a uniform prior $\tilde{p}(j) = 1/J$, we can compute the *a posteriori* probability:

$$\tilde{p}[\mathbf{s}, j|\mathbf{x}_i] = \frac{e^{l_i l(\mathbf{x}_i, \mathbf{s}, H_j)}}{\sum\limits_{j'} \sum\limits_{\mathbf{s}' \in \mathcal{S}_{l_i}(H_{j'})} e^{l_i l(\mathbf{x}_i, \mathbf{s}', H_{j'})}}. \tag{37}$$

Note that (37) is strikingly similar to the Gibbs probability mass function in (10):

$$\tilde{p}[\mathbf{s}, j|\mathbf{x}_i] = P[\mathbf{s}, j|\mathbf{x}_i] \qquad \text{if} \quad \gamma = l_i \tag{38}$$

although this particular choice of γ is very arbitrary from the viewpoint of DA. Further, by marginalization we get:

$$\tilde{p}[j|\mathbf{x}_i] = \sum_{\mathbf{s} \in \mathcal{S}_{l_i}(H_j)} P[\mathbf{s}, j|\mathbf{x}_i]. \tag{39}$$

Assuming that the HMMs are in fact the correct models for the observed patterns, we can compute the Bayes classifier whose classification error probability is given by:

$$\tilde{P}_e = 1 - \frac{1}{N} \sum_{i=1}^{N} \tilde{p}[c_i|\mathbf{x}_i] \tag{40}$$

where c_i is the correct class for observation $\mathbf{x}_i$. A natural design objective is to minimize $\tilde{P}_e$. This is precisely the goal of the REMAP design method [6]. It is interesting to compare REMAPs objective cost function (40) with the DA method's random classification error in (11), especially in light of the relationship derived in (38).

One difference between the two design methods is that in REMAP, unlike DA, the "scale parameter" γ is not optimized during training, but fixed to l_i (different for each observation). Another difference is the entropy constrained minimization of $\langle P_e \rangle$ in DA; entropy constrained optimization is important for avoiding poor, locally minimal solutions. The fundamental difference is that DA targets directly the MCE objective without assuming that HMMs are accurate models for speech. It simply seeks the best choice of parameters to minimize the classification error.

Another popular discriminative design method that is related to REMAP is MMI [1]. Here too the underlying assumption is that the basic data generation model (system of HMMs) is correct. To overcome the shortcomings of the ML design method, a new design objective is proposed where an information-theoretic mutual information measure is evaluated over the observation and the classes. Specifically, the MMI objective is

$$\max_{\{\Lambda_j\}} \left\{ I = \frac{1}{N} \sum_{i=1}^{N} I(j, \mathbf{x}_i) \right\} \tag{41}$$

where $I(\cdot, \cdot)$ represents the mutual information of two random variables. Assuming that all classes are equi-probable, the MMI objective can be equivalently stated as

$$\min_{\{\Lambda_j\}} \left\{ 1 - \frac{1}{N} \sum_{i=1}^{N} \log \tilde{p}[c_i|\mathbf{x}_i] \right\}. \tag{42}$$

Comparing (42) with (40), it is clear that the MMI objective function is similar to REMAPs, with the exception of the log-based nonlinear scaling of the probabilities that is used in MMI. One can interpret log scaling as a means to increase the relative importance of misclassified observations (those associated with smaller value of $\tilde{p}[c_i|\mathbf{x}_i]$) on the cost function, and thereby improve the efficiency of the design method.

Let us recall the relationship we derived in (38), in order to interpret the MMI objective from the DA viewpoint. MMI consists, roughly, of minimizing the log of the random classifier's misclassification probability, averaged over the training set. However, as in REMAP, γ is not optimized during MMI design, but fixed at l_i. Moreover, no entropy constrained optimization, or annealing, is performed.

Note that both REMAP and MMI rely on a fundamental assumption that the underlying stochastic model (HMMs) is valid. Both methods attempt to optimize all the HMM models jointly, although neither directly targets a minimization of the classifier's error rate. In contrast, Generalized Probabilistic Descent (GPD) [18], completely discards the stochastic model suggested by HMM. Instead, it views the HMM classifier simply as a maximum discriminant based mapping and attempts to optimize the discriminant parameters (HMM parameters). Since the true misclassification cost function is piece-wise constant and therefore difficult to optimize, GPD attempts to minimize a smooth approximation to this cost. In this sense, the GPD principle is more closely related to DA than it is to the other discriminative methods.

[4]Henceforth, we use the "tilde" symbol while referring to probabilities that are based on the stochastic hidden Markov model and thereby distinguish them from probabilities associated with the random classifier.

We now proceed to derive the GPD method and relate it's design objective to the free energy minimization of DA. In GPD, the piecewise constant classification error cost surface is smoothed at many levels. At the lowest (optional) level, the hard decision of choosing the path with the highest likelihood in an HMM is replaced by a soft-max function. Thus, instead of defining the discriminant for class j as the likelihood of the most likely path in an HMM, the following soft discriminant may be used:

$$g_j(\mathbf{x}, H_j) = \log\left\{\frac{1}{|\mathcal{S}_{l_i}(H_j)|}\sum_{\mathbf{s}\in\mathcal{S}_{l_i}(H_j)} e^{\alpha l(\mathbf{x},\mathbf{s},H_j)}\right\}^{1/\alpha}. \tag{43}$$

A second level of smoothness is introduced in GPD through the definition of the *class misclassification measure*. Such a measure replaces the hard decision of a maximum discriminant classifier by one that assigns nonzero weights to all classes depending on the class discriminants. The standard definition of this quantity in GPD approaches is

$$d_j(\mathbf{x}) = -g_j(\mathbf{x}, H_j) + \log\left\{\frac{1}{J-1}\sum_{k,\,k\neq j} e^{\eta g_k(\mathbf{x}, H_k)}\right\}^{1/\eta}. \tag{44}$$

The third and final level of smoothing in GPD is via the use of the "loss function." Although a number of different choices for this function have been suggested, the most common one (see e.g., [18]) is

$$\mathcal{L}_j(\mathbf{x}) = \frac{1}{1+e^{(-\kappa d_j(\mathbf{x})+\theta)}}. \tag{45}$$

In GPD, the overall performance of a classifier is measured by

$$\mathcal{L} = \sum_i \mathcal{L}_{c_i}(\mathbf{x}) \tag{46}$$

which must be *minimized* during design by optimizing the classifier parameters $\{H_j\}$.

GPD uses three different smoothing parameters (α, η, and κ), and a bias parameter, θ. The values of these parameters are usually chosen heuristically. In some GPD implementations, including the experiments in [8] and [7], the smoothing parameters were chosen to be identical. Under these special circumstances, it is possible to establish a connection between the re-estimation formulae of GPD and those of DA. Specifically, the GPD design objective is equivalent to the minimization of the average classification error $\langle P_e\rangle$ in the DA formulation if two special conditions hold. First, the three GPD smoothness parameters are chosen to be equal, i.e. $\alpha = \eta = \kappa = \gamma$, where γ is the scale parameter used in DA. Second, the bias parameter is set to $\theta = -\log(J-1)$, where J is the number of classes (words).

At this point, we wish to re-iterate that unconstrained minimization of the random classifier's classification error $\langle P_e\rangle$ is equivalent, in the DA context, to attempting free energy minimization at zero temperature without performing any annealing. Thus, under the special circumstances described above, GPD is identical to the final step in DA optimization, i.e. achieving thermal equilibrium at zero temperature. Note, however, that in DA, the iterations at zero temperature start from HMM models inherited from the optimizations performed at higher temperatures. In contrast, the GPD models are usually initialized heuristically; one common initialization for GPD is maximum likelihood designed models.

Another important difference is the role of the scale parameter γ. While γ is chosen heuristically in GPD, the value of γ at zero temperature in DA is inferred from its optimal value at the previous temperature. Moreover, γ is optimized in DA, while it remains fixed in GPD iterations.

Interestingly, the final sequence of "quenching steps" that we have proposed at the end of the DA procedure may be viewed as a series of GPD iterations performed as the smoothness parameters, α, η and κ are gradually increased. (To the best of our knowledge, such a scheme has not been proposed for GPD.) The objective of quenching is to gradually eliminate any residual smoothness in the free energy thereby ensuring that the DA cost function converges ultimately to the actual misclassification error of the classifier.

REFERENCES

[1] L. R. Bahl, P. F. Brown, P. V. DeSouza, and R. L. Mercer, "Maximum mutual information estimation of hidden Markov model parameters," in *Proc. Int. Conf. Acoustic, Speech, Signal Processing*, vol. 1, 1986, pp. 49–52.

[2] ——, "A new algorithm fot the estimation of hidden Markov model parameters," in *Proc. Int. Conf. Acoustic, Speech, Signal Processing*, 1988, pp. 493–496.

[3] J. K. Baker, "Stochastic modeling for automatic speech recognition," in *Speech Recognition*, D. R. Reddy, Ed. New York: Academic, 1975, pp. 521–542.

[4] L. E. Baum and T. Petrie, "Statistical inference for probabilistic functions of finite state Markov chains," *Ann. Math. Statist.*, vol. 37, pp. 1554–1563, 1966.

[5] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains," *Ann. Math. Statist.*, vol. 41, no. 1, pp. 164–171, 1970.

[6] H. Bourlard, Y. Konig, N. Morgan, and C. Ris, "A new training algorithm for hybrid HMM/ANN speech recognition systems," in *Signal Processing VIII, Theories Applications, Proc. EUSIPCO-96, 8th Eur. Signal Processing Conf.*, vol. 1, pp. 101–104.

[7] P.-C. Chang and B.-H. Juang, "Discriminative training of dynamic programming based speech recognizers," *IEEE Trans. Speech Audio Processing*, vol. 1, pp. 135–143, Apr. 1993.

[8] ——, "Discriminative template training for dynamic programming speech recognizers," in *Proc. Int. Conf. Acoustic, Speech, Signal Processing*, vol. 1, 1992, pp. 493–496.

[9] W. Chou, B.-H. Juang, and C.-H. Lee, "Segmental GPD training of HMM based speech recognizer," in *Proc. Int. Conf. Acoustics, Speech, Signal Processing*, vol. 1, 1992, pp. 473–476.

[10] R. Cole, R. Stern, and M. Lasry, "Performing fine phonetic distinctions: Templates vs. Frames," in *Proc. Int. Conf. Acoustic, Speech, Signal Processing*, 1982, pp. 558–561.

[11] R. Cole, Y. Muthuswamy, and M. Fanty, "The ISOLET spoken letter database," Oregon Graduate Institute, Tech. Rep. 90-004, 1990.

[12] M. Fanty and R. Cole, "Spoken letter recognition," in *Proc. Neural Information Processing Systems Conf.*, Nov. 1990, pp. 45–51.

[13] J. D. Ferguson, "Hidden Markov analysis: An introduction," in *Hidden Markov Models for Speech*. Princeton, NJ: Inst. Defense Analysis, 1980.

[14] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Norwell, MA: Kluwer, 1992.

[15] C. Gelin-Huet, K. Rose, and A. Rao, "The deterministic annealing approach for discriminative continuous HMM design," in *Proc. Eurospeech*, vol. 7, 1999, pp. 2717–2720.

[16] J.-S. R. Jang, C.-T. Sun, and E. Mizutani, *Neuro-Fuzzy and Soft Computing: A Computer Approach to Learning and Machine Intelligence*. Englewood Cliffs, NJ: Prentice-Hall, 1997.

[17] F. Jelinek, "Continuous speech recognition by statistical methods," *Proc. IEEE*, vol. 64, pp. 532–556, Apr. 1972.

[18] 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, May 1997.

[19] B.-H. Juang and S. Katagiri, "Discriminative learning for minimum error classification," *IEEE Trans. Signal Processing*, vol. 40, pp. 3043–3054, Dec. 1992.

[20] J.-C. Junqua, "SmarTspelL: A multipass recognition system for name retrieval over the telephone," *IEEE Trans. Speech Audio Processing*, vol. 5, pp. 173–182, Mar. 1997.

[21] S. Katagiri, B.-H. Juang, and C.-H. Lee, "Pattern recognition using a family of design algorithms based upon the generalized probabilistic descent method," *Proc. IEEE*, vol. 86, pp. 2345–2373, Nov. 1998.

[22] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Commun.*, vol. 28, pp. 84–95, Jan. 1980.

[23] P. C. Loizou and A. S. Spanias, "High performance alphabet recognition," *IEEE Trans. Speech Audio Processing*, vol. 4, pp. 430–445, Nov. 1996.

[24] E. McDermott and S. Katagiri, "Prototype-based minimum classification error/generalized probabilistic descent training for various speech units," *Comput. Speech Lang.*, vol. 8, pp. 351–368, Oct. 1994.

[25] D. Miller, A. Rao, K. Rose, and A. Gersho, "A global optimization technique for statistical classifier design," *IEEE Trans. Signal Processing*, vol. 44, pp. 3108–3122, Dec. 1996.

[26] D. Miller, K. Rose, and P. A. Chou, "Deterministic annealing for trellis quantizer and HMM design using Baum–Welch re-estimation," in *Proc. Int. Conf. Acoustic, Speech, Signal Processing*, vol. 5, 1994, pp. 261–264.

[27] J. W. Picone, "Signal modeling techniques in speech recognition," *Proc. IEEE*, vol. 81, pp. 1215–1247, Sept. 1993.

[28] W. H. Press, B. P. Flannery, S. A. Teulosky, and W. T. Vetterling, *Numerical Recipes In C: The Art of Scientific Computing*. Cambridge, MA: Cambridge Univ. Press, 1988.

[29] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, pp. 257–285, Feb. 1989.

[30] L. R. Rabiner and B.-H. Juang, "An introduction to hidden Markov models," *IEEE ASSP Mag.*, pp. 4–16, Jan. 1986.

[31] ——, *Fundamentals of Speech Recognition*. Englewood Cliffs, NJ: Prentice-Hall, 1993.

[32] L. R. Rabiner, S. Levinson, A. Rosenberg, and J. Wilpon, "Speaker independent recognition of isolated words using clustering techniques," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-27, pp. 336–349, Aug. 1979.

[33] L. R. Rabiner and J. Wilpon, "Some performance benchmarks for isolated word speech recognition systems," *Comput. Speech Lang.*, vol. 2, pp. 343–357, 1987.

[34] A. Rao, D. Miller, K. Rose, and A. Gersho, "A deterministic annealing approach for parsimonious design of piecewise regression models," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 21, pp. 159–173, Feb. 1999.

[35] ——, "A generalized VQ method for combined compression and estimation," in *Proc. Int. Conf. Acoustic, Speech, Signal Processing*, 1996, pp. 2032–2035.

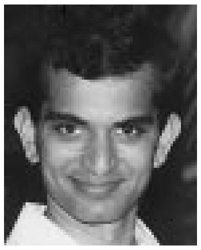
[36] ——, "Mixture of experts regression modeling by deterministic annealing," *IEEE Trans. Signal Processing*, vol. 45, pp. 2811–2820, Nov. 1997.

[37] K. Rose, "Deterministic annealing for clustering, compression, classification, regression, and related optimization problems," *Proc. IEEE*, vol. 86, pp. 2210–2239, Nov. 1998.

[38] K. Rose, E. Gurewitz, and G. C. Fox, "Statistical mechanics and phase transitions in clustering," *Phys. Rev. Lett.*, vol. 65, pp. 945–948, 1990.

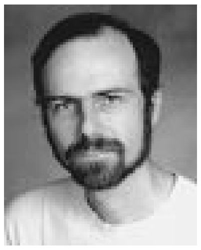
[39] ——, "Vector quantization by deterministic annealing," *IEEE Trans. Inform. Theory*, vol. 38, pp. 1249–1258, July 1992.

[40] ——, "Constrained clustering as an optimization method," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 15, pp. 785–794, Aug. 1993.



Ajit V. Rao received the B.Tech degree in electronics and communication engineering in 1992 from the Indian Institute of Technology, Madras and the M.S. and Ph.D. degrees in 1993 and 1998, respectively, from the University of California, Santa Barbara.

He was with Texas Instruments, Dallas, TX (Summer 1995) and SignalCom, Santa Barbara (1998–2000). He is currently with Microsoft Corporation, Santa Barbara. His research interests include compression of speech, audio and video for packet and wireless networks, speech recognition, and statistical pattern recognition.



Kenneth Rose (S'85–M'91) received the B.Sc. (summa cum laude) and M.Sc. (magna cum laude) degrees in electrical engineering from Tel Aviv University, Tel Aviv, Israel, in 1983 and 1987, respectively, and the Ph.D. degree from the California Institute of Technology, Pasadena, in 1990.

From July 1983 to July 1998, he was with Tadiran Ltd., Israel, where he carried out research in the areas of image coding, transmission through noisy channels, and general image processing. In January 1991, he joined the Department of Electrical and Computer Engineering, University of California, Santa Barbara, where he is currently an Associate Professor. His research interests are in information theory, source and channel coding, speech and general pattern recognition, image coding and processing, and nonconvex optimization in general.

Dr. Rose currently serves as the Editor for Source-Channel Coding for the IEEE TRANSACTIONS ON COMMUNICATION. He was co-recipient of the William R. Bennett Prize Paper Award of the IEEE Communications Society (1990).