Robust Combination of Neural Networks and Hidden Markov Models for Speech Recognition

Edmondo Trentin and Marco Gori, *Fellow, IEEE*

***Abstract*—Acoustic modeling in state-of-the-art speech recognition systems usually relies on hidden Markov models (HMMs) with Gaussian emission densities. HMMs suffer from intrinsic limitations, mainly due to their arbitrary parametric assumption. Artificial neural networks (ANNs) appear to be a promising alternative in this respect, but they historically failed as a general solution to the acoustic modeling problem. This paper introduces algorithms based on a gradient-ascent technique for global training of a hybrid ANN/HMM system, in which the ANN is trained for estimating the emission probabilities of the states of the HMM. The approach is related to the major hybrid systems proposed by Bourlard and Morgan and by Bengio, with the aim of combining their benefits within a unified framework and to overcome their limitations. Several viable solutions to the "divergence problem"—that may arise when training is accomplished over the maximum-likelihood (ML) criterion—are proposed. Experimental results in speaker-independent, continuous speech recognition over Italian digit-strings validate the novel hybrid framework, allowing for improved recognition performance over HMMs with mixtures of Gaussian components, as well as over Bourlard and Morgan's paradigm. In particular, it is shown that the maximum *a posteriori* (MAP) version of the algorithm yields a 46.34% relative word error rate reduction with respect to standard HMMs.**

***Index Terms*—Global optimization, hidden Markov model (HMM), hybrid system, maximum a posteriori criterion (MAP), maximum-likelihood (ML) training, speech recognition.**

I. Introduction

AUTOMATIC speech recognition (ASR) [1], [2] can be defined as the classification of the correct word(s) $\mathcal{W}$ from a certain dictionary, according to the word(s) posterior probability

$$\Pr(\mathcal{W}|Y) = \frac{\Pr(Y|\mathcal{W})\Pr(\mathcal{W})}{\Pr(Y)} \tag{1}$$

given the acoustic feature vectors Y. The quantities $\Pr(\mathcal{W})$ and $\Pr(Y|\mathcal{W})$ are usually referred to as the language model and the *acoustic model*, respectively. Hidden Markov models (HMMs) [1], [3] are the most popular approach to the problem of acoustic modeling in ASR.

Although HMMs yield good (yet not perfect) recognition performance under many experimental conditions, they suffer from major limitations [4], [5]. In particular, HMMs rely on strong assumptions on the statistical properties of the phenomena under consideration. The assumption of a given parametric form for the probability density functions (pdfs) that represent the *emission* probabilities associated with states in continuous-density HMMs (CDHMMs)[1] [6] is arbitrary and constraining. In addition, possible correlations among individual acoustic features are not taken into account, since concrete implementations of CDHMMs do usually assume statistical independence among input features, i.e. diagonal covariance matrices. Given these limitations, the use of artificial neural networks (ANNs) appears promising: ANNs can be trained as nonparametric probabilistic models and they may show interesting generalization capabilities [7]. The latter can be further improved on the basis of ad hoc regularization techniques.

In spite of their ability to classify short-time acoustic-phonetic units, such as individual phonemes or few isolated words [8]–[12], ANNs historically failed as a general framework for ASR [5]. Failure emerges when dealing with long sequences of acoustic observations, like those required in order to represent words, or sentences, from a large dictionary. This is due to the lack of ability to model long-term time dependencies in ANNs, even when recurrent architectures are considered [13]. In the early 1990s, this fact led to the idea of combining HMMs and ANNs within unifying novel models, broadly known as *hybrid* ANN/HMM. A number of significant, different hybrid paradigms for ASR were proposed in the literature, e.g., [14]–[22]. Surveys on the topic can be found in [5], [23]. The approach presented in this paper is related to some extent to Bourlard and Morgan's hybrid [4], [24], [25], as well as to Bengio's optimization scheme [26].

Bourlard and Morgan's approach relies basically on the maximization of the *posterior probability* of a Markov model $\mathcal{M}$ given an acoustic observation sequence Y. For each state q_ℓ in $\mathcal{M}$, an output unit of a multilayer perceptron (MLP) is trained to estimate the corresponding state-posterior probability, called *conditional transition probability*, $\Pr(q_\ell|\mathbf{y}_{\ell-k},\ldots,\mathbf{y}_{\ell+k},q_{\ell-1})$, given a fixed number $2k+1$ of acoustic vectors $\mathbf{y}_{\ell-k},\ldots,\mathbf{y}_{\ell+k}$ (a *window* or *context* of size k centered in the current acoustic observation $\mathbf{y}_\ell$) and the previous state $q_{\ell-1}$. This is accomplished using the *backpropagation* (BP) algorithm [27]. The speech recognition performance of the system was originally unsatisfactory ([4, p. 117]). This was attributed to a mismatch between the actual state-prior probabilities (estimated from relative frequencies over the training data) and priors implicitly induced by the structure of the model. A step backward (from state-posteriors

Manuscript received February 21, 2001; revised August 26, 2002 and March 19, 2003.

E. Trentin is with the Dipartimento di Ingegneria dell'Informazione, Università di Siena, 56100-Siena, Italy. He is also with ITC-irst-Pantè di Povo, Trento, Italy (e-mail: trentin@dii.unisi.it).

M. Gori is with the Dipartimento di Ingegneria dell'Informazione, Università di Siena, 56100-Siena, Italy (e-mail: marco@dii.unisi.it).

Digital Object Identifier 10.1109/TNN.2003.820838

[1]Gaussian pdfs or mixtures of Gaussian components are commonly used to model the emission probabilities in CDHMMs.

to likelihoods) was then made via *normalization* of ANN outputs, dividing them by the *prior* probabilities of the corresponding states. In so doing, probabilities were reduced to scaled likelihoods, normalized by the unconditional likelihood of each observation using Bayes' theorem. This solution allowed the system to reach, and even improve, the recognition performance of HMMs in several continuous speech tasks.

The training procedure is crucial. Indeed, BP requires knowledge of target output values to compute the gradient of the cost function. On the contrary, no supervised labeling of acoustic frames is actually available in the general case. Bourlard and Morgan propose an iterative training scheme: 1) start up with an initial segmentation of the observation sequences; 2) perform BP training of the ANN according to the current segmentation; 3) using the newly trained ANN to estimate the state-posterior probabilities, apply the Viterbi algorithm [3] to produce a new and more reliable segmentation of the data; 4) use such a segmentation to train the network again, and so on, in an iterative fashion. One problem with this approach is that the optimization of the system parameters is not accomplished according to a globally-defined criterion, but relies on the *heuristic* BP/Viterbi iterative scheme. Also the step-back from posterior probabilities to likelihoods through Bayes' theorem represents a divergence from the theoretical framework, to some extent. The nature of the initial segmentation is quite critical, too. Starting from a bad segmentation may even prevent the algorithm from convergence.

In Bengio's hybrid model the ANN learns to map a high-dimensional input sequence $\mathbf{u}_1^L$ onto a low-dimensional observation sequence $\mathbf{x}_1^L$ [26], [28], [29]. Let x_t denote a generic component of t-th observation $\mathbf{x}_t$, and assume that the HMM is trained to maximize a criterion function C. If C is a continuous and differentiable function of the observations $\mathbf{x}_t, t = 1, \ldots, L$, then gradient-ascent can be applied to learn the weights w of the ANN, relying on the computation of the partial derivatives $\partial C / \partial x_t$ and on their backpropagation through the ANN. In CDHMMs, C is the likelihood of the observations given the model, i.e. the *maximum likelihood* (ML) criterion. In this case, the quantity $\partial C / \partial x_t$ is computed applying the chain rule and observing that the *Forward-Backward* (or *Baum-Welch*) algorithm [1] implicitly computes the derivative of the likelihood C with respect to the emission probabilities of the HMM. This property is reviewed in Section II, where it is part of the calculations required to develop novel training algorithms. Unfortunately, using the ANN as a feature extractor for a CDHMM does not allow for any solutions to the intrinsic problems of HMMs and increases the overall number of parameters to be learned from the data.

This paper proposes ANN/HMM architectures and training techniques aimed at combining the advantages of Bourlard and Morgan's architecture and Bengio's training scheme, tackling their intrinsic drawbacks. The proposed hybrid system relies on a connectionist nonparametric estimate of the emission probabilities of an HMM, with gradient-ascent global training techniques. The ANN has an output unit for each state of each HMM in the recognition system.[2] Each output unit provides an estimate of the corresponding emission probability given the current acoustic observation in the feature space. Training of the other probabilistic quantities in the underlying HMM, i.e. *initial* and *transition* probabilities, still relies on the Baum-Welch algorithm. The Viterbi algorithm is eventually applied to the recognition step.

A generic ML training algorithm is given in Section II, and the corresponding implementation issues are covered in Appendix II. The calculations and the ML learning rule obtained underly all the following variants of the algorithm itself. A critical point has then to be considered, referred to as the "divergence problem" (Section III): how can we ensure that the ANN outputs do actually estimate likelihoods? In a broader sense, the requirement that the ANN be a *pdf* can be relaxed, requiring only that its outputs are bound, i.e., no divergence is allowed to occur during training. Major answers to the question are given in the paper by: i) imposing architectural constraints on the ANN topology, Section III-A; ii) imposing a probabilistic constraint on ANN weights, Section III-B; iii) factorizing the emission probabilities via Bayes' theorem, i.e. performing connectionist *state-posterior* probability estimation, Section III-C; iv) extremizing a *maximum a posteriori* (MAP) training criterion, instead of the bare ML, Section III-D: this implies maximization of the likelihood of a "constrained" model, and the simultaneous minimization of the likelihood of a "unconstrained" model, relying on the bare ML algorithm. Solution iv) turns out to be particularly effective. Experimental results are reported in Section IV, where a real-world, speaker independent continuous speech recognition task, namely Italian connected digit-strings recognition, is then approached.

II. Derivation of a Gradient-Based Algorithm to Train the ANN/HMM Hybrid

Let us introduce the global criterion function C to be maximized by the model during training, namely the *likelihood* L of the acoustic observations given the model, that is $C = L$ where[3] [1], [29]

$$L = \sum_{i \in \mathcal{F}} \alpha_{i,T}. \tag{2}$$

The sum is extended to the set $\mathcal{F}$ of all possible *final* states [29] within the HMM corresponding to the current phonetic transcription. The transcription in terms of HMMs is supposed to involve Q states, and T is the length of the current observation sequence $Y = \mathbf{y}_1, \ldots, \mathbf{y}_T$. The *forward* terms $\alpha_{i,t} = \Pr(q_{i,t}, \mathbf{y}_1, \ldots, \mathbf{y}_t)$ and the *backward* terms $\beta_{i,t} = \Pr(\mathbf{y}_{t+1}, \ldots, \mathbf{y}_T | q_{i,t})$ for state i at time t can be computed recursively as follows [1]:

$$\alpha_{i,t} = b_{i,t} \sum_j a_{ji} \alpha_{j,t-1} \tag{3}$$

and

$$\beta_{i,t} = \sum_j b_{j,t+1} a_{ij} \beta_{j,t+1} \tag{4}$$

[2]In the following, the term HMM will be used to denote the set of all left-to-right HMMs in the recognition system.

[3]A standard notation is used in the following to refer to quantities involved in HMM training. See, for instance, [30].

where a_{ij} denotes the transition probability from ith state to jth state, $b_{i,t}$ denotes emission probability associated with i-th state over tth observation $\mathbf{y}_t$, and the sums are extended to all possible states within the HMM. The initialization of the forward probabilities is accomplished as in CDHMMs [1], whereas the backward terms at time T are initialized in a slightly different manner, namely

$$\beta_{i,T} = \begin{cases} 1, & \text{if } i \in \mathcal{F} \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Assuming that an ANN may represent the emission *pdfs* in a proper manner,[4] and given a generic weight w of the ANN, hill-climbing gradient-ascent over C prescribes a learning rule of the kind:

$$\Delta w = \eta \frac{\partial L}{\partial w} \tag{6}$$

where η is the *learning rate*. In the following, we derive a learning rule for the ANN aimed at maximizing the likelihood of the acoustic observations, which is the usual criterion adopted to train HMMs. Eventually, the learning rule can be applied in parallel with the Baum–Welch algorithm, limiting the latter to the ML estimation of those quantities (namely the initial and transition probabilities in the underlying HMM) that do not explicitly depend on the ANN weights. Section III points out that a direct application of the present learning rule may lead to the "divergence problem," requiring further elaboration on the basic algorithm. Nonetheless, the present calculations, and the following learning rule, form the basis for all the variants of the algorithms introduced in later sections of the paper. For these reasons, we believe it is crucial to present the basic ML learning rule at this point.

Let us observe, after [29], that the following property can be easily shown to hold true by straightforwardly taking the partial derivatives of the left- and right-hand sides of (3) with respect to $b_{i,t}$

$$\frac{\partial \alpha_{i,t}}{\partial b_{i,t}} = \frac{\alpha_{i,t}}{b_{i,t}}. \tag{7}$$

In addition, borrowing the scheme proposed by [31] and [29], the following theorem can be proved to hold true (see Appendix I): $\partial L/\partial \alpha_{i,t} = \beta_{i,t}$, for each $i = 1,\ldots,Q$ and for each $t = 1,\ldots,T$. Given this theorem and (7), repeatedly applying the chain rule we can expand $\partial L/\partial w$ by writing

$$\begin{aligned} \frac{\partial L}{\partial w} &= \sum_i \sum_t \frac{\partial L}{\partial b_{i,t}} \frac{\partial b_{i,t}}{\partial w} \\ &= \sum_i \sum_t \frac{\partial L}{\partial \alpha_{i,t}} \frac{\partial \alpha_{i,t}}{\partial b_{i,t}} \frac{\partial b_{i,t}}{\partial w} \\ &= \sum_i \sum_t \beta_{i,t} \frac{\alpha_{i,t}}{b_{i,t}} \frac{\partial b_{i,t}}{\partial w} \end{aligned} \tag{8}$$

where the sums are extended over all states $i = 1,\ldots,Q$ of the HMM corresponding to the correct transcription of the training utterance, and to all $t = 1,\ldots,T$, respectively. From now on, attention is focused on the calculation of $\partial b_{i,t}/\partial w$, where $b_{i,t}$ is the output from ith unit of the ANN at time t.

Let us consider a multilayer feedforward network (e.g., an MLP), the jth output of which, computed over tth input observation $\mathbf{y}_t$, is expected to be a nonparametric estimate of the emission probability $b_{j,t}$ associated with jth state of the HMM at time t. An activation function $f_j(x_j(t))$, either linear or nonlinear, is attached to each unit j of the ANN, where $x_j(t)$ denotes input to the unit itself at time t. The corresponding output $o_j(t)$ is given by $o_j(t) = f_j(x_j(t))$. The net is assumed to have n layers $\mathcal{L}_0, \mathcal{L}_1, \ldots, \mathcal{L}_n$, where $\mathcal{L}_0$ is the input layer and is not counted, and $\mathcal{L}_n$ is the output layer. For notational convenience we write $i \in \mathcal{L}_k$ to denote the index of ith unit in layer $\mathcal{L}_k$.

When considering the case of a generic weight w_{jk} between kth unit in layer $\mathcal{L}_{n-1}$ and jth unit in the output layer, we can write

$$\begin{aligned} \frac{\partial b_{j,t}}{\partial w_{jk}} &= \frac{\partial f_j(x_j(t))}{\partial w_{jk}} \\ &= f_j'(x_j(t)) \frac{\partial \sum_{i \in \mathcal{L}_{n-1}} w_{ji} o_i(t)}{\partial w_{jk}} \\ &= f_j'(x_j(t))\, o_k(t). \end{aligned} \tag{9}$$

By defining the quantity[5]

$$\delta_i(j,t) = \begin{cases} f_j'(x_j(t)), & \text{if } i = j \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

for each $i \in \mathcal{L}_n$, we can rewrite (9) in the compact form

$$\frac{\partial b_{j,t}}{\partial w_{jk}} = \delta_j(j,t) o_k(t). \tag{11}$$

Consider now the case of connection weights in the first hidden layer. Let w_{kl} be a generic weight between lth unit in layer $\mathcal{L}_{n-2}$ and kth unit in layer $\mathcal{L}_{n-1}$.

$$\begin{aligned} \frac{\partial b_{j,t}}{\partial w_{kl}} &= \frac{\partial f_j(x_j(t))}{\partial w_{kl}} \\ &= f_j'(x_j(t)) \sum_{i \in \mathcal{L}_{n-1}} \frac{\partial w_{ji} o_i(t)}{\partial w_{kl}} \\ &= f_j'(x_j(t)) \frac{\partial w_{jk} o_k(t)}{\partial o_k(t)} \frac{\partial o_k(t)}{\partial w_{kl}} \\ &= f_j'(x_j(t))\, w_{jk} \frac{\partial f_k(x_k(t))}{\partial w_{kl}} \\ &= f_j'(x_j(t))\, w_{jk} f_k'(x_k(t)) \frac{\partial \sum_{i \in \mathcal{L}_{n-2}} w_{ki} o_i(t)}{\partial w_{kl}} \\ &= f_j'(x_j(t))\, w_{jk} f_k'(x_k(t))\, o_l(t). \end{aligned} \tag{12}$$

Similarly to definition (10), we introduce the quantity

$$\delta_k(j,t) = f_k'(x_k(t)) \sum_{i \in \mathcal{L}_n} w_{ik} \delta_i(j,t) \tag{13}$$

[4]This topic is investigated in Section III.

[5]Dependency on the specific HMM state j under consideration and on current time t is written explicitly, since this will turn out to be useful in the final formulation of the learning rule.

for each $k \in \mathcal{L}_{n-1}$, which allows us to rewrite (12) in the form

$$\frac{\partial b_{j,t}}{\partial w_{kl}} = \delta_k(j,t) o_l(t) \tag{14}$$

which is formally identical to (11).

Finally, we carry out the calculations for a generic, hypothetic weight w_{lm} between mth unit in layer $\mathcal{L}_{n-3}$ and lth unit in layer $\mathcal{L}_{n-2}$, but the same results will also apply to subsequent layers $(\mathcal{L}_{n-4}, \mathcal{L}_{n-5}, \ldots, \mathcal{L}_0)$

$$\begin{aligned}\frac{\partial b_{j,t}}{\partial w_{lm}} &= \frac{\partial f_j(x_j(t))}{\partial w_{lm}} \\ &= f_j'(x_j(t)) \sum_{i \in \mathcal{L}_{n-1}} \frac{\partial w_{ji} o_i(t)}{\partial w_{lm}} \\ &= f_j'(x_j(t)) \sum_{i \in \mathcal{L}_{n-1}} \frac{\partial w_{ji} o_i(t)}{\partial o_i(t)} \frac{\partial o_i(t)}{\partial w_{lm}} \\ &= f_j'(x_j(t)) \sum_{i \in \mathcal{L}_{n-1}} w_{ji} \frac{\partial o_i(t)}{\partial w_{lm}}. \end{aligned} \tag{15}$$

Efforts are now concentrated on the development of the term $\partial o_i(t)/\partial w_{lm}$. This is accomplished by writing

$$\begin{aligned}\frac{\partial o_i(t)}{\partial w_{lm}} &= \frac{\partial f_i(x_i(t))}{\partial w_{lm}} \\ &= f_i'(x_i(t)) \sum_{k \in \mathcal{L}_{n-2}} \frac{\partial w_{ik} o_k(t)}{\partial w_{lm}} \\ &= f_i'(x_i(t)) \frac{\partial w_{il} o_l(t)}{\partial o_l(t)} \frac{\partial o_l(t)}{\partial w_{lm}} \\ &= f_i'(x_i(t)) w_{il} \frac{\partial f_l(x_l(t))}{\partial w_{lm}} \\ &= f_i'(x_i(t)) w_{il} f_l'(x_l(t)) \frac{\partial x_l(t)}{\partial w_{lm}} \\ &= f_i'(x_i(t)) w_{il} f_l'(x_l(t)) \sum_{k \in \mathcal{L}_{n-3}} \frac{\partial w_{lk} o_k(t)}{\partial w_{lm}} \\ &= f_i'(x_i(t)) w_{il} f_l'(x_l(t)) o_m(t) \end{aligned} \tag{16}$$

which can be substituted into (15) obtaining

$$\frac{\partial b_{j,t}}{\partial w_{lm}} = f_j'(x_j(t)) \sum_{i \in \mathcal{L}_{n-1}} w_{ji} f_i'(x_i(t)) w_{il} f_l'(x_l(t)) o_m(t). \tag{17}$$

Again, as in (10) and (13), for each $l \in \mathcal{L}_{n-2}$ we define the quantity:[6]

$$\delta_l(j,t) = f_l'(x_l(t)) \sum_{i \in \mathcal{L}_{n-1}} w_{il} \delta_i(j,t) \tag{18}$$

and rewrite (17) in the familiar, compact form

$$\frac{\partial b_{j,t}}{\partial w_{lm}} = \delta_l(j,t) o_m(t) \tag{19}$$

that is formally identical to (11) and (14).

[6]For lower layers the sum in (18) must be accomplished over the units belonging to the immediately upper layer.

Using the above developments to expand (8) and substituting it into (6), the latter can now be restated in the form of a general *learning rule* for a generic weight w_{jk} of the network, by writing

$$\Delta w_{jk} = \eta \sum_{i=1}^{Q} \sum_{t=1}^{T} \beta_{i,t} \frac{\alpha_{i,t}}{b_{i,t}} \delta_j(i,t) o_k(t) \tag{20}$$

where the term $\partial_j(i,t)$ is computed via (10), (13) or (18), according to the fact that the layer under consideration is the output layer, the first hidden layer, or one of the other hidden layers, respectively. In the following, we will refer to learning rule (20) as the *bare maximum likelihood* (BML) training algorithm for the ANN/HMM hybrid. It is convenient to rewrite it as

$$\Delta w_{jk} = \eta \sum_{t=1}^{T} o_k(t) \sum_{i=1}^{Q} \beta_{i,t} \frac{\alpha_{i,t}}{b_{i,t}} \delta_j(i,t). \tag{21}$$

This reduces the number of calculations by a factor Q. Once trained, the ANN can be used along with the Viterbi algorithm to perform decoding of novel sequences of acoustic observations.

Appendix B discusses major implementation issues. In fact, the BML algorithm deals with a single training sequence Y, while actual training sets may be composed of thousands of training utterances collected among a number of speakers. Appendix II-A introduces a *batch* learning rule in the linear domain,[7] aimed at the maximization of the overall likelihood of a whole set of training sequences given the model. For this reason, in the following we will refer to (21) as the *on-line* learning rule for the ANN/HMM hybrid in the linear domain. Furthermore, the analytical developments presented in this section are focused on the likelihood L and they rely on quantities which are defined in the linear domain, while numerical stability problems in HMMs may suggest to perform calculations in the logarithmic domain, aiming at the maximization of the *log-likelihood*. *On-line* and *batch* learning rules in the logarithmic domain are presented in Appendix II-B.

III. The "Divergence Problem"

The learning rules developed so far are expected to be effective under the assumption that the ANN actually realizes a (generic and nonparametric) model of a *pdf*. Whereas ANNs *may* reasonably approximate any continuous *pdf* as close as desired, in general they completely lack of any constraints ensuring that, at any given time during training, the function they compute actually *is* a *pdf*. In practice, a direct application of the above training scheme may not lead to a correct solution, since the training criterion (ML) simply encourages the ANN to yield high output values (inflating the values of the connection weights), i.e., high emission probabilities, over all input patterns. In so doing, the overall (pseudo)likelihood of the training input sequences may diverge toward infinity. Eventually, the resulting model turns out to be meaningless (and useless). We refer to such a phenomenon as the "divergence problem." A strictly related scenario was pointed out by Bengio [29], where the ANNs for feature extraction might simply learn to

[7]The term "linear" refers to the fact that ANN outputs are expected to be the values of pdfs themselves. Equations in the logarithmic domain are introduced later.

concentrate all the inputs closely around the mean vectors of the Gaussian emission probabilities, the latter quickly becoming Dirac's deltas. While using Gaussian mixtures provided Bengio with the satisfaction of the "integral-equal-to-one" condition, in the present setup the problem is even harder.

In practice, according to [29], the ANN may be expected to reach "more interesting" extremes of the criterion function. Although some experiments do confirm the expectation to a limited extent [32], the problem turns out to be relevant and further theoretical developments are sought. Explicit introduction of the probabilistic constraint "integral-equal-to-one" in the training criterion does not lead to any feasible analytical formulation. Different alternatives are presented in Sections III-A through III-D below. We stress the fact that, in practice, the ANN is not necessarily required to realize a pdf, but *divergence* of its output values should rather be avoided. Some of the solutions that are outlined in the following are thus aimed at the definition of architectures/training algorithms that prevent the ANN to develop unbound outputs while increasing the likelihood of the acoustic observations.

A. Imposing Architectural Constraints on the ANN Topology

Radial basis functions (RBF) networks [33], [34] can be interpreted, under certain assumptions, as mixtures of Gaussian components, i.e., as *pdfs*. For this reason, RBFs may look promising in the present framework. For instance, the RBF may be initialized with the same parameters of a pretrained HMM, and further trained using gradient-ascent. In fact, there are two major drawbacks that severely limit the applicability of RBFs herein: i) whenever they realize bare mixtures of Gaussian, no significant gain in modeling capabilities may be expected over the standard CDHMM; ii) if trained via the gradient method, hidden-to-output weights may diverge in order to increase the overall likelihood (as discussed earlier) since no explicit constraint ensures that, for each output unit, they sum to one.

An alternative is represented by a modified MLP having locally tuned Gaussian output units. Presence of the latter ones might ensure the satisfaction of the pdf constraints. Such an architecture may appear unpromising, due to the fact that it basically replaces mixtures of Gaussians (as in CDHMMs) with individual Gaussians (one for each output unit, that is one for each state of the underlying HMM). Nonetheless, it should be noticed that the MLP performs a nonlinear transformation of the input features that is aimed at maximizing the likelihood. This means that the output Gaussians are defined over a transformed space, which is expected to better suit the ML framework for the recognition problem at hand. This architecture, along with the BML algorithm, provides us with a scheme that strictly resembles Bengio's paradigm, i.e., the ANN extracts "optimal" features to be modeled with a Gaussian-based HMM. The "divergence problem" may thus rise again in the same form as in Bengio's case (as outlined earlier).

B. Imposing a Probabilistic Constraint on the ANN Weights

The present approach is inspired from several regularization techniques, in particular *weight decay* [35] and *soft weight sharing* [36]. The basic idea is to avoid divergence of weight values by encouraging simpler solutions, having small weights, rather than complex solutions, i.e., with large weights. The ANN training scheme must be modified in order to extremize the criterion function while keeping the weights "as small as possible".

In the present setup, the idea can be formalized imposing a probabilistic constraint on the ANN weights, by replacing the criterion C (the bare likelihood of the acoustic observations given the model) with a new criterion C', namely a *joint* probability having the form

$$C' = P(Y|\mathcal{M})P(W|\mathcal{M}) \tag{22}$$

where $L = P(Y|\mathcal{M})$ is the likelihood of the observation sequence Y given the model $\mathcal{M}$, and $W = (w_1, \dots, w_s)$ is the random vector of all connection weights in the ANN. Weights are assumed to be random variables distributed according to certain *pdfs*. The goal is now the maximization of criterion C'.

Exactly like in (6), we can use gradient ascent to maximize C' by taking

$$\Delta w = \eta \frac{\partial C'}{\partial w} \tag{23}$$

for a generic weight w. From (22)

$$\begin{aligned}\frac{\partial C'}{\partial w} &= \frac{\partial P(Y|\mathcal{M})P(W|\mathcal{M})}{\partial w}\\ &= P(W|\mathcal{M})\frac{\partial P(Y|\mathcal{M})}{\partial w} + P(Y|\mathcal{M})\frac{\partial P(W|\mathcal{M})}{\partial w}.\end{aligned} \tag{24}$$

The choice of the form for $P(W|\mathcal{M})$ is crucial. For instance, if a uniform distribution is chosen, then no constraints on the weights are imposed, (24) reduces to (8), and the criteria C and C' coincide.[8] In order to encourage small weights, a distribution with a peak around zero, and penalizing large arguments, is sought. A Normal distribution centered in zero appears promising in this respect: the zero-mean Gaussian is the pdf underlying classical regularization techniques for ANNs training. Under this hypothesis, assuming that the weights are independent random variables, we can write

$$P(W|\mathcal{M}) = \prod_{j=1}^{s} P(w_j|\mathcal{M}) = \prod_{j=1}^{s} N\left(w_j; 0, \sigma_j^2\right) \tag{25}$$

where $N(w_j; 0, \sigma_j^2)$ denotes the jth (univariate) Gaussian, having zero mean and variance σ_j^2, evaluated over the current value of weight w_j. Partial derivatives with respect to a generic weight w can be explicitly calculated

$$\frac{\partial P(W|\mathcal{M})}{\partial w} = \frac{\prod_{j=1}^{s} P(w_j|\mathcal{M})}{P(w|\mathcal{M})}\frac{\partial P(w|\mathcal{M})}{\partial w} \tag{26}$$

and substituted into (24) to obtain a proper learning rule. It is possible to introduce the quantity[9] $\lambda = (1/\sigma^2)$, such that

$$P(w|\mathcal{M}) = \sqrt{(\lambda/2\pi)}\exp(-(\lambda/2)w^2)$$

[8]Except for the multiplicative constant $P(W|\mathcal{M})$, which is irrelevant in the criterion extremization perspective.

[9]This notation emphasizes the relationship with soft weight grouping techniques [36], [37].

and to write

$$\frac{\partial P(w|\mathcal{M})}{\partial w} = -\sqrt{\frac{\lambda}{2\pi}}\lambda w \exp\left(-\frac{\lambda}{2}w^2\right) = -\lambda w P(w|\mathcal{M}) \tag{27}$$

that can be substituted into (26) and the latter, in turn, into (24) and (23) to obtain the following *on-line learning rule*

$$\Delta w_{jk} = \eta P(W|\mathcal{M}) \times \left\{ \sum_{i=1}^{Q} \sum_{t=1}^{T} \beta_{i,t} \frac{\alpha_{i,t}}{b_{i,t}} \delta_j(i,t) o_k(t) - P(Y|\mathcal{M}) \lambda_{jk} w_{jk} \right\} \tag{28}$$

once the term $\partial P(Y|\mathcal{M})/(\partial w)$ has been calculated using (8) through (19) of Section II, which are not affected by the present setup. The quantities $P(W|\mathcal{M})$ and $P(Y|\mathcal{M})$ (the likelihood L) can be computed, at each iteration of the algorithm, in a straightforward manner from (25) and (2), respectively. The term λ_{jk} is the inverse of the variance of the Gaussian pdf associated with weight w_{jk}, and the other symbols have the same meaning as in (20). We will refer to learning rule (28) as the *Soft Weight Sharing ML* (SWS-ML) training algorithm for the ANN/HMM hybrid.

In passing, note that the multiplicative terms η and $P(W|\mathcal{M})$ in (28) may be conveniently absorbed within a unique *variable learning rate* η', defined as $\eta' = \eta P(W|\mathcal{M})$, that is implicitly *adapted* at each iteration of the learning process. Following the calculations outlined in Appendexes II-A and II–B, *batch* as well as *logarithmic* learning rules can still be obtained.

C. Factorization of Emission Probabilities Using Bayes' Theorem

Using Bayes' theorem, we can rewrite the emission probability for state q_j at time t over acoustic frame $\mathbf{y}_t$ as

$$b_{j,t} = \Pr(\mathbf{y}_t|q_j) = \frac{\Pr(q_j|\mathbf{y}_t)p(\mathbf{y}_t)}{\Pi_j} \tag{29}$$

where $p(\mathbf{y}_t)$ is the value of the pdf of the acoustic observations in the feature space, evaluated over input $\mathbf{y}_t$, and $\Pi_j = \Pr(q_j)$ is the prior probability of state q_j. For notational convenience, we write $\Pr(\cdot)$ to denote a probability, and $p(.)$ to denote a pdf. The distribution $p(\mathbf{y})$ and the quantities Π_j for $j = 1, \ldots, Q$ are independent of time, i.e., they refer to the whole universes of possible acoustic vectors and HMM states, regardless of the time they occur. The first part of (29), i.e., $b_{j,t} = \Pr(\mathbf{y}_t|q_j)$, holds true only under the assumption, implicit in the definition of HMM, that the emission pdf associated with a specific state is independent of time [1]. In the following, the meaning of Q is slightly different with respect to the previous sections, since we need to consider *all* the states in the HMM, not only those actually implied in the current phonetic transcription. This does not affect the formulas developed so far, that still hold formally true, since the forward and backward terms are null for nonimplied states.

In order to use (29) in the training algorithm, feasible ways to estimate the quantities involved in the factorization are sought. The estimation is accomplished as follows:

1) $p(\mathbf{y})$ models the distribution of all acoustic vectors within the acoustic feature space. Given the training data, and pooling all the acoustic frames $\mathbf{y}$ (regardless of the observation sequence they belong to, or the time in which they occur), estimating $p(\mathbf{y})$ can be considered a typical *density estimation* problem. This can be approached by relying on statistical parametric (e.g., ML, minimax, or Bayesian learning) or nonparametric techniques [38]. Estimation is carried out on the whole training set, before starting the training of the ANN/HMM hybrid. If the input data are normalized in a way that reduces them to a uniform distribution, then $p(\mathbf{y})$ can be dropped from (29).
2) The state-priors Π_j for $j = 1, \ldots, Q$ can be estimated over the whole training set, before starting the training, on the basis of high-level considerations. These refer to the task domain, the linguistic constraints (e.g., vocabulary and grammar) and the topology of the HMM. As an alternative, they can be estimated as follows. Consider the correct concatenations of HMMs associated with the phonetic transcriptions of the training utterances, i.e., the same concatenations used for standard Baum-Welch training. Estimates are then obtained from the relative frequencies of individual HMMs states in the corresponding Viterbi segmentation of such training sequences.
3) The estimation of the quantity $\Pr(q_j|\mathbf{y}_t)$ is critical and more difficult than the other terms in (29). We use an ANN to carry out the estimation. More precisely, a "normalized" version of the ANN outputs is considered by taking

$$\Pr(q_j|\mathbf{y}_t) = \frac{f_j(x_j(t))}{\sum_{i=1}^{Q} f_i(x_i(t))} \tag{30}$$

for $j = 1, \ldots, Q$, where Q is the total number of states of all the HMMs in the recognition system, and $f_i(x_i(t))$ denotes the ith output from the ANN when fed with input $\mathbf{y}_t$. The normalization of the ANN outputs is a suitable way to ensure that the estimates of state-posterior probabilities sum to one ($\sum_{i=1}^{Q} \Pr(q_j|\mathbf{y}_t) = 1$). The only other constraint that has to be be satisfied is that each output ranges within the [0,1] interval. Activation functions of output neurons must be chosen accordingly, e.g., sigmoids.

Note how the Bayes' factorization has moved the focus from the (unfeasible) problem of constraining an ANN to estimate a pdf, to the (affordable) requirement of ensuring a correct probabilistic interpretation of ANN outputs. It should be noticed that, although the ANN is now used to estimate state-posterior probabilities, the overall approach still differs from Bourlard and Morgan's paradigm. Indeed, the probabilistic quantities estimated by the ANN are different, as is the role they play in the ANN/HMM calculations. In addition, we keep on moving within a "joint" ANN/HMM training framework, aiming at the optimization of a global criterion. Finally, the present "normalization" of network outputs is mathematically motivated (and does not require any step back from the original theoretical framework).

From now on, efforts are focused on modifying the calculations of Section II according to the new role of the ANN. Equations (2)through (8) are not affected by the present discussion as they still hold. Starting from (9) the calculation is accomplished by taking the factorization (29) into account. The partial derivative of the emission probability $b_{j,t}$ with respect to a *generic* weight w is now given by

$$\begin{aligned}\frac{\partial b_{j,t}}{\partial w} &= \frac{\partial}{\partial w}\left\{\frac{\Pr(q_j|\mathbf{y}_t)p(\mathbf{y}_t)}{\Pi_j}\right\}\\ &= \frac{p(\mathbf{y}_t)}{\Pi_j}\frac{\partial \Pr(q_j|\mathbf{y}_t)}{\partial w}\\ &= \frac{p(\mathbf{y}_t)}{\Pi_j}\frac{\partial}{\partial w}\left\{\frac{f_j(x_j(t))}{\sum_{i=1}^{Q} f_i(x_i(t))}\right\}\\ &= \frac{p(\mathbf{y}_t)}{\Pi_j}\left[\sum_{i=1}^{Q} f_i(x_i(t))\right]^{-2}\\ &\quad\times\left\{\sum_{i=1}^{Q} f_i(x_i(t))\frac{\partial f_j(x_j(t))}{\partial w}\right.\\ &\quad\left. - f_j(x_j(t))\sum_{i=1}^{Q}\frac{\partial f_i(x_i(t))}{\partial w}\right\}. \end{aligned} \tag{31}$$

For a weight $w = w_{\iota k}$ in the topmost (output) layer, the same calculation carried out in (9) allows us to write

$$\frac{\partial f_\iota(x_\iota(t))}{\partial w_{\iota k}} = f'_\iota(x_\iota(t))\,o_k(t) \tag{32}$$

while, for $i = 1, \ldots, Q, i \neq \iota$, it is immediately seen that

$$\frac{\partial f_i(x_i(t))}{\partial w_{\iota k}} = 0. \tag{33}$$

Applying definition (10), (31) now takes the following form

$$\begin{aligned}\frac{\partial b_{j,t}}{\partial w_{\iota k}} &= \frac{p(\mathbf{y}_t)}{\Pi_j}\left[\sum_{i=1}^{Q} f_i(x_i(t))\right]^{-2}\\ &\quad\times\left\{\delta_\iota(j,t)o_k(t)\sum_{i=1}^{Q} f_i(x_i(t))\right.\\ &\quad\left. - f_j(x_j(t))\sum_{i=1}^{Q}\delta_\iota(i,t)o_k(t)\right\}. \end{aligned} \tag{34}$$

Let us now turn our attention to a generic weight $w = w_{kl}$ in the first hidden layer. Following calculations from (12) we can arrive at

$$\frac{\partial f_j(x_j(t))}{\partial w_{kl}} = f'_j(x_j(t))\,w_{jk}f'_k(x_k(t))\,o_l(t) \tag{35}$$

for $j = 1, \ldots, Q$. Definition (13) and (35) enable us to rewrite (31) as

$$\begin{aligned}\frac{\partial b_{j,t}}{\partial w_{kl}} &= \frac{p(\mathbf{y}_t)}{\Pi_j}\left[\sum_{i=1}^{Q} f_i(x_i(t))\right]^{-2}\\ &\quad\times\left\{\delta_k(j,t)o_l(t)\sum_{i=1}^{Q} f_i(x_i(t))\right.\\ &\quad\left. - f_j(x_j(t))\sum_{i=1}^{Q}\delta_k(i,t)o_l(t)\right\}. \end{aligned} \tag{36}$$

Finally, let us consider a connection weight in the following hidden layers. As in Section II, the calculations are accomplished for a generic weight $w = w_{lm}$ holding between m-th unit in layer $\mathcal{L}_{n-3}$ and lth unit in layer $\mathcal{L}_{n-2}$, but the same results also apply to subsequent layers ($\mathcal{L}_{n-4}, \mathcal{L}_{n-5}, \ldots, \mathcal{L}_0$).

The calculations in (15) and (16) can still be used to develop the quantity $\partial f_j(x_j(t))/(\partial w_{lm})$

$$\begin{aligned}&\frac{\partial f_j(x_j(t))}{\partial w_{lm}}\\ &\quad= f'_j(x_j(t))\sum_{i\in\mathcal{L}_{n-1}} w_{ji}f'_i(x_i(t))\,w_{il}f'_l(x_l(t))\,o_m(t) \end{aligned} \tag{37}$$

for $j = 1, \ldots, Q$. Note that for lower layers the summation must be accomplished over the units belonging to the layer immediately above. Again, (18) can be applied to (37), allowing us to rewrite (31) in the form

$$\begin{aligned}\frac{\partial b_{j,t}}{\partial w_{lm}} &= \frac{p(\mathbf{y}_t)}{\Pi_j}\left[\sum_{i=1}^{Q} f_i(x_i(t))\right]^{-2}\\ &\quad\times\left\{\delta_l(j,t)o_m(t)\sum_{i=1}^{Q} f_i(x_i(t))\right.\\ &\quad\left. - f_j(x_j(t))\sum_{i=1}^{Q}\delta_l(i,t)o_m(t)\right\}. \end{aligned} \tag{38}$$

Expanding (8) according to (34), (36), or (38), and then substituting it into (6) yields the new *learning rule* for a generic weight w_{jk} in the ANN

$$\begin{aligned}\Delta w_{jk} =& \eta\sum_{i=1}^{Q}\sum_{t=1}^{T}\beta_{i,t}\frac{\alpha_{i,t}}{b_{i,t}}\frac{p(\mathbf{y}_t)}{\Pi_i}\left[\sum_{m=1}^{Q} f_m(x_m(t))\right]^{-2}\\ &\times\left\{\delta_j(i,t)o_k(t)\sum_{m=1}^{Q} f_m(x_m(t))\right.\\ &\left. - f_i(x_i(t))\sum_{m=1}^{Q}\delta_j(m,t)o_k(t)\right\} \end{aligned} \tag{39}$$

where the notation for the indexes has been properly modified to avoid ambiguities. In the following, we will refer to the present training scheme as the *Bayes* training algorithm for the ANN/HMM hybrid. This approach is intrinsically discriminative, since weights for states which are not implied in the current transcription are modified (i.e., the output for the corresponding state is decreased) to increase the likelihood of implied states. *Batch* as well as *logarithmic* versions of the *Bayes* learning rule are discussed in Appendix III.

D. Maximum A Posteriori Discriminative Training Criterion

The adoption of a "discriminative" training criterion, namely the *maximum a posteriori* (MAP) criterion [29], instead of the bare ML, appears promising from two distinct viewpoints: (i) divergence of ANN outputs is avoided by forcing probabilities along "incorrect" paths of the HMM to be small in order to increase the posterior probability of the correct model given the acoustic observations; (ii) better overall performance is expected due to the discriminative training procedure [4]. The MAP criterion was used in a similar manner in [29] to tackle the problem of Gaussian pdfs that reduce to Dirac's Deltas in Bengio's model. Once the MAP criterion has been fixed, a new learning rule is obtained that involves the same calculations (and inherits properties) of the training schemes developed in the ML setup. The ANN may then be used to estimate the emission probabilities (with or without probabilistic constraints on its weights) or the state-posterior probabilities (as in the *Bayes* case), either in the linear or logarithmic domains.

We formally define the MAP criterion $\mathcal{C}_{\text{MAP}}$ as follows:

$$\mathcal{C}_{\text{MAP}} = P(\mathcal{M}|Y) = \frac{P(Y|\mathcal{M})P(\mathcal{M})}{P(Y)} \tag{40}$$

where Y is the observation sequence, and the interpretation of $\mathcal{M}$ (the model) is the same as in Section III-B. The term $P(\mathcal{M})$ is the *prior* probability of the model, and $P(Y)$ is the overall likelihood of the acoustic observation sequence Y, that does not depend on the specific choice for the model $\mathcal{M}$.

$P(\mathcal{M})$ is independent of the ANN outputs, since it depends on the language model only. It can thus be computed separately (if each word model has equal prior probability, $P(\mathcal{M})$ may even be dropped from (40)). In particular, in the on-line learning case, a suitable training algorithm is required to maximize the following equivalent criterion

$$\mathcal{C}'_{\text{MAP}} = \frac{P(Y|\mathcal{M})}{P(Y)}. \tag{41}$$

A gradient-based learning rule for a generic connection weight w can be obtained by taking the partial derivative of $\mathcal{C}'_{\text{MAP}}$ with respect to w as follows:

$$\begin{aligned}\Delta w &= \eta \frac{\partial \mathcal{C}'_{\text{MAP}}}{\partial w} \\ &= \eta \frac{\partial}{\partial w}\left\{\frac{P(Y|\mathcal{M})}{P(Y)}\right\} \\ &= \frac{\eta}{[P(Y)]^2}\left\{P(Y)\frac{\partial P(Y|\mathcal{M})}{\partial w} - P(Y|\mathcal{M})\frac{\partial P(Y)}{\partial w}\right\} \\ &= \frac{\eta}{P(Y)}\left\{\frac{\partial P(Y|\mathcal{M})}{\partial w} - \frac{P(Y|\mathcal{M})}{P(Y)}\frac{\partial P(Y)}{\partial w}\right\}. \end{aligned} \tag{42}$$

In practice, the quantity $P(Y|\mathcal{M})$ and its derivative with respect to w are computed according to the calculations introduced in Section II. Indeed, $P(Y|\mathcal{M})$ is the likelihood of the observations given the model $\mathcal{M}$, where $\mathcal{M}$ is the "correct" model. A model is said to be correct if it is *constrained* by the (known) transcription of the acoustic observation sequence Y [28], [39]. The quantity $P(Y)$ and its derivative with respect to w are computed by applying the same calculations to an *unconstrained*, or *recognition* model, as explained in [28], [39]. The latter is the same model used at *recognition* time with the Viterbi algorithm, where no prior knowledge on the exact transcription of the observations under consideration is given. It is representative of all allowable paths through all the HMMs states. We will refer to the present training scheme as the *MAP* learning algorithm for the ANN/HMM hybrid.

It is worth observing that, given (42), whenever the likelihood of Y given the correct model is low, the weight w is modified in order to extremize the MAP criterion by increasing $P(Y|\mathcal{M})$, almost regardless of the behavior of the ANN over the unconstrained model. When, on the contrary, $P(Y|\mathcal{M})$ is already high, the MAP criterion is mostly increased by lowering the likelihood of Y over the unconstrained model. In particular, divergence of ANN outputs is avoided. In all the other midrange cases, the learning rule for w imposes modifications that are a tradeoff between maximization of $P(Y|\mathcal{M})$ and minimization of the likelihood of Y over the unconstrained model, realizing a discriminative training scheme. Note also that, for implementation purposes, it may be convenient to define an alternative learning rate $\eta' = \eta/P(Y)$, i.e., the likelihood $P(Y)$ can be dropped and considered "absorbed" within the learning rate itself, given the arbitrariness of the latter.

Equation (42) defines an *on-line* learning rule based on the MAP criterion. A *batch* algorithm can be obtained, as well, considering again the setup introduced in Appendix II-A.

IV. Experimental Results

For the present experiments, *clean* speech signals from the `cdigits` part of the *SPK* database[10] were considered. The database was collected under laboratory conditions at the *Istituto per la Ricerca Scientifica e Tecnologica*, ITC-irst (Trento, Italy). A close-talk microphone was used for the recordings, under quiet laboratory conditions. The `cdigits` dataset is formed by continuous speech, namely 1000 utterances of connected Italian digit strings, acquired over 40 speakers (21 male and 19 female). The corpus was divided into two equally sized subsets, to be used for training and test, respectively. Each subset was thus composed of 500 utterances, each utterance consisting in 8 digits. Twenty speakers (10 male and 10 female) were inserted in the training set, while the others (11 male and 9 female) formed the test set.

Spectral analysis of the speech signals (acquired at a sampling rate of 16 kHz) was accomplished over 20 ms Hamming windows having an overlap of 10 ms, in order to extract 8 *mel frequency scaled cepstral coefficients* (MFSCCs) [40] as well as the signal log-energy as acoustic features. The log-energy of the signal was computed after application of the pre-emphasis filter. The nine features were normalized in order to have input values distributed in a uniform manner over the $[0, 1]$ interval, before feeding them into the connectionist models. Normalization was accomplished over the whole training set.

Results are reported in terms of recognition rate on the specific test database. The most popular evaluation criteria are the *word error rate* (WER) and its complementary, the *word recognition rate* (WRR). The former is defined as $\text{WER} = 100(\text{Ins} + \text{Del} + \text{Sub})/N_{\text{words}}\%$, where N_{words} is the total number of words in the uttered text, and the number of errors is expressed counting out word insertions (Ins), deletions (Del), and substitutions (Sub), respectively. The latter is defined as $\text{WRR} = 100\% - \text{WER}$. Since we are dealing with word *sequences*, further evaluation criteria may be introduced. The *percent correct* (PC) is defined as $\text{PC} = 100((N_{\text{words}} - (\text{Sub} + \text{Del}))/N_{\text{words}})\%$, which does not take into consideration insertion errors; indeed, Ins may be considerably reduced by imposing a suitable language model. Furthermore, the *string recognition rate* (SRR) is the fraction of test sequences that are recognized without any errors. Since an objective comparison between the *acoustic* models is sought,

[10]SPK is available from the European Language Resources Association (ELRA, http://www.icp.inpg.fr/ELRA/). It is divided into two parts: `idigits` is formed of isolated words, and `cdigits` features continuous speech.

TABLE I
PHONETIC TRANSCRIPTION OF THE DICTIONARY (IN TERMS OF SAMPA UNITS) AND CORRESPONDING NUMBER OF HMM STATES FOR THE ITALIAN CONNECTED DIGITS TASK

Word	*Phonetic transcription*	*Number of HMM states*
@bg (silence)		1
uno	u n o	3
due	d u e	3
tre	t r e	3
quattro	k w a tt r o	6
cinque	tS i n k w e	6
sei	s e i	3
sette	s e tt e	4
otto	o tt o	3
nove	n o v e	4
zero	ts e r o	4

the language model was tuned—from time to time—in order to have the same number of insertions and deletions for any given acoustic model.

Each word in the dictionary was modeled using one left-to-right HMM, without state-skips. The number of states in each HMM varied between 3 and 6, according to the phonetic transcription of Italian digits shown in Table I. A 1-state model for the "silence" (or "background noise," *@bg*) was used, for a total of 40 states. The phonetic transcription was accomplished using the SAMPA (Speech Assessment Methods Phonetic Alphabet) units, a machine-readable phonetic alphabet that was originally developed under the ESPRIT project *SAM* (Speech Assessment Methods).

Mixtures of eight Gaussian components with diagonal covariance matrices were used to model emission probabilities for each state of the CDHMM, without tying. The *segmental k-means* algorithm [30] was applied in order to initialize the HMM. Training was accomplished via the Baum–Welch algorithm, and the Viterbi decoding technique was applied over the test set.

The architecture of the MLP was defined as a two-layer feed-forward net, with 93 sigmoid units in the hidden layer and 40 sigmoids in the output layer. This topology was selected using cross-validation techniques during a preliminary experimental stage accomplished over speaker-dependent isolated digits from the idigits part of the SPK database [32]. Initialization of the MLP was accomplished according to the following algorithm:

1) an initial segmentation of the training set is obtained on the basis of prior knowledge on the correct HMM associated with each utterance and on the Viterbi hypothesis of segmentation provided by the pretrained HMM;
2) target values are assigned according to this segmentation, namely a positive target for unit i is assigned over all patterns belonging to state i given the segmentation, while a null target is assigned to all the other units (discriminative approach);
3) BP training of the MLP is accomplished over the labeled training set;
4) the trained MLP is used within the hybrid architecture along with the Viterbi algorithm to yield a novel segmentation of the training sequences;
5) the last two steps of the algorithm may be repeated for a certain number of iterations.

Note that this initialization technique basically coincides with Bourlard and Morgan's hybrid training scheme, once a probabilistic interpretation of ANN outputs is given in terms of "state posterior probabilities."

Preliminary experiments accomplished over a reduced-scale task from the idigits dataset suggested that training on-line in the linear domain may yield slightly better results with respect to batch training (or learning in the logarithmic domain) [32]. For this reason, training was accomplished via on-line BML, SWS-ML, *Bayes*, and MAP algorithms in the linear domain.

The state-priors Π_i to be used in the *Bayes* learning rule were estimated from the relative frequencies of the CDHMM states over the training set, during the initialization alignment procedure. No explicit estimation of the acoustic observations distribution $p(\mathbf{y})$ was accomplished to apply the *Bayes* algorithm herein, due to the uniform normalization of the input data.

Experimental results are reported in Table II. It is worth underlining that the absolute recognition rates on the given task (e.g., the "low" WRR yielded by the HMM) are conditioned by: (a) the limited amount of training data; (b) the strong acoustic mismatch between the vocal characteristics of speakers in the training and in the test sets. A comparison with Bourlard and Morgan's hybrid architecture is provided. The comparison was carried out by using the same ANN topology, and by starting from the same initial segmentation of training data labeled with 0/1 target values. The variant "Case 4" of the algorithm, described in [4] on page 164, was applied. Normalization of ANN outputs, i.e. division by the corresponding state-priors (estimated from the training set) in order to reduce to "likelihoods," was accomplished at recognition time, as recommended in [4, Ch. 7].

As far as the ANN/HMM hybrid is concerned, the BML algorithm (not shown in the table) did not provide us with any significant results. The BML increased the likelihood, but worsened the WRR due to the occurrence of the divergence problem. All the other variants of the algorithm (SWS-ML, *Bayes*, and MAP) improved over the HMM, as well as over Bourlard and Morgan's approach. In particular, it is seen that the ANN/HMM recognizer trained with the MAP version of the algorithm yields a 46.34% relative WER reduction over the standard HMM.

V. SUMMARY AND CONCLUSION

This paper introduced a novel ANN/HMM hybrid system where a feed-forward ANN carries out estimation of emission probabilities associated with the states of an underlying HMM

TABLE II
STRING RECOGNITION RATE (SRR), PERCENT CORRECT (PC) AND WORD RECOGNITION RATE (WRR) ON THE SPK TEST SET. THE HMM HAS 5869 FREE PARAMETERS AND ALL THE ANN/HMM HYBRIDS HAVE 4823 FREE PARAMETERS

Architecture/algorithm	*SRR (%)*	*PC (%)*	*WRR (%)*
HMM with 8-Gaussian mixtures	46.60	93.10	**90.03**
Bourlard and Morgan's hybrid	47.40	93.20	**90.20**
ANN/HMM hybrid trained via SWS-ML	64.80	95.67	**93.90**
ANN/HMM hybrid trained via *Bayes*	60.60	95.47	**93.25**
ANN/HMM hybrid trained via MAP	68.60	96.33	**94.65**

topology. Initial and transition probabilities are estimated via Baum-Welch, while gradient-ascent techniques aimed at the extremization of the ML or MAP criteria were developed. The Viterbi decoding strategy is then applied at recognition time.

The BML version of the algorithm required extensions in order to tackle the "divergence problem." For this reason, the SWS-ML, *Bayes*, and MAP versions of the algorithm were proposed, providing us with viable solutions to this problem, resulting in increased likelihood and improved recognition performance over the HMM. The MAP version of the algorithms results particularly effective.

ASR experiments allowed for the analysis of the behavior of the training schemes, along with an *ad hoc* initialization technique. The proposed models turn out to overcome the limitations of standard HMMs that were pointed out in Section I. In particular, the MAP algorithm applied to an MLP with sigmoidal hidden and output units compared favorably with a) CDHMMs with mixtures of Gaussian pdfs, b) Bourlard and Morgan approach, and c) with all the other versions of the proposed hybrid.

APPENDIX

I. PROOF OF THEOREM $\partial L/\partial\alpha_{i,t} = \beta_{i,t}$

This proof was given by Bengio [28], [29], [41] in the case of *log-likelihoods*. Let us consider $t = T$. In this assumption we can write

$$\begin{aligned}\frac{\partial L}{\partial \alpha_{i,T}} &= \frac{\partial \sum_{k\in\mathcal{F}} \alpha_{k,T}}{\partial \alpha_{i,T}} \\ &= \begin{cases} 1, & \text{if } i \in \mathcal{F} \\ 0, & \text{otherwise}\end{cases} \\ &= \beta_{i,T} \end{aligned} \tag{43}$$

which completes the inductive basis. Let us now assume that the theorem has been proved for $T, T-1, \ldots, t+1$ and let us show that it holds true for t.

Using the chain rule it is possible to write

$$\frac{\partial L}{\partial \alpha_{i,t}} = \sum_k \frac{\partial L}{\partial \alpha_{k,t+1}} \frac{\partial \alpha_{k,t+1}}{\partial \alpha_{i,t}} \tag{44}$$

which, using the inductive hypothesis and the definition (3) of $\alpha_{i,t}$, can be rewritten as

$$\begin{aligned}\frac{\partial L}{\partial \alpha_{i,t}} &= \sum_k \beta_{k,t+1} \frac{\partial b_{k,t+1} \sum_j a_{jk}\alpha_{j,t}}{\partial \alpha_{i,t}} \\ &= \sum_k \beta_{k,t+1} b_{k,t+1} a_{ik}\end{aligned} \tag{45}$$

that is, by definition (4) of $\beta_{i,t}$

$$\frac{\partial L}{\partial \alpha_{i,t}} = \beta_{i,t}. \tag{46}$$

This completes the proof. Q.E.D

II. IMPLEMENTATION ISSUES

A. On-Line Versus Batch Training

Given a training set $\mathcal{Y}$ containing a number of training sequences, i.e., $\mathcal{Y} = Y_1, Y_2, \ldots, Y_n$, a new training criterion $\mathcal{C}$ has to be considered, namely the overall likelihood $\mathcal{L}$ of the training sequences $\mathcal{Y}$ given the model $\mathcal{M}$

$$\mathcal{C} = P(\mathcal{Y}|\mathcal{M}) = \prod_{i=1}^{n} P(Y_i|\mathcal{M}). \tag{47}$$

Being the learning rule (21) aimed at the maximization of the criterion C, defined by (2), its sequential application to $Y_1, Y_2, \ldots, Y_n$ may possibly fail in the maximization of $\mathcal{C}$. The problem can be qualitatively stated by saying that the "local" maximization of the likelihood of a given acoustic sequence Y_i does not imply maximization of the global likelihood to any extent.

In order to maximize $\mathcal{C}$, the amount of change prescribed by gradient ascent for a given weight w in the ANN is

$$\Delta w = \eta \frac{\partial \mathcal{C}}{\partial w} = \eta \frac{\partial}{\partial w} \prod_{i=1}^{n} P(Y_i|\mathcal{M}) \tag{48}$$

where (47) has been taken into account. Taking the partial derivatives of the product with respect to w we can rewrite (48) in the following form

$$\Delta w = \eta \sum_{i=1}^{n} \left\{ \frac{\partial P(Y_i|\mathcal{M})}{\partial w} \prod_{j\neq i} P(Y_j|\mathcal{M}) \right\} \tag{49}$$

that implies the calculation of $\partial P(Y_i|\mathcal{M})/(\partial w)$ as in the on-line case, and prescribes to update the weights of the ANN according to a linear combination of individual contributes yielded by learning rule (21), computed over each sequence of the training set. We refer to (49) as the *batch* learning rule in the linear domain.

This scheme requires storing all the values of Δw for each weight in the ANN and for each training sequence Y_i, as well as the likelihoods $L_i = P(Y_i|\mathcal{M})$ for $i=1,\ldots,n$. If $P(Y_i|\mathcal{M})>0$ for each observation sequence Y_i, one way to reduce the complexity is to rewrite (49) as

$$\begin{aligned}\Delta w &= \eta \sum_{i=1}^{n} \left\{ \frac{\partial P(Y_i|\mathcal{M})}{\partial w} \frac{\prod_{j=1}^{n} P(Y_j|\mathcal{M})}{P(Y_i|\mathcal{M})} \right\} \\ &= \eta \left\{ \prod_{j=1}^{n} P(Y_j|\mathcal{M}) \right\} \sum_{i=1}^{n} \left\{ \frac{\partial P(Y_i|\mathcal{M})}{\partial w} \frac{1}{P(Y_i|\mathcal{M})} \right\}. \end{aligned} \quad (50)$$

For instance, the condition $P(Y_i|\mathcal{M}) > 0$ may be imposed by constraining the outputs of the ANN to be greater than zero. From an implementation viewpoint, applying (50) as a batch learning rule in the linear domain requires storing only: i) the value of $\Pi_{j=1}^{n} P(Y_j|\mathcal{M})$, updated multiplying by $P(Y_i|\mathcal{M})$ at the end of the forward pass over each training sequence $Y_1,\ldots,Y_n$; and ii) an *accumulator* for each weight of the ANN. Accumulators are set to zero before starting the training, and are updated after presentation of each new training sequence Y_i, by summing the quantity $(\partial P(Y_i|\mathcal{M})/\partial w)(1/P(Y_i|\mathcal{M}))$, computed via the on-line learning rule. As a matter of fact, given the arbitrariness in the choice of the learning rate η, the multiplicative factor $\Pi_{j=1}^{n} P(Y_j|\mathcal{M})$ in (50) could even be absorbed by η, defining a new learning rate $\eta' = \eta \Pi_{j=1}^{n} P(Y_j|\mathcal{M})$. This simplification further reduces the storage requirements, since keeping track of the product of the $P(Y_j|\mathcal{M})$ is no longer needed.

B. Learning in the Logarithmic Domain

Numerical stability considerations in the computation of forward $(\alpha_{i,t})$ and backward $(\beta_{i,t})$ probabilities over long paths in the *trellis* [1] may suggest to carry out calculations in the logarithmic domain. This Appendix develops an on-line learning rule in the logarithmic domain, assuming that $\log-\alpha$s and $\log-\beta$s are used along the trellis, and that the ANN outputs are interpreted as estimates of the natural logarithm of the emission probabilities associated with states of the HMM.

Let us consider again the criterion (2) for a given observation sequence Y. Using the same notation introduced in Section II, and given definition (3), we can rewrite (8) as

$$\begin{aligned}\frac{\partial L}{\partial w} &= \sum_i \sum_t \frac{\partial L}{\partial \log b_{i,t}} \frac{\partial \log b_{i,t}}{\partial w} \\ &= \sum_i \sum_t \frac{\partial L}{\partial \alpha_{i,t}} \frac{\partial \alpha_{i,t}}{\partial \log b_{i,t}} \frac{\partial \log b_{i,t}}{\partial w} \\ &= \sum_i \sum_t \beta_{i,t} \frac{\partial \alpha_{i,t}}{\partial \log \alpha_{i,t}} \frac{\partial \log \alpha_{i,t}}{\partial \log b_{i,t}} \frac{\partial \log b_{i,t}}{\partial w} \\ &= \sum_i \sum_t \beta_{i,t} \alpha_{i,t} \frac{\partial \log \alpha_{i,t}}{\partial \log b_{i,t}} \frac{\partial \log b_{i,t}}{\partial w} \\ &= \sum_i \sum_t \beta_{i,t} \alpha_{i,t} \frac{\partial \log b_{i,t}}{\partial w} \end{aligned} \quad (51)$$

since $\partial \log \alpha_{i,t} / \partial \log b_{i,t} = 1$. The ANN outputs $f_i(x_i(t))$ are now expected to yield the logarithms of the emission probabilities, and the quantity $\partial \log b_{i,t}/\partial w$ in (51) can be replaced by $\partial f_i(x_i(t))/\partial w$. The same calculations accomplished in Section II, in the linear case, to compute the latter quantity can still be applied ((9) and following), without any changes. The same definitions of $\delta_j(i,t)$ given therein can be used. From these considerations and from (51) the *on-line learning rule in the logarithmic domain* turns out to be the following:

$$\Delta w_{jk} = \eta \sum_{i=1}^{Q} \sum_{t=1}^{T} \beta_{i,t} \alpha_{i,t} \delta_j(i,t) o_k(t) \quad (52)$$

where w_{jk} is a generic connection weight between kth and jth units in adjacent layers of the ANN, the outputs of the latter are assumed to be logarithms of probabilities, the αs and βs are in the linear domain (explicit conversion via the exponential function is thus required), and the other quantities are defined in the same manner as in Section II.

The extension of the learning rule in the logarithmic domain to the batch case, i.e. whenever n training observation sequences $Y_1,\ldots,Y_n$ are considered, follows the steps outlined in Appendix II-A. In particular, the criterion (47) is used, and (50) still holds. The difference is that in the present case the on-line learning rule for the logarithmic domain, namely (51), is used to compute $\partial L_i/\partial w$, where L_i is the likelihood of ith observation sequence Y_i. The same considerations concerning implementation and numerical stability topics given in Appendix II-A apply to this case, too.

III. Batch and Logarithmic Versions of the Bayes Learning Rule

In the batch case, criterion (47) is adopted and (50) still holds, but the mathematics developed in Section III-C must be used to compute the term $\partial P(Y_i|\mathcal{M})/\partial w$ to be substituted into (50).

Concerning the algorithm in the logarithmic domain, (51) still holds. Let us now concentrate on the term $\partial \log b_{jt}/\partial w$ for j-th emission probability at time t, and for a generic ANN weight w. From (29) we can write

$$\frac{\partial \log b_{jt}}{\partial w} = \frac{\partial}{\partial w} \left\{ \log \Pr(q_j|\mathbf{y}_t) + \log p(\mathbf{y}_t) - \log \Pi_j \right\}. \quad (53)$$

Since $p(\mathbf{y}_t)$ and Π_j do not depend on w, (53) reduces to

$$\frac{\partial \log b_{jt}}{\partial w} = \frac{\partial}{\partial w} \log \Pr(q_j|\mathbf{y}_t) \quad (54)$$

that, according to (30), can be rewritten in the following form

$$\frac{\partial \log b_{jt}}{\partial w} = \frac{\partial}{\partial w} \left\{ \log f_j(x_j(t)) - \log \sum_{i=1}^{Q} f_i(x_i(t)) \right\}. \quad (55)$$

Calculating the partial derivatives of the logarithms and applying the same calculations as in the linear domain, namely

(32) and the following, an on-line learning rule in the logarithmic domain for a generic weight $w_{\iota k}$ can be obtained in the following form

$$\Delta w_{\iota k} = \eta \sum_{i=1}^{Q} \sum_{t=1}^{T} \beta_{i,t}\alpha_{i,t} \times \left\{ \frac{1}{f_i\left(x_i(t)\right)} \delta_\iota(i,t) o_k(t) - \frac{1}{\sum_{j=1}^{Q} f_j\left(x_j(t)\right)} \sum_{j=1}^{Q} \delta_\iota(j,t) o_k(t) \right\} \quad (56)$$

where the indexes have been properly adjusted.

Acknowledgment

The authors are grateful to R. De Mori, P. Frasconi, and N. Kasabov for their support and useful suggestions. The invaluable inspiration and contributions from our friend Y. Bengio are gratefully acknowledged. Very special thanks to M. Matassoni, who helped us in setting up the experimental environments, to F. Brugnara, and to M. Cettolo, for his peer review of the manuscript and the stimulating discussions. One of the authors (E. Trentin) is deeply grateful to N. Mana, for her constant and patient support.

References

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

[2] R. De Mori, *Spoken Dialogues With Computers*. London, UK: Academic, 1998.

[3] X. D. Huang, Y. Ariki, and M. Jack, *Hidden Markov Models for Speech Recognition*. Edinburgh, Scotland: Edinburgh Univ. Press, 1990.

[4] H. Bourlard and N. Morgan, *Connectionist Speech Recognition. A Hybrid Approach*. Boston, MA: Kluwer Academic, 1994, vol. 247, The Kluwer International Series in engineering and computer science.

[5] E. Trentin and M. Gori, "A survey of hybrid ANN/HMM models for automatic speech recognition," *Neurocomputing*, vol. 37, no. 1–4, pp. 91–126, Mar. 2001.

[6] E. Trentin, F. Brugnara, Y. Bengio, C. Furlanello, and R. De Mori, "Statistical and neural network models for speech recognition," in *Connectionist Approaches to Clinical Problems in Speech and Language*, R. Daniloff, Ed. Mahwah, NJ: Lawrence Erlbaum Associates, 2002.

[7] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford, U.K.: Oxford Univ. Press, 1995.

[8] A. Waibel, "Modular construction of time-delay neural networks for speech recognition," *Neural Computation*, vol. 1, pp. 39–46, 1989.

[9] R. P. Lippmann, "Review of neural networks for speech recognition," *Neural Computaion*, vol. 1, pp. 1–38, 1989.

[10] M. Gori, Y. Bengio, and R. De Mori, "BPS: A learning algorithm for capturing the dynamical nature of speech," in *Proc. IEEE Int. Joint Conf. Neural Networks*, Washington, DC, 1989, pp. 643–644.

[11] T. Robinson and F. Fallside, "A recurrent error propagation network speech recognition system," *Computer Speech and Language*, vol. 5, no. 3, pp. 259–274, July 1991.

[12] E. Trentin and M. Matassoni, "The regularized SNN-TA model for recognition of noisy speech," in *Proc. Int. Joint Conf. Neural Networks (IJCNN2000)*, Como, Italy, July 24–27, 2000, pp. V 97–V 102.

[13] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Networks*, vol. 5, pp. 157–166, Mar. 1994.

[14] M. A. Franzini, K. F. Lee, and A. Waibel, "Connectionist Viterbi training: A new hybrid method for continuous speech recognition," in *Proc. Int. Conf. Acoustics, Speech and Signal Processing*, Albuquerque, NM, 1990, pp. 425–428.

[15] E. Levin, "Word recognition using hidden control neural architecture," in *Proc. Int. Conf. Acoustics, Speech and Signal Processing*, Albuquerque, NM, 1990, pp. 433–436.

[16] L. T. Niles and H. F. Silverman, "Combining hidden Markov models and neural network classifiers," in *Proc. Int. Conf. Acoustics, Speech and Signal Processing*, Albuquerque, NM, 1990, pp. 417–420.

[17] D. Kimber, M. A. Bush, and G. N. Tajchman, "Speaker-independent vowel classification using hidden Markov models and LVQ2," in *Proc. Int. Conf. Acoustics, Speech and Signal Processing*, Albuquerque, NM, 1990, pp. 497–500.

[18] P. Haffner, M. Franzini, and A. Waibel, "Integrating time alignment and neural networks for high performance continuous speech recognition," in *Proc. Int. Conf. Acoustics, Speech and Signal Processing*, Toronto, ON, Canada, 1991, pp. 105–108.

[19] J. Tebelskis, A. Waibel, B. Petek, and O. Schmidbauer, "Continuous speech recognition using linked predictive networks," in *Advances in Neural Information Processing Systems 3*, R. P. Lippman, R. Moody, and D. S. Touretzky, Eds. San Mateo, CA: Morgan Kaufmann, 1991, pp. 199–205.

[20] G. Rigoll, "Maximum mutual information neural networks for hybrid connectionist-HMM speech recognition systems," *IEEE Trans. Speech Audio Processing*, vol. 2, pp. 175–1184, Jan. 1994.

[21] G. Zavaliagkos, Y. Zhao, R. Schwartz, and J. Makhoul, "A hybrid segmental neural net/hidden Markov model system for continuous speech recognition," *IEEE Trans. Speech Audio Processing*, vol. 2, pp. 151–160, Jan. 1994.

[22] S. Moon and J. N. Hwang, "Robust speech recognition based on joint model and feature space optimization of hidden Markov models," *IEEE Trans. Neural Networks*, vol. 8, pp. 194–204, Mar. 1997.

[23] P. Wilinski, B. Solaiman, A. Hillion, and W. Czamecki, "Toward the border between neural and markovian paradigms," *IEEE Trans. Systems, Man, Cybern.*, vol. 28, pp. 146–159, 1998.

[24] H. Bourlard and C. Wellekens, "Links between hidden Markov models and multilayer perceptrons," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 12, pp. 1167–1178, 1990.

[25] H. Bourlard and N. Morgan, "Continuous speech recognition by connectionist statistical methods," *IEEE Trans. Neural Networks*, vol. 4, pp. 893–909, Nov. 1993.

[26] Y. Bengio, R. De Mori, G. Flammia, and R. Kompe, "Global optimization of a neural network-hidden Markov model hybrid," *IEEE Trans. Neural Networks*, vol. 3, pp. 252–259, 1992.

[27] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing*, D. E. Rumelhart and J. L. McClelland, Eds. Cambridge, MA: MIT Press, 1986, vol. 1, ch. 8, pp. 318–362.

[28] Y. Bengio, "A connectionist approach to speech recognition," *Int. J. Pattern Recogn. Artificial Intell.*, vol. 7, no. 4, pp. 647–667, 1993.

[29] ——, *Neural Networks for Speech and Sequence Recognition*. London, U.K.: Int. Thomson Computer Press, 1996.

[30] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, pp. 257–286, 1989.

[31] J. S. Bridle, "Alphanets: A recurrent 'neural' network architecture with a hidden Markov model interpretation," *Speech Commun.*, vol. 9, no. 1, pp. 83–92, 1990.

[32] E. Trentin. (2001) Robust combination of neural networks and hidden Markov models for speech recognition. Univ. Florence, Florence , Italy, Dipl. di Sistemi e Informatica, Università di Firenze (Ph.D. dissertation). [Online] http://www.dii.unisi.it/~trentin/ThesisDownload.html

[33] M. J. D. Powell, "Radial basis functions for multivariable interpolation: a review," in *Algorithms for Approximation: IMA-1985 Conference*, J. C. Mason and M. G. Cox, Eds. Oxford, U.K: Clarendon, 1987.

[34] T. Poggio and F. Girosi, "Networks for approximation and learning," *Proc. IEEE*, vol. 78, pp. 1481–1497, 1990.

[35] D. Plaut, S. Nowlan, and G. Hinton, "Experiments on Learning by Back-Propagation," Dept. Computer Science, Carnegie Mellon Univ., Pittsburgh, PA, Tech. Rep. CMU-CS-86-126, 1986.

[36] S. J. Nowlan and G. E. Hinton, "Simplifying neural networks by soft weight-sharing," *Neural Computation*, vol. 4, no. 4, pp. 473–493, 1992.

[37] E. Trentin, "Networks with trainable amplitude of activation functions," *Neural Networks*, vol. 14, pp. 471–493, May 2001.

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

[39] E. Trentin, Y. Bengio, C. Furlanello, and R. De Mori, "Neural networks for speech recognition," in *Spoken Dialogues With Computers*, R. De Mori, Ed. London, U.K.: Academic, 1998, pp. 311–361.

[40] S. B. Davis and P. Mermelstein, "Comparison of parametric representations of monosyllabic word recognition in continuously spoken sentences," *IEEE Trans. Acoust., Speech Signal Processing*, vol. ASSP-28, pp. 357–366, 1980.

[41] Y. Bengio, R. Mori, G. Flammia, and R. Kompe, "Global optimization of a neural network–hidden Markov model hybrid," *IEEE Trans. Neural Networks*, vol. 3, pp. 252–259, 1992.

Edmondo Trentin received the Laurea degree *cum laude* in computer science from the Università di Milano, Italy, in 1990, and the Ph.D. degree in automation and information engineering from the Universita di Firenze, Italy, in 2001.

From 1990 to the end of 1993, he worked as a Project Leader at SGS-Elsag. From 1994 to 2000, he was a researcher at ITC-irst (Trento, Italy), working in the area of interactive sensory systems. Since December 2000, he has been carrying out his research and teaching activities as a Research Associate with the Dipartimento di Ingegneria dell'Informazione, Università di Siena, Italy. His research interests include learning, artificial neural networks (ANNs), statistical pattern recognition, hidden Markov models (HMM), and hybrid ANN/HMM systems. He has been involved in projects of scientific and technological relevance in the fields of speech processing and bioinformatics. He is the author of more than 30 scientific publications.

Dr. Trentin is a member of International Neural Network Society (INNS), Italian Neural Networks Society (SIREN), and International Association for Pattern Recognition, Italian Chapter (IAPR-IC).

Marco Gori (S'88–M'91–SM'97–F'01) received the Laurea degree in electronic engineering from Università di Firenze, Italy, in 1984, and the Ph.D. degree in 1990 from the Università di Bologna, Italy.

From October 1988 to June 1989, he was a visiting student with the School of Computer Science, McGill University, Montreal. In 1992, he became an Associate Professor of computer science at Università di Firenze and, in November 1995, he joined the University of Siena, where he is currently Full Professor. His main research interests are in neural networks, pattern recognition, and applications of machine learning to information retrieval on the Internet. He has lead a number of research projects on these themes with either national or international partners, and has been involved in the organization of many scientific events, including the IEEE-INNS International Joint Conference on Neural Networks, for which he acted as the program chair (July 24–28, 2000).

Dr. Gori serves as an Associate Editor of a number of technical journals related to his areas of expertise, including *Pattern Recognition*, the IEEE TRANSACTIONS ON NEURAL NETWORKS, *Neurocomputing*, and the *International Journal on Pattern Recognition and Artificial Intelligence*. He is the Italian chairman of the IEEE Neural Network Council (R.I.G.), is acting as the co-chair of the TC3 technical committee of the International Association for Pattern Recognition (IAPR) on Neural Networks, and is the President of the Italian Association for Artificial Intelligence.