Deep Dynamic Models for Learning Hidden Representations of Speech Features

Li Deng and Roberto Togneri

Abstract Deep hierarchical structure with multiple layers of hidden space in human speech is intrinsically connected to its dynamic characteristics manifested in all levels of speech production and perception. The desire and an attempt to capitalize on a (superficial) understanding of this deep speech structure helped ignite the recent surge of interest in the deep learning approach to speech recognition and related applications [28, 29, 51], and a more thorough understanding of the deep structure of speech dynamics and the related computational representations is expected to further advance the research progress in speech technology. In this chapter, we first survey a series of studies on representing speech in a hidden space using dynamic systems and recurrent neural networks, emphasizing different ways of learning the model parameters and subsequently the hidden feature representations of time-varying speech data. We analyze and summarize this rich set of deep, dynamic speech models into two major categories: 1) top-down, generative models adopting localist representations of speech classes and features in the hidden space; and 2) bottom-up, discriminative models adopting distributed representations. With detailed examinations of and comparisons between these two types of models, we focus on the localist versus distributed representations as their respective hallmarks and defining characteristics. Future directions are discussed and analyzed of ways to leverage the strengths of both the localist and distributed representations while overcoming their respective weaknesses, beyond blind integration of the two by using the generative model to pre-train the discriminative one as a popular method of training deep neural networks.

Li Deng
Microsoft Research, Redmond WA 98034 USA, e-mail: deng@microsoft.com

Roberto Togneri
The University of Western Australia, Crawley WA 6009, Australia e-mail: Roberto.Togneri@uwa.edu.au

1 Introduction

Before around 2010-2011, speech recognition technology had been dominated by a "shallow" architecture — hidden Markov models (HMMs) with each state characterized by a Gaussian mixture model (GMM). While significant technological success had been achieved using complex and carefully engineered variants of GMM-HMMs and acoustic features suitable for them, researchers long before that time had clearly realized that the next generation of speech recognition technology would require solutions to many new technical challenges under diversified deployment environments and that overcoming these challenges would likely require "deep" architectures that can at least functionally emulate the human speech system known to have dynamic and hierarchical structure in both production and perception [29, 36, 41, 96]. An attempt to incorporate a primitive level of understanding of this deep speech structure, initiated at the 2009 NIPS Workshop on Deep Learning for Speech Recognition and Related Applications [29], has helped create an impetus in the speech recognition community to pursue a deep representation learning approach based on the deep neural network (DNN) architecture, which was pioneered by the machine learning community only a few years earlier [52, 53] but rapidly evolved into the new state of the art in speech recognition with industry-wide adoption [16, 28, 29, 51, 59, 78, 93, 94, 108]. In the mean time, it has been recognized that the DNN approach (with its interface to the HMM) has not modeled speech dynamics properly. The deep and temporally recurrent neural network (RNN) has been developed to overcome this problem [13, 49], where the internal representation of dynamic speech features is discriminatively formed by feeding the low-level acoustic features into the hidden layer together with the recurrent hidden features from the past history. Even without stacking RNNs one on top of another as carried out in [49] or feeding DNN features as explored in [13], an RNN itself is a deep model since temporal unfolding of the RNN creates as many layers in the network as the length of the input speech utterance.

On the other hand, before the recent rise of deep learning for speech modeling and recognition, many earlier attempts had been made to develop computational architectures that are "deeper" than the conventional GMM-HMM architecture. One prominent class of such models are hidden dynamic models where the internal representation of dynamic speech features is generated probabilistically from the higher levels in the overall deep speech model hierarchy [12, 19, 23, 37, 65, 86, 92, 102]. Despite separate developments of the RNNs and of the hidden dynamic models, they share the same motivation — more realistically representing the dynamic structure of speech. Nevertheless, the different ways in which these two types of deep dynamic models are constructed endow them with distinct pros and cons. Investigations of the contrast between the two model types and the similarity to each other will yield insights into the strategies for developing new types of deep dynamic models with the hidden representations of speech features superior to the existing RNNs and hidden dynamic models. This forms the main motivation of this chapter.

In this chapter, we will focus on the most prominent contrast between the above two types of models in terms of the opposing localist and distributed representations

adopted by the hidden layers in the models. In the distributed representation adopted by the RNNs, we cannot interpret the meaning of activity on a single unit or neuron in isolation. Rather, the meaning of the activity on any particular unit depends on the activities of other units. Using distributed representations, multiple concepts (i.e., phonological/linguistic symbols) can be represented at the same time on the same set of neuronal units by superimposing their patterns together. The strengths of distributed representations used by the RNN include robustness, representational and mapping efficiency, and the embedding of symbols into continuous-valued vector spaces which enable the use of powerful gradient-based learning methods. The localist representation adopted by the generative hidden dynamic models has very different properties. It offers very different advantages — easy to interpret, understand, diagnose, and easy to work with. Section 6 of this chapter will compare the two types of models, in terms of the localist versus distributed representations as well as other attributes, with respect to their strengths and weaknesses in detail. Based on such comparisons, Section 7 will discuss how to exploit the advantages of both types of models and of the representations they have adopted while circumventing their weaknesses. Before that and in Sections 2 to 5, we will first provide a detailed review on major deep dynamic models in the literature relevant to our topic, focusing on the algorithms for learning model parameters from data and for computing the representations in the hidden spaces.

2 Generative Deep-Structured Speech Dynamics: Model Formulation

2.1 Generative Learning in Speech Recognition

In speech recognition, the most common generative learning approach is based on the GMM-HMM; e.g., [9, 30, 58, 88]. A GMM-HMM is a model that describes two dependent random processes, an observable process $\mathbf{x}_{1:T}$ and a hidden Markov process $y_{1:T}$. The observation x_t is assumed to be "generated" by the hidden state y_t according to a Gaussian mixture distribution. The GMM-HMM can be parameterized by $\lambda = (\pi, A, B)$; π is a vector of state prior probabilities; $A = (a_{i,j})$ is a state transition probability matrix; and $B = \{b_1, \ldots, b_n\}$ is a set where b_j represents the Gaussian mixture model of state j. The state is typically associated with a subsegment of a phone in speech. One important innovation in speech recognition is the introduction of context-dependent states (e.g. [32, 88]), motivated by the desire to reduce output variability associated with each state, a common strategy for "detailed" generative modeling. A consequence of using context dependency is a vast expansion of the HMM state space, which, fortunately, can be controlled by regularization methods such as state tying.

The introduction of the HMM and the related statistical methods to speech recognition in mid 1970s [2, 57] can be regarded the most significant paradigm shift in

the field, as discussed in [3]. One major reason for this early success was due to the highly efficient maximum likelihood learning method invented about ten years earlier [5]. This MLE method, often called the Baum-Welch algorithm, had been the principal way of training the HMM-based speech recognition systems until 2002, and is still one major step (among many) in training these systems nowadays. It is interesting to note that the Baum-Welch algorithm serves as one major motivating example for the later development of the more general Expectation-Maximization (EM) algorithm [17].

The goal of maximum likelihood learning is to minimize an empirical risk with respect to the joint likelihood loss (extended to sequential data), *i.e.*,

$$R_{\text{emp}}(f) = -\sum_i \ln p(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}; \pi, A, B) \tag{1}$$

where $\mathbf{x}$ represents acoustic data, usually in form of a sequence feature vectors extracted at frame-level; $\mathbf{y}$ represents a sequence of linguistic units. It is crucial to apply certain type of regularization to improve generalization. This leads to a practical training objective referred to as *accuracy-regularization* which takes the following general form:

$$J(f) = R_{\text{emp}}(f) + \gamma C(f) \tag{2}$$

where $C(f)$ is a regularizer that measures "complexity" of f, and γ is a tradeoff parameter. In large-vocabulary speech recognition systems, it is normally the case that word-level labels are provided, while state-level labels are latent. Moreover, in training HMM-based speech recognition systems, parameter tying is often used as a type of regularization [55]. For example, similar acoustic states of the triphones can share the same Gaussian mixture model. In this case, the $C(f)$ term is expressed by

$$C(f) = \prod_{(m,n)\in\mathcal{T}} \delta(b_m = b_n) \tag{3}$$

where $\mathcal{T}$ represents a set of tied state pairs.

The use of the generative model of HMMs, including the most popular Gaussian-mixture HMM, for representing the (piece-wise stationary) dynamic speech pattern and the use of MLE for training the tied HMM parameters constitutes one of the most prominent and successful examples of generative learning in speech recognition. This success was firmly established by the speech recognition community, and has been widely spread to the machine learning and related communities; in fact, the HMM has become a standard tool not only in speech recognition but also in machine learning and their related fields such as bioinformatics and natural language processing. For many machine learning as well as speech recognition researchers, the success of the HMM in speech recognition is a bit surprising due to the well-known weaknesses of the HMM.

Another clear success of the generative learning paradigm in speech recognition is the use of the GMM-HMM as prior "knowledge" within the Bayesian framework for environment-robust speech recognition. The main idea is as follows. When the speech signal, to be recognized, is mixed with noise or another non-intended

speaker, the observation is a combination of the signal of interest and interference of no interest, both unknown. Without prior information, the recovery of the speech of interest and its recognition would be ill defined and subject to gross errors. Exploiting generative models of GMM-HMMs, or often simpler GMMs, as Bayesian priors for "clean" speech overcomes the ill-posed problem. Further, the generative approach allows probabilistic construction of the model for the relationship between the noisy speech observation, clean speech, and interference, which is typically nonlinear when the log-domain features are used. A set of generative learning approaches in speech recognition following this philosophy are variably called "parallel model combination" [45], vector Taylor series (VTS) method [1, 26], and Algonquin [44]. Notably, the comprehensive application of such a generative learning paradigm for single-channel multitalker speech recognition is reported and reviewed in [89], where the authors apply successfully a number of well established ML methods including loopy belief propagation and structured mean-field approximation. Using this generative learning scheme, speech recognition accuracy with loud interfering speakers is shown to exceed human performance.

Despite some success of GMM-HMMs in speech recognition, their weaknesses, such as the conditional independence assumption, have been well known for speech recognition applications [3,4]. Since the early 1990's, speech recognition researchers have begun the development of statistical models that capture the dynamic properties of speech in the temporal dimension more faithfully than HMMs. This class of beyond-HMM models have been variably called the stochastic segment model [81, 82], trended or nonstationary-state HMM [18, 24], trajectory segmental model [54, 81], trajectory HMMs [63, 111, 112], stochastic trajectory models [47], hidden dynamic models [12, 19, 23, 37, 65, 86, 92, 102], buried Markov models [8], structured speech model [40], and the hidden trajectory model [39] depending on different "prior knowledge" applied to the temporal structure of speech and on various simplifying assumptions to facilitate the model implementation. Common to all these beyond-HMM models is some temporal trajectory structure built into the models, hence trajectory models. Based on the nature of such a structure, we can classify these models into two main categories. In the first category are the models focusing on a temporal correlation structure at the "surface" acoustic level. The second category consists of hidden dynamics, where the underlying speech production mechanisms are exploited as the Bayesian prior to represent the temporal structure that accounts for the observed speech pattern. When the mapping from the hidden dynamic layer to the observation layer is limited to linear (and deterministic), then the generative hidden dynamic models in the second category reduces to the first category.

The temporal span of the generative trajectory models in both categories above is controlled by a sequence of linguistic labels, which segment the full sentence into multiple regions from left to right; hence segment models.

In a general form, the trajectory/segment models with hidden dynamics make use of the switching state space formulation. They use temporal recursion to define the hidden dynamics, $\mathbf{z}(k)$, which may correspond to articulatory movement during human speech production. Each discrete region or segment, s, of such dynamics is

characterized by the s-dependent parameter set $\boldsymbol{\Lambda}_s$, with the "state noise" denoted by $\mathbf{w}_s(k)$. The memory-less nonlinear mapping function is exploited to link the hidden dynamic vector $\mathbf{z}(k)$ to the observed acoustic feature vector $\mathbf{o}(k)$, with the "observation noise" denoted by $\mathbf{v}_s(k)$, and parameterized also by segment-dependent parameters. The combined "state equation" (4) and "observation equation" (5) below form a general switching nonlinear dynamic system model:

$$\mathbf{z}(k+1) = \mathbf{g}_k[\mathbf{z}(k), \boldsymbol{\Lambda}_s] + \mathbf{w}_s(k) \tag{4}$$

$$\mathbf{o}(k') = \mathbf{h}_{k'}[\mathbf{z}(k'), \boldsymbol{\Omega}_{s'}] + \mathbf{v}_{s'}(k'). \tag{5}$$

where subscripts k and k' indicate that the functions $\mathbf{g}[.]$ and $\mathbf{h}[.]$ are time varying and may be asynchronous with each other. s or s' denotes the dynamic region correlated with phonetic categories.

The model expressed by (4) and (5) is not only dynamic, but also deep since there is a hierarchy of information flow from discrete linguistic symbols s to the hidden dynamic vector $\mathbf{z}(k)$ and then to the observed vectors $\mathbf{o}(k)$. We call this type of model a generative deep-structured dynamic model. Being "generative" here means that the model provides a causal relationship from the (top) linguistic labels to intermediate and then to the (bottom) observed acoustic variables. This distinguishes from the "discriminative" deep-structured models where the information flow starts from the (bottom) observed acoustic variables to the intermediate representations and then to the (top) linguistic labels.

There have been several studies on switching nonlinear state space models for speech recognition, both theoretical [21, 37] and experimental [12, 61, 65, 86]. The specific forms of the functions of $\mathbf{g}_k[\mathbf{z}(k), \boldsymbol{\Lambda}_s]$ and $\mathbf{h}_{k'}[\mathbf{z}(k'), \boldsymbol{\Omega}_{s'}]$ and their parameterization are determined by prior knowledge based on the current understanding of the nature of the temporal dimension in speech. In particular, state equation (4) takes into account the temporal elasticity in spontaneous speech and its correlation with the "spatial" properties in hidden speech dynamics such as articulatory positions or vocal tract resonance frequencies; see [23] for a comprehensive review of this body of work.

When nonlinear functions of $\mathbf{g}_k[\mathbf{z}(k), \boldsymbol{\Lambda}_s]$ and $\mathbf{h}_{k'}[\mathbf{z}(k'), \boldsymbol{\Omega}_{s'}]$ in (4) and (5) are reduced to linear functions (and when synchrony between the two equations are eliminated), the switching nonlinear dynamic system model is reduced to its linear counterpart, the switching linear dynamic system. It can be viewed as a hybrid of standard HMMs and linear dynamical systems, with a general mathematical description of

$$\mathbf{z}(k+1) = \mathbf{A}_s\mathbf{z}(k) + \mathbf{B}_s\mathbf{w}_s(k) \tag{6}$$

$$\mathbf{o}(k) = \mathbf{C}_s\mathbf{z}(k) + \mathbf{v}_s(k). \tag{7}$$

There has also been an interesting set of work on the switching linear dynamic system applied to speech recognition. The early set of studies have been carefully reviewed in [81] for generative speech modeling and for its speech recognition applications. The studies reported in [42, 72] further applied this system model to

noise-robust speech recognition and explored several approximate inference techniques, overcoming intractability in decoding and parameter learning. The study reported in [91] applied another approximate inference technique, a special type of Gibbs sampling commonly used in machine learning, to an speech recognition problem.

During the development of trajectory/segment models for speech recognition, a number of machine learning techniques invented originally in non-speech recognition communities, e.g. variational learning [61], pseudo-Bayesian [42, 65], Kalman filtering [81], extended Kalman filtering [23, 37, 101], Gibbs sampling [91], orthogonal polynomial regression [24], etc., have been usefully applied with modifications and improvement to suit the speech-specific properties and speech recognition applications. However, the success has mostly been limited to small-scale tasks. We can identify four main sources of difficulty (as well as new opportunities) in successful applications of trajectory/segment models to large-scale speech recognition. First, scientific knowledge on the precise nature of the underlying articulatory speech dynamics and its deeper articulatory control mechanisms is far from complete. Coupled with the need for efficient computation in training and decoding for speech recognition applications, such knowledge has been forced to be again simplified, reducing the modeling power and precision further. Second, most of the work in this area has been placed within the generative learning setting, having a goal of providing parsimonious accounts (with small parameter sets) for speech variations due to contextual factors and co-articulation. In contrast, the recent joint development of deep learning by both ML and speech recognition communities, which we will review in Section 6, combines generative and discriminative learning paradigms and makes use of massive instead of parsimonious parameters. There is a huge potential for synergy of research here. Third, although structural ML learning of switching dynamic systems via Bayesian nonparametrics has been maturing and producing successful applications in a number of ML and signal processing tasks (e.g. the tutorial paper [43]), it has not entered mainstream speech recognition; only isolated studies have been reported on using Bayesian nonparametrics for modeling aspects of speech dynamics [83] and for language modeling [14]. Finally, most of the trajectory/segment models developed by the speech recognition community have focused on only isolated aspects of speech dynamics rooted in deep human production mechanisms, and have been constructed using relatively simple and largely standard forms of dynamic systems.

In the remainder of this section, we will review two special cases of the general dynamic models of speech represented by (4) to (7) with hidden structure. These models are considered to be “deep”, in that the hidden structure is modeled as an intermediate information processing stage connecting the linguistic information to the observable acoustics.

2.2 A Hidden Dynamic Model with Nonlinear Observation Equation

Let us consider in detail the hidden dynamic model (HDM) using the extended Kalman filter [102]. The hidden dynamics is chosen to be the vocal-tract-resonances (VTRs), which are closely related to the smooth and target-oriented movement of the articulators. The first component of the HDM, also called the state equation, is a target-directed, continuously-valued (hidden) Markov process that is used to describe the hidden VTR dynamics according to:

$$\mathbf{z}(k+1) = \mathbf{\Phi}_s \mathbf{z}(k) + (\mathbf{I}_m - \mathbf{\Phi}_s)\mathbf{T}_s + \mathbf{w}(k) \tag{8}$$

where $\mathbf{z}(k)$ is the $m \times 1$ VTR state vector, $\mathbf{T}_s$ is the $m \times 1$ phone target vector parameter and $\mathbf{\Phi}_s$ is the $m \times m$ diagonal "time-constant" matrix parameter associated with the phone regime s. The phone regime is used to describe the segment of speech that is attributed to the phone identified by the model pair $(\mathbf{\Phi}_s, \mathbf{T}_s)$. The process noise $\mathbf{w}(k)$ is an i.i.d, zero-mean, Gaussian process with covariance $\mathbf{Q}$. The target-directed nature of the process is evident by noting that $\mathbf{z}(k) \to \mathbf{T}_s$ as $k \to \infty$ independent of the initial value of the state.

The second component of the HDM is the observation equation used to describe the static mapping from the lower dimensional hidden VTR state vector (typically $m = 3$ for the first three VTR resonances) to the higher dimensional observable acoustic feature vector. The general form of this mapping adopted in the current study assumes a static, multivariate nonlinear mapping function as follows:

$$\mathbf{o}(k) = h_r(\mathbf{z}(k)) + \mathbf{v}(k). \tag{9}$$

where the $n \times 1$ acoustic observation $\mathbf{o}(k)$ is the set of acoustic feature vectors for frame k (the usual Mel-frequency cepstral co-efficient (MFCC) features with $n = 12$), and $h_r(\mathbf{z}(k))$ is the $n \times m$ static, non-linear mapping function on the state vector $\mathbf{z}(k)$ associated with the manner of articulation r. The manner of articulation describes how the phone is articulated to produce the acoustic observations arising from the speech production process and will usually be different for the different broad phonetic classes (e.g. vowels, voiced stops, etc.) . The observation noise $\mathbf{v}(k)$ is an i.i.d, zero-mean, Gaussian process with covariance $\mathbf{R}$. The multivariate mapping function $h_r(\mathbf{z}(k))$ is implemented by a m-J-n feedforward multi-layer perceptron (MLP) with J hidden nodes, a linear activation function on the output layer, and the antisymmetric hyperbolic tangent function on the hidden layer. There is a unique MLP network for each distinct r.

The switching state behaviour of this model is represented by an M-state discrete-time random sequence, where $s \equiv s(k) \in [1, 2 \ldots, M]$ is a random variable that takes on one of the M possible "phone" regimes (or states) at time k. An additional R-state discrete-time random sequence also exists where $r \equiv r(k) \in [1, 2, \ldots R]$ is a random variable that takes on one of the R possible manner of articulation states at time k. In practice both sequences are unknown and need to be estimated, both when training

the model (i.e. estimating the parameters) and testing (i.e. using the model to rescore or decode an unknown observation sequence).

An important property of this model is the continuity of the hidden state variable $\mathbf{z}(k)$ across phone regimes. That is, $\mathbf{z}(k)$ at the start of segment $l+1$ is set to the value computed at the end of segment l. This provides a long-span continuity constraint across adjacent phone regimes that structurally models the inherent context dependencies and coarticulatory effects [35].

An important concern is the specific modelling of the state dynamic and observation process. The target-directed state dynamic is reasonable but requires knowledge of the per-phone target and time-constant values. If these are not known these have to be jointly estimated. The non-linear mapping from the state vector to observation vector is more problematic as the MLP weights also have to be estimated and this creates a system with too many degrees of freedom. Possible solutions to do this have included: using prior VTR measurement data to independently train the MLP [102], using a more simple linear mapping [61], or restricting to observation features like LPC cepstra which permit an analytical mapping with the VTR resonances [31]. Finally we also assume that the phone sequence or segmentation of model regimes, $s(k)$, is known in advance, which, in practice, requires training on phonetically transcribed speech corpora.

2.3 A Linear Hidden Dynamic Model Amenable to Variational EM Training

An alternative approach to implementing the hidden dynamic model is to reformulate it in the context of a segmental switching state space model and to apply the variational EM algorithm to learn the model parameters. The state equation and observation equation in this reformulated model, as described in [61], are

$$\mathbf{x}_n = \mathbf{A}_s \mathbf{x}_{n-1} + (\mathbf{I} - \mathbf{A}_s)\mathbf{u}_s + \mathbf{w}, \tag{10}$$

$$\mathbf{y}_n = \mathbf{C}_s \mathbf{x}_n + \mathbf{c}_s + \mathbf{v}, \tag{11}$$

where n and s are frame number and phone index respectively, $\mathbf{x}$ is the hidden dynamics and $\mathbf{y}$ is the acoustic feature vector (such as MFCC). The hidden dynamics is chosen to be the vocal-tract-resonances (VTRs). The state equation (10) is a linear dynamic equation with phone dependent system matrix $\mathbf{A}_s$ and target vector $\mathbf{u}_s$ and with built-in continuity constraints across the phone boundaries. The observation equation (11) represents a phone-dependent VTR-to-acoustic linear mapping. The choice of linear mapping is mainly due to the difficulty of algorithm development. The resulting algorithm can also be generalized to mixtures of linear mapping and piece-wise linear mapping within a phone. Gaussian white noises $\mathbf{w}_n$ and $\mathbf{v}_n$ are added to both the state and observation equations to make the model probabilistic. Similar models have been proposed and used previously [35, 62].

To facilitate algorithm development, the model is also expressed in terms of probability distributions:

$$\begin{aligned} p(s_n = s \mid s_{n-1} = s') &= \pi_{ss'}, \\ p(\mathbf{x}_n \mid s_n = s, \mathbf{x}_{n-1}) &= \mathcal{N}(\mathbf{x}_n \mid \mathbf{A}_s x_{n-1} + \mathbf{a}_s, \mathbf{B}_s), \\ p(\mathbf{y}_n \mid s_n = s, \mathbf{x}_n) &= \mathcal{N}(\mathbf{y}_n \mid \mathbf{C}_s \mathbf{x}_n + \mathbf{c}_s, \mathbf{D}_s), \end{aligned} \tag{12}$$

where $\pi_{ss'}$ is the phone transition probability matrix, $\mathbf{a}_s = (\mathbf{I} - \mathbf{A}_s)\mathbf{u}_s$ and $\mathcal{N}$ denotes a Gaussian distribution with mean and precision matrix (inverse of the covariance matrix) as the parameters. The joint distribution over the entire time sequence is given by

$$p(\mathbf{y}_{1:N}, \mathbf{x}_{1:N}, s_{1:N}) = \prod_n p(\mathbf{y}_n|s_n, \mathbf{x}_n) p(\mathbf{x}_n|s_n, \mathbf{x}_{n-1}) p(s_n|s_{n-1}). \tag{13}$$

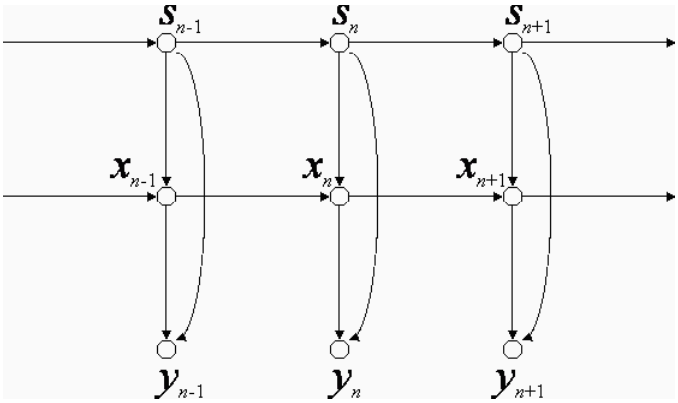

Fig. 1 HDM represented as a Bayesian network

The conditional independence relations of the model can be seen more clearly from a graphic form (Bayesian network) as shown in Fig. 1.

There are a few issues to be solved before any estimation or learning algorithms can be applied to speech, and they are discussed here:

1. Parameter initialization: It is important to initialize the parameters appropriately for an iterative local optimization procedure such as EM. The HDM enjoys the benefit of being closely related to speech-specific knowledge and some key parameters, especially the phone targets, can be reliably initialized from a formant synthesizer. Due to the small number of total parameters, others can be easily initialized by a small amount of hand-labeled VTR data.
2. Segmental constraint: The probabilistic form of the model allows phone transitions to occur at each frame, which is undesirable for speech. In training, we construct a series of time-varying transition matrices $\pi_{ss'}$ based on the given phonetic transcript (or one created from a lexicon if only word transcripts are given)

and some initial segmentation to impose the segmental constraint and force the discrete-state component of the model to be consistent with the phonetic transcript. Such an approach also greatly reduces the number of possible phones s that have to be summed at each time step.

3 Generative Deep-Structured Speech Dynamics: Model Estimation

3.1 Learning a Hidden Dynamic Model Using the Extended Kalman Filter

The estimation problem that we investigate is as follows. Given multiple sets of observation sequences, $\mathbf{o}(k)$, for each distinct phone regime, we seek to determine the optimal estimates for the unknown values of the state-equation parameters $\mathbf{\Phi}$ and $\mathbf{T}$, and the observation-equation parameters, $\mathbf{W}$, which is the MLP weight vector of the nonlinear mapping function $h(\mathbf{z}(k))$. For clarity of notation we will drop the s and r subscripts since it is understood the estimation equations only apply for observations taken over a particular phone regime segment.

The expectation-maximisation (EM) algorithm is a widely used algorithm for the estimation of the parameters in general state-space models and in the current research on the HDM [34, 35]. The EM algorithm provides new estimates of the parameters after the set of all available N observation vectors have been presented. The EM algorithm can be considered a batch or off-line estimation method most suited to applications where all of the data is available. We now present the EM algorithm for the specific type of model given by (8) and (9) following [101, 102].

E-step

For a sequence of N observation vectors, the E-step involves computation of the conditional expectation of the log joint likelihood between $\mathbf{Z} = \{\mathbf{z}(0), \mathbf{z}(1), \ldots, \mathbf{z}(N)\}$ and $\mathbf{O} = \{\mathbf{o}(0), \mathbf{o}(1), \ldots, \mathbf{o}(N)\}$ given the observation $\mathbf{O}$ and parameter set $\overline{\Theta}$ estimated at the previous step, that is:

$$\begin{aligned} Q(\Theta|\overline{\Theta}) &= E\{\log L(\mathbf{Z},\mathbf{O}|\Theta)|\mathbf{O},\overline{\Theta}) \\ &= -\frac{1}{2}\sum_{k=0}^{N-1} E_N[\mathbf{e}_{k1}^T\mathbf{Q}^{-1}\mathbf{e}_{k1}|\mathbf{O},\overline{\Theta}] \\ &\quad -\frac{1}{2}\sum_{k=0}^{N-1} E_N[\mathbf{e}_{k2}^T\mathbf{R}^{-1}\mathbf{e}_{k2}|\mathbf{O},\overline{\Theta}] + const \end{aligned} \tag{14}$$

where $\mathbf{e}_{k1} = [\mathbf{z}(k+1) - \mathbf{\Phi}\mathbf{z}(k) - (\mathbf{I}-\mathbf{\Phi})\mathbf{T}]$ and $\mathbf{e}_{k2} = [\mathbf{o}(k) - h(\mathbf{z}(k))]$ and E_N denotes the expectation based on N samples. The standard EKF smoother is used to provide estimates of the hidden dynamic variable, $\mathbf{z}(k) \equiv \widehat{\mathbf{z}}(k|N) = E_N[\mathbf{z}(k)|\mathbf{O},\overline{\Theta}]$. The Jacobian matrix for the $n \times m$ non-linear mapping function $h(\mathbf{z}(k))$ used in the EKF recursion is given by:

$$\begin{aligned} H_z^{ji}[\widehat{\mathbf{z}}(k+1|k)] &= \left[\frac{\partial o_j(k+1)}{\partial \widehat{z}_i(k+1|k)}\right] \\ &= \left[\sum_{h=1}^{J} W_{2j}^h g^{'}(\mathbf{W}_{1h}^T \widehat{\mathbf{z}}(k+1|k)) W_{1h}^i\right] \end{aligned} \tag{15}$$

where $o_j(k)$ is the j^{th} component of the observation vector at time k, $\widehat{z}_i(k+1|k)$ is the i^{th} component of the predicted state vector $\widehat{z}(k+1|k)$ at time k, W_{lh}^i is the i^{th} component of the MLP weight vector, $\mathbf{W}_{lh}$, of node h in layer l (layer 1 is the hidden layer and layer 2 is the output layer), J is the number of nodes in the hidden layer and $g^{'}(x)$ is the derivative of the activation function in the hidden layer.

It should be noted that the continuity condition on $\widehat{\mathbf{z}}(k)$ is also applied to the EKF error covariance.

M-step

In the M-step the Q function in (14) is maximised with respect to the parameter set $\Theta = (\mathbf{T}, \mathbf{\Phi}, \mathbf{W})$. We consider the first summation involving the parameters $\mathbf{T}$ and $\mathbf{\Phi}$:

$$Q_1(\mathbf{Z},\mathbf{O},\Theta) = \sum_{k=0}^{N-1} E_N[\mathbf{e}_{k1}^T \mathbf{Q}^{-1} \mathbf{e}_{k1} | O, \overline{\Theta}] \tag{16}$$

Minimisation of Q_1 ,which implies maximisation of Q , proceeds by setting the partial derivatives with respect to $\mathbf{T}$ and $\mathbf{\Phi}$ to zero, that is:

$$\frac{\partial Q_1}{\partial \mathbf{\Phi}} \propto \sum_{k=0}^{N-1} E_N\{[\mathbf{z}(k+1) - \mathbf{\Phi}\mathbf{z}(k) - (\mathbf{I}-\mathbf{\Phi})\mathbf{T}][\mathbf{T} - \mathbf{z}(k)]^T | \mathbf{O}, \overline{\Theta}\} = 0 \tag{17}$$

$$\frac{\partial Q_1}{\partial \mathbf{T}} \propto \sum_{k=0}^{N-1} E_N\{[\mathbf{z}(k+1) - \mathbf{\Phi}\mathbf{z}(k) - (\mathbf{I}-\mathbf{\Phi})\mathbf{T}] | \mathbf{O}, \overline{\Theta}\} = 0 \tag{18}$$

The resulting equations to be solved are nonlinear high-order equations in terms of $\mathbf{\Phi}$ and $\mathbf{T}$:

$$N\mathbf{\Phi}\mathbf{T}\mathbf{T}^T - \mathbf{\Phi}\mathbf{T}\mathbf{A}^T - \mathbf{\Phi}\mathbf{A}\mathbf{T}^T - N\mathbf{T}\mathbf{T}^T + \mathbf{T}\mathbf{A}^T + \mathbf{B}\mathbf{T}^T + \mathbf{\Phi}\mathbf{C} - \mathbf{D} = 0 \tag{19}$$

$$\mathbf{B} - \mathbf{\Phi}\mathbf{A} - N\mathbf{T} + N\mathbf{\Phi}\mathbf{T} = 0 \tag{20}$$

where:

$$\mathbf{A} = \sum_{k=0}^{N-1} E_N[\mathbf{z}(k)|\mathbf{O},\overline{\Theta}], \quad \mathbf{C} = \sum_{k=0}^{N-1} E_N[\mathbf{z}(k)\mathbf{z}(k)^T|\mathbf{O},\overline{\Theta}] \tag{21}$$

$$\mathbf{B} = \sum_{k=0}^{N-1} E_N[\mathbf{z}(k+1)|\mathbf{O},\overline{\Theta}], \quad \mathbf{D} = \sum_{k=0}^{N-1} E_N[\mathbf{z}(k+1)\mathbf{z}(k)^T|\mathbf{O},\overline{\Theta}] \tag{22}$$

are the relevant sufficient statistics that are computed by the EKF smoother during the E-step. By simplifying (19) and (20) we can first form:

$$\widehat{\mathbf{\Phi}} = \mathbf{X}\mathbf{Y}^{-1} \tag{23}$$

where $\widehat{\mathbf{\Phi}}$ is the estimate of the system matrix, and then:

$$\widehat{\mathbf{T}} = \frac{1}{N}(\mathbf{I} - \widehat{\mathbf{\Phi}})^{-1}(\mathbf{B} - \widehat{\mathbf{\Phi}}\mathbf{A}) \tag{24}$$

where $\widehat{\mathbf{T}}$ is the estimate of the target vector.

We now consider the second summation of the Q function in (14) involving the parameter $\mathbf{W}$:

$$Q_2(\mathbf{Z},\mathbf{O},\Theta) = \sum_{k=0}^{N-1} E_N[\mathbf{e}_{k2}^T\mathbf{R}^{-1}\mathbf{e}_{k2}|\mathbf{O},\overline{\Theta}] \tag{25}$$

Minimisation of Q_2, which leads to maximisation of Q, proceeds by setting the partial derivatives with respect to $\mathbf{W}$ to zero, that is:

$$\frac{\partial Q_2}{\partial \mathbf{W}} \propto \sum_{k=0}^{N-1} E_N[\frac{\partial}{\partial \mathbf{W}}\{[\mathbf{o}(k) - h(\mathbf{z}(k)]^2\}|\mathbf{O},\overline{\Theta}] = 0 \tag{26}$$

That is, Q_2 is minimised when the error signal, $\mathbf{e}_{k2} = \mathbf{o}(k) - h(\mathbf{z}(k))$, is minimised. Since the multi-variate mapping function is a feedforward MLP network, then the standard back-propagation is used with $\widehat{\mathbf{z}}(k|N)$ as the input and $\mathbf{o}(k)$ as the desired output to provide estimates of the MLP weights, $\mathbf{W}$.

3.2 Learning a Hidden Dynamic Model Using Variational EM

Model Inference and Learning

For the system described by (10)-(13 inference refers to the calculation of posterior distribution $p(s_{1:N}, \mathbf{x}_{1:N} \mid \mathbf{y}_{1:N})$ given all model parameters, while learning refers to the estimation of model parameters $\Theta = \{\mathbf{A}_{1:S}, \mathbf{a}_{1:S}, \mathbf{B}_{1:S}, \mathbf{C}_{1:S}, \mathbf{c}_{1:S}, \mathbf{D}_{1:S}\}$ given the complete distribution, usually in a maximum likelihood (ML) sense. Under this EM framework, inference is the E step and learning is the M step. In this model, however,

the posterior turns out to be a Gaussian mixture whose number of components is exponential in the number of states (or phones) and frames, and is therefore computationally intractable. Here we develop two approximations, called GMM and HMM posteriors, respectively, based on *variational* techniques. The idea is to choose the approximate posterior $q(s_{1:N}, \mathbf{x}_{1:N} \mid \mathbf{y}_{1:N})$ with a sensible and tractable structure and optimize it by minimizing its Kullback-Liebler (KL) distance to the exact posterior. It turns out that this optimization can be performed efficiently without having to compute the exact (but intractable) posterior.

It is necessary to say a few words about previous approaches and other related work in the literature before presenting the current one. Most of our previous algorithms are developed under the assumption of hard phone boundaries which are either known or estimated separately by some heuristic methods [65], and the intractable exact posterior is approximated by a single Gaussian. This is also true for most of the work in a broad range of literatures for switching state space models. In contrast, the approach presented here uses soft phone assignments that are estimated under a unified EM framework as in [46, 85], but unlike [46, 85], our approximation doesn't factorize s from $\mathbf{x}$ and results in a multimodal posterior over $\mathbf{x}$ instead of a unimodal one, which is justifiably more suitable for speech applications.

The GMM posterior

Under this approximation q is restricted to be:

$$q(s_{1:N}, \mathbf{x}_{1:N}) = \prod_n q(\mathbf{x}_n \mid s_n) q(s_n), \tag{27}$$

where from now on the dependence of the q's on the data $\mathbf{y}$ is omitted but always implied.

Minimizing the KL divergence between q and p is equivalent to maximizing the following functional $\mathcal{F}$,

$$\begin{aligned} \mathcal{F}[q] = \sum_{s_{1:N}} \int d\mathbf{x}_{1:N}\, q(s_{1:N}, \mathbf{x}_{1:N}) \cdot \\ [\log p(\mathbf{y}_{1:N}, \mathbf{x}_{1:N}, s_{1:N}) - \log q(s_{1:N}, \mathbf{x}_{1:N})], \end{aligned} \tag{28}$$

which is also a lower bound of the likelihood function and will be subsequently used as the objective function in the learning (M) step.

By taking *calculus of variation* to optimize $\mathcal{F}$ w.r.t. $q(\mathbf{x}_n|s_n)$ and $q(s_n)$, it turns out that each component $q(\mathbf{x}_n|s_n)$ follows a Gaussian distribution, i.e.,

$$q(\mathbf{x}_n \mid s_n = s) = \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\rho}_{s,n}, \boldsymbol{\Gamma}_{s,n}), \tag{29}$$

and the parameters $\boldsymbol{\rho}_{s,n}$ and $\boldsymbol{\Gamma}_{s,n}$ are given by

$$\boldsymbol{\Gamma}_{s,n} = \mathbf{C}_s^T \mathbf{D}_s \mathbf{C}_s + \mathbf{B}_s + \sum_{s'} \gamma_{s',n+1} \mathbf{A}_{s'}^T \mathbf{B}_{s'} \mathbf{A}_{s'}, \tag{30}$$

$$\begin{aligned}\boldsymbol{\Gamma}_{s,n}\boldsymbol{\rho}_{s,n} &= \mathbf{B}_s(\mathbf{A}_s\sum_{s'}\gamma_{s',n-1}\boldsymbol{\rho}_{s',n-1}+\mathbf{a}_s)\\ &+\sum_{s'}\gamma_{s',n+1}\mathbf{A}_{s'}^T\mathbf{B}_{s'}(\boldsymbol{\rho}_{s',n+1}-\mathbf{a}_{s'})\\ &+\mathbf{C}_s^T\mathbf{D}_s(\mathbf{y}_n-\mathbf{c}_s),\end{aligned} \tag{31}$$

where $\gamma_{s,n}=q(s_n=s)$ and is computed from

$$\begin{aligned}\log\gamma_{s,n} = &\ f_1(\boldsymbol{\rho}_{s,n},\boldsymbol{\Gamma}_{s,n},\Theta)+f_2(\boldsymbol{\rho}_{s',n-1},\boldsymbol{\Gamma}_{s',n-1},\Theta)\\ &+\ f_3(\boldsymbol{\rho}_{s',n+1},\boldsymbol{\Gamma}_{s',n+1},\Theta).\end{aligned} \tag{32}$$

and the f's denote linear functions whose expressions are too lengthy to be written down here. Eq (30) and (31) are coupled linear equations given model parameters Θ and γ's and can be solved efficiently by sparse matrix techniques. Eq (32) is a nonlinear equation by itself and has to be solved by iteration. Eqs (30), (31) and (32) constitute the inference or E step of the algorithm and have to be solved iteratively all together after some proper initializations.

Model learning involves taking derivatives of $\mathcal{F}$ w.r.t. all the model parameters and setting them to zero. This results in a set of linear equations which can be solved easily. Since this step is standard in all EM approaches with no special difficulties, the detailed equations are omitted.

The HMM posterior

Under this approximation q is taken to be

$$q(s_{1:N},\mathbf{x}_{1:N})=\prod_{n=1}^{N}q(\mathbf{x}_n\mid s_n)\cdot\prod_{n=2}^{N}q(s_n\mid s_{n-1})\cdot q(s_1). \tag{33}$$

First we define two posterior transition probabilities:

$$\begin{aligned}\eta_{s's,n} &= q(s_n=s\mid s_{n-1}=s'),\\ \bar{\eta}_{s's,n} &= q(s_n=s\mid s_{n+1}=s')=\frac{\eta_{s's,n+1}\gamma_{s,n}}{\gamma_{s',n+1}},\end{aligned} \tag{34}$$

where γ is the same as in the previous section. It turns out that each $q(\mathbf{x}_n|s_n)$ is again a Gaussian distribution, and $\boldsymbol{\rho}_{s,n}$ and $\boldsymbol{\Gamma}_{s,n}$ are given by coupled linear equations having the same form as (30) and (31), except that the γ's are replaced by η's and $\bar{\eta}$'s. These equations can again be solved by sparse matrix techniques. The γ's and η's themselves can be solved by the following efficient backward-forward procedure given the model parameters and all the $\boldsymbol{\rho}$'s and $\boldsymbol{\Gamma}$'s.

1. Initialize: $z_{s,N+1}=1$ for all s.
2. Backward pass: for $n=N,\cdots,2$

$$z_{s,n} = \sum_{s'} \exp(f_{ss',n}) z_{s',n+1} \ ,$$
$$\eta_{ss',n} = \frac{1}{z_{s,n}} \exp(f_{ss',n}) z_{s',n+1} \ . \tag{35}$$

3. For $n = 1$:

$$z_1 = \sum_{s} \exp(f_{s,1}) z_{s,2} \ ,$$
$$\gamma_{s,1} = \frac{1}{z_1} \exp(f_{s,1}) z_{s,2} \ . \tag{36}$$

4. Forward pass: for $n = 2, \cdots, N$

$$\gamma_{s,n} = \sum_{s'} \eta_{s's,n} \gamma_{s',n-1} \ . \tag{37}$$

Again, f's are functions of the $\boldsymbol{\rho}$'s, $\boldsymbol{\Gamma}$'s and model parameters whose expressions are too lengthy to be given here. Also remember that the complete E step still has to iterate between the calculation of $q(\mathbf{x}_n \mid s_n)$ and $q(s_n \mid s_{n-1})$. The model learning is quite similar to the GMM case and the detailed equations are omitted.

There are a number of important issues to be addressed when using the above algorithms for speech:

1. Hidden dynamics recovery: It is both informative (especially for debugging) and desirable to recover the hidden VTR, and it is calculated by:

$$\hat{x}_n = \sum_{s} \gamma_{s,n} \boldsymbol{\rho}_{s,n} \tag{38}$$

   for both the GMM and HMM posterior assumptions.
2. Recognition strategy: Here we seek the most likely phone sequence given a sequence of observations. For the GMM case, this is simply accomplished by choosing the maximum γ at each frame; while for the HMM posterior we need to perform Viterbi decoding by using γ and η, e.g., the initialization and induction equation for the scoring are:

$$V_1(s) = \gamma_{s,1}, \ V_n(s') = \max_{1 \le s \le S} \left[V_{n-1}(s) \eta_{ss',n} \right] \gamma_{s',n}. \tag{39}$$

   It is highly desirable to incorporate segmental (or minimal duration) constraint and language weighting in the recognition stage and this is implemented by Viterbi decoding with modified transition matrices for both cases (in GMM the transition matrix is created from scratch while in HMM the changes are merged into η). Such a strategy allows the hidden dynamic model to be used in phone recognition directly *without* resorting to an N-best list provided by HMM.

4 Discriminative Deep Neural Networks Aided by Generative Pre-Training

After providing detailed reviews of a range of generative deep-structured dynamic models of speech, we now turn to their discriminative counterpart. Recall that the generative model expressed by (4) and (5) have deep structure, with causal relations from the top discrete linguistic symbols *s* through to hidden dynamic vectors and then to the bottom observed vectors. The reverse direction, from bottom to top, is referred to as the inference step, which is required to perform learning (i.e., training) and for decoding in speech recognition whose goal is to estimate the linguistic symbol sequences. Now we discuss the discriminative version of the deep-structured models, where the direct information flow is opposite: bottom up rather top down. That is, the observed acoustic variables are used to directly compute the intermediate representations, and then to compute the estimate of linguistic labels. It turns out that the deep neural network (DNN) is an excellent candidate for this type of model, as (non-recurrent) neural networks are known to lack the modeling power for explicit speech dynamics.

Historically, the DNN had been very difficult to learn before 2006 [11, 80]. The difficulty was alleviated around 2006 with the work of [52, 53], where a generative pre-training procedure was developed and reported. In this section, we will review this advance and the more recent impact by the DNN on speech recognition research and deployment. We will then analyze the weaknesses of the DNN-based methods, especially those in modeling speech dynamics. This analysis paves a natural path to the recurrent versions of the DNN as well as their connections to and the differences between the generative deep-structured dynamic models of speech reviewed in the preceding two sections.

4.1 Restricted Boltzmann Machines

The generative pre-training procedure first reported in [52, 53] starts with the restricted Boltzmann machine (RBM), which is a special type of Markov random field that has one layer of (typically Bernoulli) stochastic hidden units and one layer of (typically Bernoulli or Gaussian) stochastic visible or observable units.

In an RBM, the joint distribution $p(\mathbf{v},\mathbf{h};\theta)$ over the visible units $\mathbf{v}$ and hidden units $\mathbf{h}$, given the model parameters θ, is defined in terms of an energy function $\mathrm{E}(\mathbf{v},\mathbf{h};\theta)$ of

$$p(\mathbf{v},\mathbf{h};\theta) = \frac{\exp(-\mathrm{E}(\mathbf{v},\mathbf{h};\theta))}{Z}, \tag{40}$$

where $Z = \sum_{\mathbf{v}}\sum_{\mathbf{h}}\exp(-\mathrm{E}(\mathbf{v},\mathbf{h};\theta))$ is a normalization factor, and the marginal distribution that the model assigns to a visible vector $\mathbf{v}$ (we don't care about $\mathbf{h}$ since it is hidden) is

$$p(\mathbf{v};\theta)=\frac{\sum_{\mathbf{h}}\exp(-\mathrm{E}(\mathbf{v},\mathbf{h};\theta))}{Z}. \tag{41}$$

For a Bernoulli (visible)-Bernoulli (hidden) RBM, the energy function is defined as

$$\mathrm{E}(\mathbf{v},\mathbf{h};\theta)=-\sum_{i=1}^{I}\sum_{j=1}^{J}w_{ij}v_ih_j-\sum_{i=1}^{I}b_iv_i-\sum_{j=1}^{J}a_jh_j, \tag{42}$$

where w_{ij} represents the symmetric interaction term between the visible unit v_i and the hidden unit h_j, b_i and a_j are the bias terms, and I and J are the numbers of visible and hidden units. The conditional distributions (for Bernoulli stochastic variables, i.e. binary data) can be efficiently calculated as

$$p(h_j=1|\mathbf{v};\theta)=\sigma\left(\sum_{i=1}^{I}w_{ij}v_i+a_j\right), \tag{43}$$

$$p(v_i=1|\mathbf{h};\theta)=\sigma\left(\sum_{j=1}^{J}w_{ij}h_j+b_i\right), \tag{44}$$

where $\sigma(x)=1/(1+\exp(x))$.

Similarly, for a Gaussian (visible)-Bernoulli (hidden) RBM, the energy is

$$\mathrm{E}(\mathbf{v},\mathbf{h};\theta)=-\sum_{i=1}^{I}\sum_{j=1}^{J}w_{ij}v_ih_j-\frac{1}{2}\sum_{i=1}^{I}(v_i-b_i)^2-\sum_{j=1}^{J}a_jh_j, \tag{45}$$

The corresponding conditional distributions (for Bernoulli or binary $\mathbf{h}$ and Gaussian or continuous-valued $\mathbf{v}$) become

$$p(h_j=1|\mathbf{v};\theta)=\sigma\left(\sum_{i=1}^{I}w_{ij}v_i+a_j\right), \tag{46}$$

$$p(v_i|\mathbf{h};\theta)=\mathcal{N}\left(\sum_{j=1}^{J}w_{ij}h_j+b_i,\ 1\right), \tag{47}$$

where v_i takes real values and follows a Gaussian distribution with mean $\sum_{j=1}^{J}w_{ij}h_j+b_i$ and variance one. Gaussian-Bernoulli RBMs can be used to convert real-valued stochastic variables to binary stochastic variables, which can then be further processed using the Bernoulli-Bernoulli RBMs.

Taking the gradient of the log likelihood $\log p(\mathbf{v};\theta)$ we can derive the update rule for the RBM weights as

$$\triangle w_{ij}\propto E_{data}\left(v_ih_j\right)-E_{model}\left(v_ih_j\right), \tag{48}$$

where $E_{data}\left(v_ih_j\right)$ is the expectation observed in the training set under the distribution defined by the given observations, $p(h|\mathrm{v};\theta)$, and $E_{model}\left(v_ih_j\right)$ is that same

expectation under the distribution defined by the model, $p(\mathbf{v},\mathbf{h};\theta)$. Calculation of $E_{data}\left(v_i h_j\right)$ is facilitated by using $p(h_j=1|\mathbf{v};;\theta)$ to weight samples $v_i h_j$ given observations $\mathbf{v}$. Unfortunately, $E_{model}\left(v_i h_j\right)$ is intractable to compute so the contrastive divergence (CD) approximation to the gradient is used where $E_{model}\left(v_i h_j\right)$ is replaced by running the Gibbs sampler initialized at the data for one full step. The steps in approximating $E_{model}\left(v_i h_j\right)$ is as follows:

1. Initialize $\mathbf{v_0}$ at data
2. Sample $\mathbf{h_0} \sim p(\mathbf{h}|\mathbf{v_0})$
3. Sample $\mathbf{v_1} \sim p(\mathbf{v}|\mathbf{h_0})$
4. Sample $\mathbf{h_1} \sim p(\mathbf{h}|\mathbf{v_1})$

Then the $(\mathbf{v_1}, \mathbf{h_1})$ is a sample from the model, acting as a very rough estimate of $E_{model}\left(v_i h_j\right)$. Use of $(\mathbf{v_1},\mathbf{h_1})$ to approximate $E_{model}\left(v_i h_j\right)$ gives rise to the algorithm of CD-1. The sampling process is pictorially depicted in Fig. 2 where $< v_i h_j >^k \equiv (\mathbf{v}_k,\mathbf{h}_k)$.

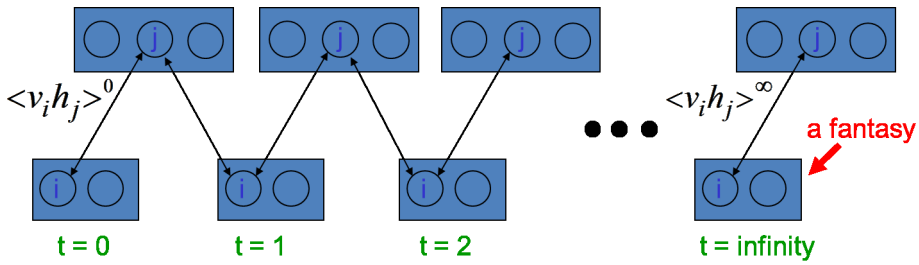

Fig. 2 A pictorial view of sampling from an RBM during RBM learning (courtesy of Geoff Hinton)

Careful training of RBMs is essential to the success of applying the RBM and related deep learning techniques to solve practical problems. See the technical report [50] for a very useful practical guide for training RBMs.

The RBM discussed above is a generative model, which characterizes the input data distribution using hidden variables and there is no label information involved. However, when the label information is available, it can be used together with the data to form the joint "data" set. Then the same CD learning can be applied to optimize the approximate "generative" objective function related to data likelihood. Further, and more interestingly, a "discriminative" objective function can be defined in terms of the conditional likelihood of labels. This discriminative RBM can be used to "fine tune" an RBM for classification tasks [60].

4.2 Stacking up RBMs to form a DBN

Stacking a number of the RBMs learned layer by layer from bottom up gives rise to a deep belief network (DBN), an example of which is shown in Fig. 3. The stacking procedure is as follows. After learning a Gaussian-Bernoulli RBM (for applications with continuous features such as speech) or Bernoulli-Bernoulli RBM (for applications with nominal or binary features such as a black-white image or coded text), we treat the activation probabilities of its hidden units as the data for training the Bernoulli-Bernoulli RBM one layer up. The activation probabilities of the second-layer Bernoulli-Bernoulli RBM are then used as the visible data input for the third-layer Bernoulli-Bernoulli RBM, and so on. Mathematically for a DBN with M layers we can model the joint distribution between the observations $\mathbf{v}$ and the L hidden layers $\{\mathbf{h}^k : k = 1,2,\ldots M\}$ as follows

$$p\left(\mathbf{v},\mathbf{h}^1,\ldots,\mathbf{h}^M\right) = p\left(\mathbf{v} \mid \mathbf{h}^1\right)\left(\prod_{k=1}^{M-2} p(\mathbf{h}^k|\mathbf{h}^{k+1})\right) p(\mathbf{h}^{M-1},\mathbf{h}^M) \tag{49}$$

This allows us to derive relevant distributions, e.g. the posterior distribution $p(\mathbf{h}^M|\mathbf{v})$. Some theoretical justification of this efficient layer-by-layer greedy learning strategy is given in [52], where it is shown that the *stacking* procedure above improves a variational lower bound on the likelihood of the training data under the composite model. That is, the greedy procedure above achieves approximate maximum likelihood learning. Note that this learning procedure is unsupervised and requires no class label.

When applied to classification tasks, the generative pre-training can be followed by or combined with other, typically discriminative, learning procedures that fine-tune all of the weights jointly to improve the performance of the network. This discriminative fine-tuning is performed by adding a final layer of variables that represent the desired outputs or labels provided in the training data. Then, the back-propagation algorithm can be used to adjust or fine-tune the network weights in the same way as for the standard feed-forward neural network. When used in this way we refer to this as the deterministic neural network or DNN. What goes to the top, label layer of this DNN depends on the application. For speech recognition applications, the top layer, denoted by $\mathbf{h}^M = \{l_1, l_2, \ldots, l_j, \ldots, l_L\}$, in Fig. 3 , can represent either syllables, phones, sub-phones, phone states, or other speech units used in the HMM-based speech recognition system.

The generative pre-training described above has produced better phone and speech recognition results than random initialization on a wide variety of tasks. Further research has also shown the effectiveness of other pre-training strategies. As an example, greedy layer-by-layer training may be carried out with an additional discriminative term to the generative cost function at each level. And without generative pre-training, purely discriminative training of DNNs from random initial weights using the traditional stochastic gradient decent method has been shown to work very well when the scales of the initial weights are set carefully and the mini-batch sizes, which trade off noisy gradients with convergence speed, used in

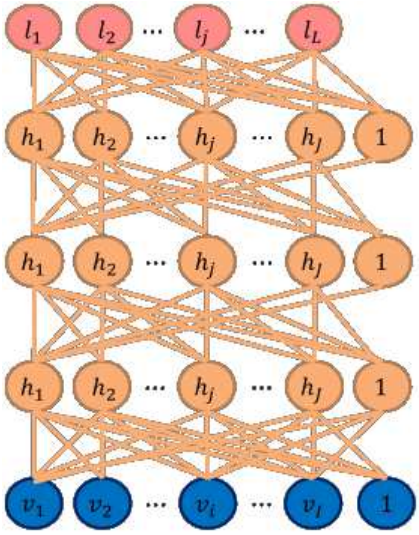

Fig. 3 An illustration of the DBN/DNN architecture

stochastic gradient decent are adapted prudently (e.g., with an increasing size over training epochs). Also, randomization order in creating mini-batches needs to be judiciously determined. Importantly, it was found effective to learn a DNN by starting with a shallow neural net with a single hidden layer. Once this has been trained discriminatively (using early stops to avoid overfitting), a second hidden layer is inserted between the first hidden layer and the labeled softmax output units and the expanded deeper network is again trained discriminatively. This can be continued until the desired number of hidden layers is reached, after which a full backpropagation "fine tuning" is applied. This discriminative "pre-training" procedure is found to work well in practice (e.g., [94, 107]).

Despite the great success in using DNNs for large vocabulary speech recognition, training is still quite slow due to the large number of parameters and the required large data set sizes. Part of current research has now begun to focus on optimization techniques to improve the training regime for DNNs [93] specifically and for speech and language processing as a whole [104].

4.3 Interfacing the DNN with an HMM to Incorporate Sequential Dynamics

The DNN discussed above is a static classifier with input vectors having a fixed dimensionality. However, many practical pattern recognition and information processing problems, including speech recognition, machine translation, natural lan-

guage understanding, video processing and bio-information processing, require sequence recognition. In sequence recognition, sometimes called classification with structured input/output, the dimensionality of both inputs and outputs are variable.

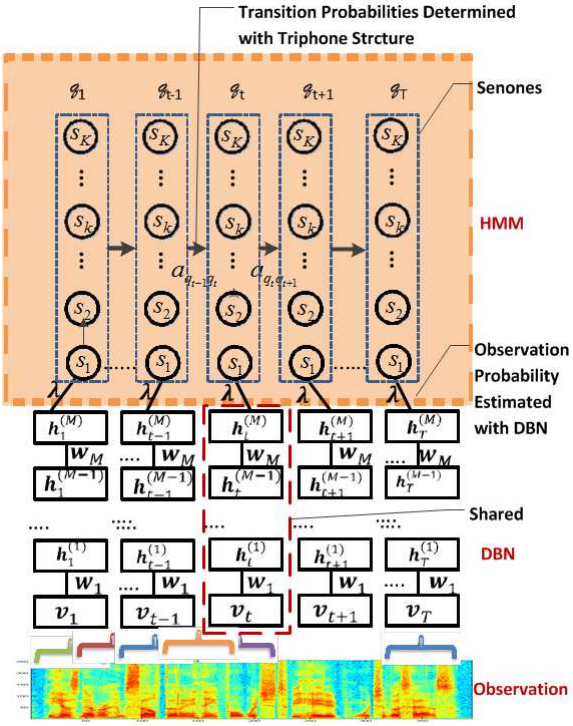

Fig. 4 Interface between DBN/DNN and HMM to form a DBN-HMM or DNN-HMM.

The HMM, based on dynamic programming operations, is a convenient tool to help port the strength of a static classifier to handle dynamic or sequential patterns. Thus, it is natural to combine the DNN and HMM to bridge the gap between static and sequence pattern recognition. A popular architecture to fulfil this is shown in Fig. 4. This architecture has been successfully used in speech recognition experiments from small to mid and to large scales, as reported in [15, 16, 51, 59, 64, 77, 79, 93, 94, 108–110]. The excellent recognition accuracy obtained by the DNN-HMM and its scalability from small to large tasks have resulted in wide industry adoption of this architecture and a huge surge of research efforts. This is so despite the recognition of the weaknesses of modeling realistic speech dynamics via the HMM and via the windowed speech frames as inputs to the DNN.

It is important to note that the unique elasticity of the temporal dynamic of speech as elaborated in [39, 40] would require temporally-correlated models more powerful than the HMM for the ultimate success of speech recognition. Integrating such

dynamic models having realistic co-articulatory properties with the DNN and possibly other deep learning models to form the coherent dynamic deep architecture is a challenging new research. Adding recurrent connections over the hidden neurons gives one reasonable way of incorporating speech dynamics into the model, at least more principled than using a long window of frames in the DNN-HMM architecture. In the next section we turn to a review and analysis of the recurrent neural network (RNN) before providing connections to the generative deep-structured dynamic speech models reviewed earlier.

5 Recurrent Neural Networks for Discriminative Modeling of Speech Dynamics

The use of RNNs or related neural predictive models for speech recognition dates back to early 1990's (e.g., [27, 90]), with relatively low accuracy and whose results could not be reproduced by other groups until recently. Since deep learning became popular in recent years, much more research has been devoted to the RNN (e.g., [48, 69–71, 73–76, 84, 98–100, 103] and its stacked versions, also called deep RNNs [49]. Most work on RNNs made use of the method of Back Propagation Through Time (BPTT) to train the RNNs, and empirical tricks need to be exploited (e.g., truncate gradients when they become too large [74]) in order to make the training effective. It is not until recently that careful analysis was made to fully understand the source of difficulties in learning RNNs and somewhat more principled, but still rather heuristic, solutions were developed. For example, in [7, 84], a heuristic strategy of gradient norm clipping was proposed to deal with the gradient exploding problem during BPTT training. There are other solutions offered to improve the learning method for the RNN (e.g., [28, 56])

5.1 RNNs Expressed in the State-Space Formalism

Let us formulate the RNN in terms of the nonlinear state space model commonly used in signal processing. We will compare it with the same state space formulation of nonlinear dynamic systems used as generative models for speech acoustics. The contrast between the discriminative RNN and the use of the same mathematical model in the generative mode allows us to shed light onto why one approach works better than another and how a combination of the two is desirable.

As shown in Fig. 5 given an input sequence $\mathbf{X} = (\mathbf{x}_1, \cdots, \mathbf{x}_t, \cdots, \mathbf{x}_T)$, the RNN computes the noise free hidden state dynamic vector sequence $\mathbf{H} = (\mathbf{h}_1, \cdots, \mathbf{h}_t, \cdots, \mathbf{h}_T)$ by iterating the following from $t = 1$ to T:

$$\mathbf{h}_t = f(\mathbf{W}_{xh}\mathbf{x}_t + \mathbf{W}_{hh}\mathbf{h}_{t-1}) = f(\mathbf{u}_t) \tag{50}$$

$$\mathbf{y}_t = g(\mathbf{W}_{hy}\mathbf{h}_t) = g(\mathbf{v}_t) \tag{51}$$

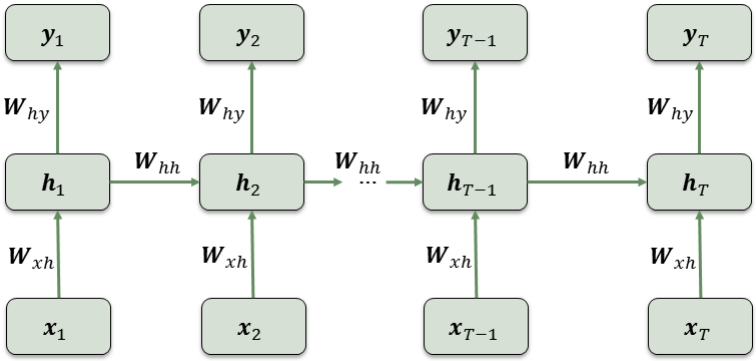

Fig. 5 Information flow in the standard recurrent neural network from observation variables to the target labels as output variables via the hidden-state vectors

where $\mathbf{Y} = (\mathbf{y}_1, \cdots, \mathbf{y}_t, \cdots, \mathbf{y}_T)$ is the "target label" output sequence, which is the "observation" sequence in the standard state-space formulation.

The desired target signal in the above state-space model is the predicted "label" or target vector, $\mathbf{l}_t$, a vector of one-hot coded class labels. Define the error function as the sum of squared differences between $\mathbf{y}_t$ and $\mathbf{l}_t$ over time, or the cross entropy between them. Then BPTT unfolds the RNN over time in computing the gradients with respect to $\mathbf{W}_{hy}, \mathbf{W}_{xh}$ and $\mathbf{W}_{hh}$, and stochastic gradient descent is applied to update these weight matrices.

5.2 The BPTT learning algorithm

The BPTT [10,56] is an extension of the classic feedforward backpropagation where the stacked hidden layers for the same training epoch, t, are replaced by unfolding the recurrent neural network in time and stacking T single hidden layers across time, $t = 1, 2, \ldots, T$. Referring to Fig. 5 and (50),(51) let us assume a recurrent neural network with K inputs, N internal hidden units, and L outputs, and define the following variables at time layer t:

- $\mathbf{x}_t$ is the $K \times 1$ vector of inputs, $\mathbf{h}_t$ is the $N \times 1$ vector of hidden unit outputs, $\mathbf{y}_t$ is the $L \times 1$ vector of outputs, and $\mathbf{l}_t$ is the $L \times 1$ vector of training output targets, where the jth vector element, e.g., $h_t(j)$ is the jth hidden unit for $j = 1, 2, \ldots, N$;
- $\mathbf{W}_{hy}$ is the $L \times N$ matrix of weights connecting the N hidden units to the L outputs, $\mathbf{W}_{xh}$ is the $N \times K$ matrix of weights connecting the K inputs to the N hidden units, and $\mathbf{W}_{hh}$ is the $N \times N$ matrix of weights connecting the N hidden units from layer $t-1$ to layer t, where the (i, j)th matrix element, e.g., $w_{hy}(i, j)$ is the weight connecting the jth hidden unit to the ith output unit for $i = 1, 2, \ldots, L$ and $j = 1, 2, \ldots, N$;

- $\mathbf{u}_t = \mathbf{W}_{xh}\mathbf{x}_t + \mathbf{W}_{hh}\mathbf{h}_{t-1}$ is the $N \times 1$ vector of hidden unit input potentials, $\mathbf{v}_t = \mathbf{W}_{hy}\mathbf{h}_t$ is the $L \times 1$ vector of output unit input potentials, from which we have $\mathbf{h}_t = f(\mathbf{u}_t)$ and $\mathbf{y}_t = g(\mathbf{v}_t)$; where
- $f(\mathbf{u}_t)$ is the hidden layer activation function ($f^{'}(\mathbf{u}_t)$ is its derivative), and $g(\mathbf{v}_t)$ is the output layer activation function ($g^{'}(\mathbf{v}_t)$ is its derivative).

Similar to classic backpropagation we begin by defining the summed squared error between the actual output, $\mathbf{y}_t$, and the target vector, $\mathbf{l}_t$, averaged across all time epochs:

$$E = c\sum_{t=1}^{T}\|\mathbf{l}_t - \mathbf{y}_t\|^2 = c\sum_{t=1}^{T}\sum_{j=1}^{L}(l_t(j) - y_t(j))^2 \tag{52}$$

where c is a conveniently chosen scale factor and seek to minimise this error w.r.t to the weights using a gradient descent. For a specific weight, w, the update rule for gradient descent is:

$$w^{new} = w - \gamma\frac{\partial E}{\partial w} \tag{53}$$

To do this we define the so-called error propagation term which is the error gradient w.r.t to the unit input potential:

$$\delta_t^y(j) = -\frac{\partial E}{\partial v_t(j)},\ \ \delta_t^h(j) = -\frac{\partial E}{\partial u_t(j)} \tag{54}$$

choose $c = 0.5$ and then use the chain rule (keeping track of the dependencies) as follows:

1. For $t = 1,2,\ldots,T$ compute the input potentials $(\mathbf{u}_t, \mathbf{v}_t)$ and activation outputs $(\mathbf{h}_t, \mathbf{y}_t)$ given the current RNN weights and input $\mathbf{x}_t$ (the forward pass).
2. At time layer $t = T$ calculate the error propagation term (where $\odot$ is the component-wise multiplication operator):

$$\begin{aligned}\delta_T^y(j) &= -\frac{\partial E}{\partial y_T(j)}\frac{\partial y_T(j)}{\partial v_T(j)} = (l_T(j) - y_T(j))g^{'}(v_T(j)) \quad \text{for } j = 1,2,\ldots,L\\ \boldsymbol{\delta}_T^y &= (\mathbf{l}_T - \mathbf{y}_T)\odot g^{'}(\mathbf{v}_T)\end{aligned} \tag{55}$$

   at the output units and

$$\begin{aligned}\delta_T^h(j) &= -\left(\sum_{i=1}^{L}\frac{\partial E}{\partial v_T(i)}\frac{\partial v_T(i)}{\partial h_T(j)}\frac{\partial h_T(j)}{\partial u_T(j)}\right) = \sum_{i=1}^{L}\delta_T^y(i)w_{hy}(i,j)f^{'}(u_T(j)) \quad \text{for } j = 1,2,\ldots,N\\ \boldsymbol{\delta}_T^h &= \mathbf{W}_{hy}^T\boldsymbol{\delta}_T^y \odot f^{'}(\mathbf{u}_T)\end{aligned} \tag{56}$$

   for the internal units (where $\boldsymbol{\delta}_T^y$ is propagated back from the output layer T).
3. At the earlier layers, $t = T-1, T-2, \ldots, 1$, calculate the error propagation term:

$$\delta_t^y(j) = (l_t(j) - y_t(j))g^{'}(v_t(j)) \quad \text{for } j = 1,2,\ldots,L$$
$$\boldsymbol{\delta}_t^y = (\mathbf{l}_t - \mathbf{y}_t) \odot g^{'}(\mathbf{v}_t) \tag{57}$$

for the output units and

$$\begin{aligned}\delta_t^h(j) &= -\left[\sum_{i=1}^{N} \frac{\partial E}{\partial u_{t+1}(i)} \frac{\partial u_{t+1}(i)}{\partial h_t(j)} + \sum_{i=1}^{L} \frac{\partial E}{\partial v_t(i)} \frac{\partial v_t(i)}{\partial h_t(j)}\right] \frac{\partial h_t(j)}{\partial u_t(j)} \\ &= \left[\sum_{i=1}^{N} \delta_{t+1}^h(i) w_{hh}(i,j) + \sum_{i=1}^{L} \delta_t^y(i) w_{hy}(i,j)\right] f^{'}(u_t(j)) \quad \text{for } j = 1,2,\ldots,N \\ \boldsymbol{\delta}_t^h &= \left[\mathbf{W}_{hh}^T \boldsymbol{\delta}_{t+1}^h + \mathbf{W}_{hy}^T \boldsymbol{\delta}_t^y\right] \odot f^{'}(\mathbf{u}_t)\end{aligned} \tag{58}$$

for the internal units (where $\boldsymbol{\delta}_t^y$ is propagated back from the output layer$t$, and $\boldsymbol{\delta}_{t+1}^h$ is propagated back from hidden layer $t+1$).

Then we adjust the weights as follows:

1. For the jth hidden to ith output layer weights at layer t:

$$\begin{aligned} w_{hy}^{new}(i,j) &= w_{hy}(i,j) - \gamma \sum_{t=1}^{T} \frac{\partial E}{\partial v_t(i)} \frac{\partial v_t(i)}{\partial w_{hy}(i,j)} = w_{hy}(i,j) - \gamma \sum_{t=1}^{T} \delta_t^y(i) h_t(j) \\ \mathbf{W}_{hy}^{new} &= \mathbf{W}_{hy} + \gamma \sum_{t=1}^{T} \boldsymbol{\delta}_y^t \mathbf{h}_t^T \end{aligned} \tag{59}$$

2. For the jth input to the ith hidden layer weights at layer t:

$$\begin{aligned} w_{xh}^{new}(i,j) &= w_{xh}(i,j) - \gamma \sum_{t=1}^{T} \frac{\partial E}{\partial u_t(i)} \frac{\partial u_t(i)}{\partial w_{xh}(i,j)} = w_{xh}(i,j) - \gamma \sum_{t=1}^{T} \delta_t^h(i) x_t(j) \\ \mathbf{W}_{xh}^{new} &= \mathbf{W}_{xh} + \gamma \sum_{t=1}^{T} \boldsymbol{\delta}_h^t \mathbf{x}_t^T \end{aligned} \tag{60}$$

3. For the jth hidden at layer $t+1$ to the ith hidden at layer t weights:

$$\begin{aligned} w_{hh}^{new}(i,j) &= w_{hh}(i,j) - \gamma \sum_{t=1}^{T} \frac{\partial E}{\partial u_t(i)} \frac{\partial u_t(i)}{\partial w_{hh}(i,j)} = w_{xhh}(i,j) - \gamma \sum_{t=1}^{T} \delta_t^h(i) h_{t-1}(j) \\ \mathbf{W}_{hh}^{new} &= \mathbf{W}_{hh} + \gamma \sum_{t=1}^{T} \boldsymbol{\delta}_h^t \mathbf{h}_{t-1}^T \end{aligned} \tag{61}$$

where γ is the learning rate.

One drawback of the BPTT is that the entire time series is needed to perform one update of the weights, thereby making BPTT a "batch" adaptation algorithm. It is possible to consider an online adaptation if one truncates the past history to no more than the last p time epochs, creating the BPTT(p) or p-BPTT variant.

The computational complexity of the BPTT is given as $O(M^2)$ per time step where $M = LN + NK + N^2$ is the number of internal units. As with classic feedforward backpropagation slow convergence can be expected with several thousand epochs needed. However unlike feedforward backpropagation the BPTT is not guaranteed to converge to a local minimum and it is far from trivial to achieve good results with much experimentation and tuning. This is mainly due to the problem of exploding and vanishing gradients as described in [84].

5.3 The EKF learning algorithm

In section 2.2 the extended Kalman filter (EKF) was used to provide estimates of the hidden state variable in the non-linear state-space system described by (4) and (5). By reformulating the state-space system such that the hidden state variable are the RNN weights and the system observations are the target vectors we can use the EKF as a learning algorithm for the RNN. First popularised in the hallmark work of [87] we proceed by restacking the $L \times N$ $\mathbf{W}_{hy}$, $N \times K$ $\mathbf{W}_{xh}$, and $N \times N$ $\mathbf{W}_{hh}$ RNN weights into a single, state vector $\mathbf{w}$ of size $LN + NK + N^2$. Then we form the following state-space system:

$$\begin{aligned} \mathbf{w}(n+1) &= \mathbf{w}(n) + \mathbf{q}(n) \\ \mathbf{l}(n) &= \mathbf{h}_n(\mathbf{w}(n), \mathbf{x}_{1:n}) \end{aligned} \tag{62}$$

where $\mathbf{l}(n) \equiv \mathbf{l}_n$ is the target vector and the desired "observation" from the system at time n, $\mathbf{q}(n)$ is the external input to the system considered as an uncorrelated Gaussian white noise process, $\mathbf{w}(n)$ are the RNN weights at time n, and $\mathbf{y}_n \equiv \mathbf{h}_n(\hat{\mathbf{w}}(n), \mathbf{x}_{1:n})$ is the time-dependent RNN output observation function at time-step n derived from the current RNN weight estimates $\hat{\mathbf{w}}(n)$ and the input vector sequence $\mathbf{x}_{1:n} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$. The EKF recursion applied to this system will estimate the unknown hidden state $\mathbf{w}(n)$, given the "observations" $\mathbf{l}(n)$, by attempting to minimise the innovation error $\boldsymbol{\xi}(n) = (\mathbf{l}(n) - \mathbf{h}_n(\hat{\mathbf{w}}(n), \mathbf{x}_{1:n})) = (\mathbf{l}_n - \mathbf{y}_n)$ in the minimum mean square error (MMSE) sense equivalent to the minimisation of the BPTT squared error of (52). The EKF recursion for this system simplifies to:

$$\begin{aligned} \mathbf{K}(n) &= \mathbf{P}(n)\mathbf{H}(n)[\mathbf{H}(n)^T\mathbf{P}(n)\mathbf{H}(n)]^{-1} \\ \hat{\mathbf{w}}(n+1) &= \hat{\mathbf{w}}(n) + \mathbf{K}(n)\boldsymbol{\xi}(n) \\ \mathbf{P}(n+1) &= \mathbf{P}(n) - \mathbf{K}(n)\mathbf{H}(n)^T\mathbf{P}(n) + \mathbf{Q}(n) \end{aligned} \tag{63}$$

where $\mathbf{K}(n)$ is the Kalman gain, $\mathbf{P}(n) = E[(\mathbf{w}(n) - \hat{\mathbf{w}}(n))(\mathbf{w}(n) - \hat{\mathbf{w}}(n))^T$ is the state error covariance and $\mathbf{H}(n) = \left.\frac{\partial \mathbf{h}_n(\mathbf{w}(n)), \mathbf{x}_{1:n})}{\partial \mathbf{w}}\right|_{\mathbf{w}=\hat{\mathbf{w}}(n)}$ is the Jacobian of partial derivatives of the the RNN output with respect to the weights. The EKF recursion requires the initial estimates $\hat{\mathbf{w}}(0)$ and $\mathbf{P}(0)$ and a model for the process noise $\mathbf{Q}(n)$. Typically

$\hat{\mathbf{w}}(0)$ is generated randomly, $\mathbf{P}(0)$ is set to a diagonal matrix with a large diagonal component and $\mathbf{Q}(n)$ is a diagonal matrix with small diagonal variance terms.

Although the EKF recursion is a very elegant approach exploiting the theory of optimum Kalman filters, the Jacobian linearisation of the non-linear h_n only guarantees convergence to a local minimum. Furthermore the calculation of the Jacobian at each iteration time step requires either direct calculation of the gradients, which is computationally expensive, or the use of an offline run of the BPTT for the gradients by backpropagation. The BPTT-EKF executes a reformulated BPTT(p) over the input data sequence $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$, where the RNN weights $\mathbf{w} = \hat{\mathbf{w}}(n)$, to calculate the gradient $\frac{\partial \mathbf{y}_n}{\partial \mathbf{w}}$ for the $\mathbf{H}(n)$. This is followed by one iteration of the EKF recursion to calculate $\hat{\mathbf{w}}(n+1)$ and so on. The BPTT-EKF exhibits an order $O(LM^2)$ computational complexity per time-step where $M = LN + NK + N^2$ is the number of internal units and has been shown to exhibit superior convergence over BPTT and can be considered one of the classic state of the art approaches to RNN training.

6 Comparing Two Types of Dynamic Models

We are now in a position to discuss similarities of and differences between the two types of deep and dynamic models: 1) the generative deep-structured dynamic model, which we reviewed in Section 2, and 2) the discriminative RNN, which we reviewed in Section 5. The "deepness" of the models is expressed in terms of the time steps. Several key aspects are compared below, one in each subsection.

6.1 Top-Down versus Bottom-Up

Top-down modeling here refers to the hierarchical way in which the speech data are modeled by the generative hidden dynamics. The modeling process starts with specification of the linguistic label sequence at the top level. Then the label sequence generates the hidden dynamic vector sequence, which in turn generates the acoustic observation sequence at the bottom level in the hierarchy. This way of modeling can be viewed as fitting the observation data. On the other hand, in bottom-up modeling based on the RNN, the information flow starts at the bottom level of acoustic observation, which activates the hidden layer or vector dynamics in the RNN. Then the output layer of the RNN computes the linguistic label or target sequence at the top level of the hierarchy. Since the top layer determines the speech-class distinction, the bottom-up modeling approach can also be called discriminative learning. We elaborate on the top-down verses bottom-up comparisons below.

6.1.1 Top-down generative hidden dynamic modeling

To facilitate the comparison, we use a general form of the generative hidden dynamic model following the discussion in Section III.A of [33] with slight modification, and note that speech recognition researchers have used many variants of this form to build speech recognizers in the past; see a survey in Sections III.D and III.E of [33] and the review in Section 2. In the discussion provided in this section, the general form of the state and observation equations in the generative hidden dynamic model takes the form of

$$\mathbf{h}_t = G(\mathbf{h}_{t-1}; \mathbf{W}_{l_t}, \mathbf{\Lambda}_{l_t}) + \text{StateNoise} \tag{64}$$

$$\mathbf{x}_t = H(\mathbf{h}_t, \mathbf{\Omega}_{l_t}) + \text{ObsNoise} \tag{65}$$

Here, $\mathbf{W}_{l_t}$ is the system matrix that shapes the (articulatory-like) state dynamics, and $\mathbf{\Lambda}_{l_t}$ serves as the "input" driving force to the state dynamics. Both of them are dependent on the label l_t at time t with segmental properties, hence the model is also called a (segmental) switching dynamic system. The system matrix is analogous to $\mathbf{W}_{hh}$ in the RNN. $\mathbf{\Omega}_{l_t}$ is the parameter set that governs the nonlinear mapping from the hidden (articulatory-like) states in speech production to acoustic features of speech. In one implementation, $\mathbf{\Omega}_{l_t}$ took the form of shallow MLP weights [35, 86, 101]. In another implementation, $\mathbf{\Omega}_{l_t}$ took the form of a set of matrices in a mixture of linear experts [67].

The state equation in various previous implementations of the hidden dynamic models of speech does not take nonlinear forms. Rather, the following linear form was used (e.g., [35]):

$$\mathbf{h}_t = \mathbf{W}_{hh}(l_t)\mathbf{h}_{t-1} + [\mathbf{I} - \mathbf{W}_{hh}(l_t)]\mathbf{t}_{l_t} + \text{StateNoise} \tag{66}$$

which exhibits the target-directed property for the articulatory-like dynamics. Here, the parameters $\mathbf{W}_{hh}$ are a function of the (phonetic) label l_t at a particular time t, and $\mathbf{t}_{l_t}$ is a mapping from the symbolic quantity l_t of a linguistic unit to the continuous-valued "target" vector with the segmental property. To make the following comparisons easy, let's keep the nonlinear form and remove both the state and observation noise, yielding the state-space generative model of

$$\mathbf{h}_t = G(\mathbf{h}_{t-1}; \mathbf{W}_{l_t}, \mathbf{t}_{l_t}) \tag{67}$$

$$\mathbf{x}_t = H(\mathbf{h}_t, \mathbf{\Omega}_{l_t}) \tag{68}$$

6.1.2 Bottom-up discriminative recurrent neural networks and the "generative" counterpart

Let us rewrite (50) and (51) into a more general form:

$$\mathbf{h}_t = F(\mathbf{h}_{t-1}, \mathbf{x}_t; \mathbf{W}_{hh}, \mathbf{W}_{xh}) \tag{69}$$
$$\mathbf{y}_t = K(\mathbf{h}_t; \mathbf{W}_{hy}). \tag{70}$$

where information flow goes from observation data $\mathbf{x}_t$ to hidden vectors $\mathbf{h}_t$ and then to the predicted target label vectors $\mathbf{y}_t$ in the bottom-up direction.

Compared with (67) and (68), which describe the information flow from the top-level label-indexed phonetic "target" vector $\mathbf{t}_{l_t}$ to hidden vectors $\mathbf{h}_t$ and then to observation data $\mathbf{x}_t$, we clearly see opposite information flows.

In order to examine other differences between the two types of models in addition to the top-down versus bottom-up difference, we keep the same mathematical description of the RNN but swap the variables of input $\mathbf{x}_t$ and output $\mathbf{y}_t$ in (69) and (70). This yields

$$\mathbf{h}_t = F_1(\mathbf{h}_{t-1}, \mathbf{y}_t; \mathbf{W}_{hh}, \mathbf{W}_{yh}) \tag{71}$$
$$\mathbf{x}_t = K_1(\mathbf{h}_t; \mathbf{W}_{hx}). \tag{72}$$

or more specifically

$$\mathbf{h}_t = f_1(\mathbf{W}_{hh}\mathbf{h}_{t-1} + \mathbf{W}_{yh}\mathbf{y}_t) \tag{73}$$
$$\mathbf{x}_t = g_1(\mathbf{W}_{hx}\mathbf{h}_t) \tag{74}$$

The "generative" version of the RNN can be illustrated by Figure 6, which is the same as the normal "discriminative" version of the RNN shown in Figure 5 except all arrows change their directions.

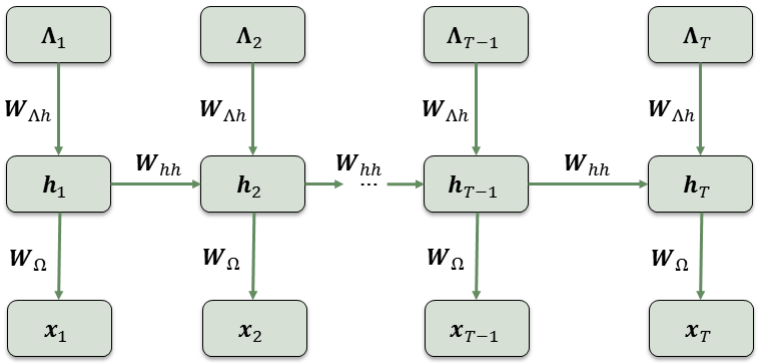

Fig. 6 Information flow in the same recurrent neural network of Figure 5 except we swap the observation variables with the output variables without changing the mathematical form of the state-space model.

Given the "generative" form of the two types of the deep, dynamic models, one (the hidden dynamic model) described by (67) and (68), and the other (the RNN) by (71) and (71), we discuss below the contrast between them with respect to the

differrent nature of the hidden-space representations (while keeping the same generative form of the models). We will also discuss below other aspects of the contrast between them including different ways of exploiting model parameters.

6.2 Localist versus distributed representations

Localist and distributed representations are important concepts in cognitive science as two distinct styles of data representation. In the localist representation, each neuron represents a single concept on a stand-alone basis. That is, localist units have their own meaning and interpretation, not so for the units in distributed representation. The latter pertains to an internal representation of concepts in such a way that they are modeled as being explained by the interactions of many hidden factors. A particular factor learned from configurations of other factors can often generalize well to new configurations, not so in localist representation.

Distributed representations, based on vectors consisting of many elements or units, naturally occur in a "connectionist" neural network, where a concept is represented by a pattern of activity across a number of many units and where at the same time a unit typically contributes to many concepts. One key advantage of such many-to-many correspondence is that they provide robustness in representing the internal structure of the data in terms of graceful degradation and damage resistance. Such robustness is enabled by redundant storage of information. Another advantage is that they facilitate automatic generalization of concepts and relations, thus enabling reasoning abilities. Further, distributed representation allows similar vectors to be associated with similar concepts and it allows efficient use of representational resources. These attractive properties of distributed representations, however, come with a set of weaknesses. These include non-obviousness in interpreting the representations, difficulties with representing hierarchical structure, and inconvenience in representing variable-length sequences. Distributed representations are also not directly suitable for input and output to a network and some translation with localist representations are needed.

On the other hand, local representation has advantages of explicitness and ease of use — the explicit representation of the components of a task is simple and the design of representational schemes for structured objects is easy. But the weaknesses are many, including inefficiency for large sets of objects, highly redundant use of connections, and undesirable growth of units in networks which represent complex structure.

All versions of the hidden dynamic models for deep speech structure [12, 19, 34, 37, 86, 101] adopt the "localist" representation of the symbolic linguistic units, and the RNN makes use of the distributed representation. This can be seen directly from (67) for the hidden dynamic model and from (71) for the RNN (in the "generative" version). In the former, symbolic linguistic units l_t as a function of time t are coded implicitly in a stand-alone fashion. The connection of symbolic linguistic units to continuous-valued vectors is made via a one-to-one mapping, denoted by $\mathbf{t}_{l_t}$

in (67), to the hidden dynamic's asymptotic "targets" denoted by vector **t**. This type of mapping is common in phonetic-oriented phonology literature, and is called the "interface between phonology and phonetics" in a functional computational model of speech production in [19]. Further, the hidden dynamic model uses the linguistic labels represented in a localist manner to index separate sets of time-varying parameters $\mathbf{W}_{l_t}$ and $\mathbf{\Omega}_{l_t}$, leading to "switching" dynamics which considerably complicates the decoding computation. This kind of parameter specification isolates the parameter interactions across different linguistic labels, gaining the advantage of explicit interpretation of the model but losing on direct discrimination across linguistic labels.

In contrast, in the state equation of the RNN model shown in (71), the symbolic linguistic units are directly represented as one-hot vectors of $\mathbf{y}_t$ as a function of time t. No mapping to separate continuous-valued "phonetic" vectors are needed. While the one-hot coding of $\mathbf{y}_t$ vectors is localist, the hidden state vector **h** provides a distributed representation and thus allows the model to store a lot of information about the past in a highly efficient manner. Importantly, there is no longer a notion of label-specific parameter sets of $\mathbf{W}_{l_t}$ and $\mathbf{\Omega}_{l_t}$ as in the hidden dynamic model. The weight parameters in the RNN are shared across all linguistic label classes. This enables direct discriminative learning for the RNN. In addition, the distributed representation used by the hidden layer of the RNN allows efficient and redundant storage of information, and has the capacity to automatically disentangle variation factors embedded in the data. However, as inherent in distributed representations discussed earlier, the RNN also carries with them the difficulty of interpreting the parameters and hidden states, and the difficulty of modeling structure.

6.3 Latent Explanatory Variables versus End-to-End Discriminative Learning

An obvious strength of the localist representation as adopted by the hidden dynamic models for deep speech structure is that the model parameters and the latent (i.e. hidden) state variables are explainable and easy to diagnose. In fact, one main motivation of many of such models is that the knowledge of hierarchical structure in speech production in terms of articulatory and vocal tract resonance dynamics can be directly (but approximately with a clear sense of the degree of approximation) incorporated into the design of the models [12, 19, 20, 22, 31, 37, 66, 68, 83, 102, 106]. Practical benefits of using interpretable, localist representation of hidden state vectors include sensible ways of initializing the parameters to be learned (e.g., with extracted formants for initializing hidden variables composed of vocal tract resonances), and straightforward methods of diagnosing analyzing errors during model implementation. Since localist representations, unlike their distributed counterpart, do not superimpose patterns for signaling the presence of different linguistic labels, the hidden state variables not only are explanatory but also unambiguous. Further, the interpretable nature of the models allows complex causal and structured rela-

tionships to be built into them, free from the common difficulty associated with distributed representations. In fact, the hidden dynamic models have been constructed with many layers in the hierarchical hidden space, all with clear physical embodiment in speech production; e.g., Chapter 2 in [23]. However, the complex structure makes it very difficult to do discriminative parameter learning. As a result, nearly all versions of hidden dynamic models have adopted maximum-likelihood learning or data fitting approaches. For example, the use of linear or nonlinear Kalman filtering (E step of the EM algorithm) for learning the parameters in the generative state-space models has been applied to only maximum likelihood estimates [95, 101].

In contrast, the learning algorithm of BPTT commonly used for end-to-end training of the RNN with distributed representations for the hidden states performs discriminative training by directly minimizing linguistic label prediction errors. It is straightforward to do so in the formulation of the learning objective because of each element in the hidden state vector contributes to all linguistic labels due to the very nature of the distributed representation. It is very unnatural and difficult to do so in the generative hidden dynamic model based on localist representations of the hidden states, where each state and the associated model parameters typically contribute to only one particular linguistic unit, which is used to index the set of model parameters.

6.4 Parsimonious versus Massive Parameters

The final aspect of comparisons between the hidden dynamic model and the RNN concerns different ways to parameterize these two types of models. Due to the interpretable latent states in the hidden dynamic model as well as the parameters associated with them, speech knowledge can be used in the design of the model, leaving the size of free parameters to be relatively small. For example, when vocal tract resonance vectors are used to represent the hidden dynamics, a dimension of eight appears to be sufficient to capture the prominent dynamic properties responsible for the observed acoustic dynamics. Somewhat higher dimensionality is needed with the use of the hidden dynamic vectors associated with the articulators' configuration in speech production. The use of such parsimonious parameter sets, often called "small is good", is also facilitated by the localist representation of hidden state components and the related parameters that are connected or indexed to a specific linguistic unit. This contrasts with the distributed representation in the RNN where both the hidden state vector elements and the connecting weights are shared across all linguistic unit, thereby demanding many folds of more model parameters.

The ability to use speech-domain knowledge to construct the model with a parsimonious parameter set is both a blessing and a curse. Examples of such knowledge used in the past are the target-directed and smooth (i.e., non-oscillatory) hidden dynamics within each phone segment, an analytical relationship between the vocal tract resonance vector (both resonance frequencies and bandwidths), and both anticipatory and regressive types of coarticulation expressed in the latent space as a

result of the the hidden dynamics. With the right prediction of time-varying trajectories in the hidden space and then causally in the observed acoustic space, powerful constraints can be placed in the model formulation to reduce over-generation in the model space and to avoid unnecessarily large model capacity. On the other hand, the use of speech knowledge limits the growth of the model size as more data are made available in training. For example, when the dimensionality of the vocal tract resonance vectors goes beyond eight, many advantages of interpretable hidden vectors no longer hold. Since speech knowledge is necessarily incomplete, the constraints imposed on the model structure may be outweighed by the opportunity lost with increasingly large amounts of training data and by the incomplete knowledge.

In contrast, the RNN uses hardly any speech knowledge to constrain the model space due to the inherent difficulty of interpreting the ambiguous hidden state represented in a distributed manner. As such, the RNN in principle has the freedom to use massive parameters in keeping with the growing size of the training data. Lack of constraints may cause the model to over-generalize. This, together with the known difficulties of the various learning algorithms for the RNN as analyzed in [6] and reviewed in Section 5, has limited the progress of using RNNs in speech recognition for many years until recently. Some recent progress of RNNs applied to speech recognition involves various methods of introducing constraints either in the model construction or in the implementation of learning algorithms. For example, in the study reported in [49], the RNN's hidden state is designed with memory units, which, while constraining the variations of the recurrent hidden units and the associated weight parameters, still allow the massive model parameters to be used by simply increasing the size of the memory units. Separately, the RNN can also be constrained during the learning stage, where the size of the gradient computed by BPTT is limited by a threshold to avoid explosion as reported in [6, 75] or where the range of the permissible RNN parameters are constrained to be within what the "echo-state property" would allow [13, 25].

6.5 Comparing recognition accuracy of the two types of models

Given the analysis on and comparisons presented so far in this section between the generative hidden dynamic model using localist representations and the discriminative RNN using distributed representations, we see both types of the models have respective strengths and weaknesses. Here we compare the empirical performance of the two types of models in terms of speech recognition accuracy. For consistency reasons, we use the TIMIT phone recognition task for the comparison since no other common tasks have been used to assess both types of models in a consistent manner. It is important to point out that both types of the dynamic models are much more difficult to implement than other models in more common use for speech recogition, e.g. the GMM-HMM and DNN-HMM. While the hidden dynamic models have been evaluated on the large vocabulary tasks involving Switch-

board databases, e.g., [12, 66, 68, 86], the RNN has been mainly evaluated on the TIMIT task, e.g., [13, 25, 49, 90].

One particular version of the hidden dynamic model, called the hidden trajectory model, was developed and evaluated after careful design with approximations aimed to overcome the various difficulties associated with localist representations as discussed earlier in this section [38–40]. The main approximation involves using the finite impulse response filter to replace the infinite impulse response one as in the original state equation (67) of the state space formulation of the model. This version gives 75.2% phone recognition accuracy as reported in [38], somewhat higher than 73.9% obtained by a plain version of the RNN (but with very careful engineering) as reported in Table I (on page 303) of [90] and somewhat lower than 76.1% obtained by an elaborated version of the RNN with LSTM memory units without stacking as reported in Table I (on page 4) of [49]. (With less careful engineering, the plain RNN could only achieve 71.8% accuracy as reported in [25].) This comparison shows that the top-down generative hidden dynamic model based on localist representation of the hidden state performs similarly to the bottom-up discriminative RNN based on distributed representation of the hidden state. This is understandable due to the pros and cons of these different types of models analyzed throughout this section.

7 Summary and Discussions on Future Directions

This paper provides an overview on a rather wide range of computational models developed for speech recognition over the past 20 some years. These models are characterized by the use of linear or nonlinear dynamics in the hidden space not directly observed. The temporal unfolding of these dynamic sequence models make the related networks deep, with the depth being the length of the data sequence to be modeled. Among all the models surveyed in this chapter, there are two fundamentally opposing categories. First, we have the top-down hidden dynamic models of a generative nature. The hidden state adopts the localist representation with explicit physical interpretation and the model parameters are indexed with respect to each of the linguistic/phonetic classes in a parsimonious manner. Second, we have the bottom-up recurrent neural network (RNN) of a discriminative nature. The hidden state adopts the distributed representation with each unit in the hidden state or layer contributing to all linguistic classes.

Sections 2 and 3 in the early part of this chapter are devoted to the first, generative type of the dynamic models. Section 4 describes an interesting class of the deep neural network models (DNN) where the network with high depth is constructed independently of the length of the data sequence. In this sense, the DNN technically does not belong to the class of deep dynamic network models discussed above. We include the DNN in this chapter not only due to its prominent role in the current speech recognition practice but also due to the interesting way in which the generative DBN is integrated into the overall DNN learning. In Section 4, we also discuss how sequence dynamics, an essential part for any sensible speech model, is incor-

porated into the DNN-based speech model using the HMM as in interface. Section 5 then turns to detailed technical reviews on the second type of the (true) dynamic and deep network models for speech, the RNN, which is viewed as a generalization of the DNN where the network's depth is linked to the length of the data sequence.

The most important material of the chapter is Section 6, which compares the two types of the deep, dynamic models in four incisive aspects. The most critical aspect of the discussion is the localist versus distributed representations for the hidden states, with the respective strengths and weaknesses analysized in detail. The recognition accuracy achieved by both types of the models is shown to be comparable between the two, implying that the strengths and weaknesses associated with the different model types balance out with each other. (We have analyzed the error patterns and found rather distinct errors produced by the generative hidden dynamic model and by the RNN although the overall error rates are comparable.)

The comprehensive comparisons conducted in Section 6 shed insights into the question of how to leverage the strengths of both types of models while overcoming their respective weaknesses. Analyzing this future direction is actually the main motivation of this chapter. The integration of the two distinct types of generative and discriminative models may be done blindly as in the case discussed in Section 4, where the generative DBN is used effectively to initialize or pre-train the discriminative DNN. However, much better strategies can be pursued as present and future directions, given our sufficient understanding by now of the nature of the respective strengths and weaknesses associated with the two model types as elaborated in Section 6. As an example, one weakness associated with the discriminative RNN, which we briefly mentioned in Section 6.2, is that distributed representations are not suitable for input to the network. This difficulty has been circumvented in the preliminary work reported in [25] by first using the DNN to extract input features, which gains the advantages of distributed representations embedded in the hidden layers of the DNN. Then the DNN-extracted features equipped with distributed representations of the data are fed into the subsequent RNN, producing dramatic improvement of phone recognition accuracy from 71.8% to as high as 81.2%. Other ways to cleverly get around the problems with localist representations in the generative, deep, and dynamic model and the problems with distributed representations in the discriminative model counterpart are expected to also improve speech recognition performance. As a further example to this end, we also discussed in Section 6 the strength of the localist representation in easy interpretation of the hidden space of the model. One can take advantage of this strength by using the generative model to create new features that can be effectively combined with other features based on distributed representations. Some advanced approximate inference and learning techniques developed for deep generative models (e.g., [97,105]) may facilitate successful implementations of this strategy by learning better generative models than the several existing inference and learning methods in the literature (e.g., variational EM and extended Kalman filtering) discussed earlier in this chapter.

References

1. ACERO, A., DENG, L., KRISTJANSSON, T., AND ZHANG, J. HMM adaptation using vector taylor series for noisy speech recognition. In *Proc. Intl. Conf. on Spoken Language Proc.* (2000), pp. 869–872.
2. BAKER, J. Stochastic modeling for automatic speech recognition. In *Speech Recognition*, D. Reddy, Ed. Academic, New York, 1976.
3. BAKER, J., DENG, L., GLASS, J., KHUDANPUR, S., LEE, C.-H., MORGAN, N., AND O'SHGUGHNESSY, D. Research developments and directions in speech recognition and understanding, part i. *IEEE Signal Processing Mag. 26*, 3 (2009), 75–80.
4. BAKER, J., DENG, L., GLASS, J., KHUDANPUR, S., LEE, C.-H., MORGAN, N., AND O'SHGUGHNESSY, D. Updated MINDS report on speech recognition and understanding. *IEEE Signal Processing Mag. 26*, 4 (2009), 78–85.
5. BAUM, L., AND PETRIE, T. Statistical inference for probabilistic functions of finite state Markov chains. *Ann. Math. Statist. 37*, 6 (1966), 1554—1563.
6. BENGIO, Y., BOULANGER, N., AND PASCANU, R. Advances in optimizing recurrent networks. In *Proc. ICASSP* (Vancouver, Canada, 2013).
7. BENGIO, Y., BOULANGER-LEWANDOWSKI, N., AND PASCANU, R. Advances in optimizing recurrent networks. In *Proc. ICASSP* (Vancouver, Canada, May 2013).
8. BILMES, J. Buried markov models: A graphical modeling approach to automatic speech recognition. *Computer Speech and Language 17* (April—July 2003), 213—231.
9. BILMES, J. What HMMs can do. *IEICE Trans. Information and Systems E89-D*, 3 (March 2006), 869–891.
10. BODEN, M. A guide to recurrent neural networks and backpropagation. Tech. rep., TECHNICAL REPORT T2002:03, SICS, 2002.
11. BOURLARD, H., AND MORGAN, N. *Connectionist Speech Recognition: A Hybrid Approach*, vol. 247 of *The Kluwer International Series in Engineering and Computer Science*. Kluwer Academic Publishers, Boston, MA, 1994.
12. BRIDLE, J., DENG, L., PICONE, J., RICHARDS, H., MA, J., KAMM, T., SCHUSTER, M., PIKE, S., AND REAGAN, R. An investigation fo segmental hidden dynamic models of speech coarticulation for automatic speech recognition. *Final Report for 1998 Workshop on Langauge Engineering, CLSP, Johns Hopkins* (1998).
13. CHEN, J., AND DENG, L. A primal-dual method for training recurrent neural networks constrained by the echo-state property. In *Proc. ICLR* (2014).
14. CHIEN, J.-T., AND CHUEH, C.-H. Dirichlet class language models for speech recognition. *IEEE Trans. on Audio, Speech and Language Processing 27* (2011), 43–54.
15. DAHL, G., YU, D., DENG, L., AND ACERO, A. Large vocabulary continuous speech recognition with context-dependent DBN-HMMs. In *Proc. IEEE Intl. Conf. on Acoust., Speech and Signal Proc.* (2011).
16. DAHL, G., YU, D., DENG, L., AND ACERO, A. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Trans. on Audio, Speech and Language Processing 20*, 1 (jan 2012), 30–42.
17. DEMPSTER, A. P., LAIRD, N. M., AND RUBIN, D. B. Maximum-likelihood from incomplete data via the EM algorithm. *J. Royal Statist. Soc. Ser. B. 39* (1977).
18. DENG, L. A generalized hidden markov model with state-conditioned trend functions of time for the speech signal. *Signal Processing 27*, 1 (1992), 65–78.
19. DENG, L. A dynamic, feature-based approach to the interface between phonology and phonetics for speech modeling and recognition. *Speech Communication 24*, 4 (1998), 299–323.
20. DENG, L. Articulatory features and associated production models in statistical speech recognition. In *Computational Models of Speech Pattern Processing*. Springer-Verlag, New York, 1999, pp. 214–224.
21. DENG, L. Computational models for speech production. In *Computational Models of Speech Pattern Processing*. Springer-Verlag, New York, 1999, pp. 199–213.

22. DENG, L. Switching dynamic system models for speech articulation and acoustics. In *Mathematical Foundations of Speech and Language Processing*. Springer-Verlag, New York, 2003, pp. 115–134.
23. DENG, L. *DYNAMIC SPEECH MODELS — Theory, Algorithm, and Applications*. Morgan and Claypool, 2006.
24. DENG, L., AKSMANOVIC, M., SUN, D., AND WU, J. Speech recognition using hidden Markov models with polynomial regression functions as non-stationary states. *IEEE Trans. Acoustics, Speech and Signal Processing 2*, 4 (1994), 101–119.
25. DENG, L., AND CHEN, J. Sequence classification using high-level features extracted from deep neural networks. In *Proc. ICASSP* (2014).
26. DENG, L., DROPPO, J., AND ACERO, A. A Bayesian approach to speech feature enhancement using the dynamic cepstral prior. In *Proc. IEEE Intl. Conf. on Acoust., Speech and Signal Proc.* (may 2002), vol. 1, pp. I–829 –I–832.
27. DENG, L., HASSANEIN, K., AND ELMASRY, M. Analysis of the correlation structure for a neural predictive model with application to speech recognition. *Neural Networks 7*, 2 (1994), 331–339.
28. DENG, L., HINTON, G., AND KINGSBURY, B. New types of deep neural network learning for speech recognition and related applications: An overview. In *Proc. IEEE ICASSP* (Vancouver, Canada, May 2013).
29. DENG, L., HINTON, G., AND YU, D. Deep learning for speech recognition and related applications. In *NIPS Workshop* (Whistler, Canada, 2009).
30. DENG, L., KENNY, P., LENNIG, M., GUPTA, V., SEITZ, F., AND MERMELSTEN, P. Phonemic hidden markov models with continuous mixture output densities for large vocabulary word recognition. *IEEE Trans. Acoustics, Speech and Signal Processing 39*, 7 (1991), 1677–1681.
31. DENG, L., LEE, L., ATTIAS, H., AND ACERO, A. Adaptive kalman filtering and smoothing for tracking vocal tract resonances using a continuous-valued hidden dynamic model. *Audio, Speech, and Language Processing, IEEE Transactions on 15*, 1 (2007), 13–23.
32. DENG, L., LENNIG, M., SEITZ, F., AND MERMELSTEIN, P. Large vocabulary word recognition using context-dependent allophonic hidden markov models. *Computer, Speech and Language 4* (1991), 345–357.
33. DENG, L., AND LI, X. Machine learning paradigms in speech recognition: An overview. *Audio, Speech, and Language Processing, IEEE Transactions on 21*, 5 (2013), 1060–1089.
34. DENG, L., AND MA, J. A statistical coarticulatory model for the hidden vocal-tract-resonance dynamics. In *EUROSPEECH* (1999), pp. 1499–1502.
35. DENG, L., AND MA, J. Spontaneous speech recognition using a statistical coarticulatory model for the hidden vocal-tract-resonance dynamics. *J. Acoustical Society of America 108* (2000), 3036–3048.
36. DENG, L., AND O'SHAUGHNESSY, D. *SPEECH PROCESSING — A Dynamic and Optimization-Oriented Approach*. Marcel Dekker Inc, NY, 2003.
37. DENG, L., RAMSAY, G., AND SUN, D. Production models as a structural basis for automatic speech recognition. *Speech Communication 33*, 2-3 (Aug 1997), 93–111.
38. DENG, L., AND YU, D. Use of differential cepstra as acoustic features in hidden trajectory modelling for phonetic recognition. In *Proc. IEEE Intl. Conf. on Acoust., Speech and Signal Proc.* (2007), pp. 445–448.
39. DENG, L., YU, D., AND ACERO, A. A bidirectional target filtering model of speech coarticulation: two-stage implementation for phonetic recognition. *IEEE Trans. on Speech and Audio Processing 14* (2006), 256–265.
40. DENG, L., YU, D., AND ACERO, A. Structured speech modeling. *IEEE Trans. on Speech and Audio Processing 14* (2006), 1492–1504.
41. DIVENYI, P., GREENBERG, S., AND MEYER, G. *Dynamics of Speech Production and Perception*. IOS Press, 2006.
42. DROPPO, J., AND ACERO, A. Noise robust speech recognition with a switching linear dynamic model. In *Proc. IEEE Intl. Conf. on Acoust., Speech and Signal Proc.* (may 2004), vol. 1, pp. I – 953–6 vol.1.

43. FOX, E., SUDDERTH, E., JORDAN, M., AND WILLSKY, A. Bayesian nonparametric methods for learning markov switching processes. *IEEE Signal Processing Mag. 27*, 6 (nov. 2010), 43 –54.
44. FREY, B., DENG, L., ACERO, A., AND KRISTJANSSON, T. Algonquin: Iterating laplaces method to remove multiple types of acoustic distortion for robust speech recognition. In *Proc. Eurospeech* (2000).
45. GALES, M., AND YOUNG, S. Robust continuous speech recognition using parallel model combination. *IEEE Trans. on Speech and Audio Processing 4*, 5 (sep 1996), 352 –359.
46. GHAHRAMANI, Z., AND HINTON, G. E. Variational learning for switching state-space models. *Neural Computation 12* (2000), 831–864.
47. GONG, Y., ILLINA, I., AND HATON, J.-P. Modeling long term variability information in mixture stochastic trajectory framework. In *Proc. Intl. Conf. on Spoken Language Proc.* (1996).
48. GRAVES, A. Sequence transduction with recurrent neural networks. In *Representation Learning Workshp, ICML* (2012).
49. GRAVES, A., MAHAMED, A., AND HINTON, G. Speech recognition with deep recurrent neural networks. In *Proc. ICASSP* (Vancouver, Canada, 2013).
50. HINTON, G. A practical guide to training restricted boltzmann machines. *Momentum 9*, 1 (2010).
51. HINTON, G., DENG, L., YU, D., DAHL, G., MOHAMED, A., JAITLY, N., SENIOR, A., VANHOUCKE, V., NGUYEN, P., SAINATH, T., AND KINGSBURY, B. Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal Processing Mag. 29*, 6 (2012), 82–97.
52. HINTON, G., OSINDERO, S., AND TEH, Y. A fast learning algorithm for deep belief nets. *Neural Computation 18* (2006), 1527–1554.
53. HINTON, G., AND SALAKHUTDINOV, R. Reducing the dimensionality of data with neural networks. *Science 313*, 5786 (2006), 504 – 507.
54. HOLMES, W., AND RUSSELL, M. Probabilistic-trajectory segmental HMMs. *Computer Speech and Language 13* (1999), 3–37.
55. HUANG, X., ACERO, A., AND HON, H.-W. *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*. Prentice Hall, 2001.
56. JAEGER, H. *Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the "echo state network" approach*. GMD Report 159, GMD - German National Research Institute for Computer Science, 2002.
57. JELINEK, F. Continuous speech recognition by statistical methods. *Proceedings of the IEEE 64*, 4 (1976), 532 – 557.
58. JUANG, B.-H., LEVINSON, S. E., AND SONDHI, M. M. Maximum likelihood estimation for mixture multivariate stochastic observations of markov chains. *IEEE Trans. Information Theory 32*, 2 (1986), 307–309.
59. KINGSBURY, B., SAINATH, T., AND SOLTAU, H. Scalable minimum bayes risk training of deep neural network acoustic models using distributed hessian-free optimization. In *Proc. Interspeech* (2012).
60. LAROCHELLE, H., AND BENGIO, Y. Classification using discriminative restricted boltzmann machines. In *Proceedings of the 25th international conference on Machine learning* (2008), ACM, pp. 536–543.
61. LEE, L., ATTIAS, H., AND DENG, L. Variational inference and learning for segmental switching state space models of hidden speech dynamics. In *Proc. IEEE Intl. Conf. on Acoust., Speech and Signal Proc.* (april 2003), vol. 1, pp. I–872 – I–875.
62. LEE, L. J., FIEGUTH, P., AND DENG, L. A functional articulatory dynamic model for speech production. In *Proc. ICASSP* (Salt Lake City, 2001), vol. 2, pp. 797–800.
63. LIU, S., AND SIM, K. Temporally varying weight regression: A semi-parametric trajectory model for automatic speech recognition. *IEEE Trans. on Audio, Speech and Language Processing 22*, 1 (Jan 2014), 151–160.
64. M., S., D. YU, L. D., AND LEE, C.-H. Exploiting deep neural networks for detection-based speech recognition. *Neurocomputing* (2013).

65. MA, J., AND DENG, L. A path-stack algorithm for optimizing dynamic regimes in a statistical hidden dynamic model of speech. *Computer Speech and Language 14* (2000), 101–104.
66. MA, J., AND DENG, L. Efficient decoding strategies for conversational speech recognition using a constrained nonlinear state-space model. *Audio and Speech Processing, IEEE Transactions on 11*, 6 (2003), 590–602.
67. MA, J., AND DENG, L. Efficient decoding strategies for conversational speech recognition using a constrained nonlinear state-space model. *Audio, Speech, and Language Processing, IEEE Transactions on 11*, 6 (2004), 590–602.
68. MA, J., AND DENG, L. Target-directed mixture dynamic models for spontaneous speech recognition. *Audio and Speech Processing, IEEE Transactions on 12*, 1 (2004), 47–58.
69. MAAS, A. L., LE, Q., O'NEIL, T. M., VINYALS, O., NGUYEN, P., AND NG, A. Y. Recurrent neural networks for noise reduction in robust asr. In *Proc. INTERSPEECH* (Portland, OR, September 2012).
70. MARTENS, J., AND SUTSKEVER, I. Learning recurrent neural networks with hessian-free optimization. In *Proc. ICML* (Bellevue, WA, June 2011), pp. 1033–1040.
71. MESNIL, G., HE, X., DENG, L., AND BENGIO, Y. Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding. In *Proc. INTERSPEECH* (Lyon, France, August 2013).
72. MESOT, B., AND BARBER, D. Switching linear dynamical systems for noise robust speech recognition. *IEEE Trans. on Audio, Speech and Language Processing 15*, 6 (August 2007), 1850–1858.
73. MIKOLOV, T. *Statistical Language Models Based on Neural Networks*. PhD thesis, Ph. D. thesis, Brno University of Technology, 2012.
74. MIKOLOV, T., DEORAS, A., POVEY, D., BURGET, L., AND CERNOCKY, J. Strategies for training large scale neural network language models. In *Proc. IEEE ASRU* (Honolulu, HI, December 2011), IEEE, pp. 196–201.
75. MIKOLOV, T., KARAFIÁT, M., BURGET, L., CERNOCKÝ, J., AND KHUDANPUR, S. Recurrent neural network based language model. In *Proc. INTERSPEECH* (Makuhari, Japan, September 2010), pp. 1045–1048.
76. MIKOLOV, T., KOMBRINK, S., BURGET, L., CERNOCKY, J., AND KHUDANPUR, S. Extensions of recurrent neural network language model. In *Proc. IEEE ICASSP* (Prague, Czech, May 2011), pp. 5528–5531.
77. MOHAMED, A., DAHL, G., AND HINTON, G. Acoustic modeling using deep belief networks. *IEEE Trans. on Audio, Speech and Language Processing 20*, 1 (jan. 2012), 14–22.
78. MOHAMED, A., DAHL, G. E., AND HINTON, G. E. Deep belief networks for phone recognition. In *NIPS Workshop on Deep Learning for Speech Recognition and Related Applications* (2009).
79. MOHAMED, A., SAINATH, T., DAHL, G., RAMABHADRAN, B., HINTON, G., AND PICHENY, M. Deep belief networks using discriminative features for phone recognition. In *Proc. IEEE Intl. Conf. on Acoust., Speech and Signal Proc.* (may 2011), pp. 5060 –5063.
80. MORGAN, N. Deep and wide: Multiple layers in automatic speech recognition. *IEEE Trans. on Audio, Speech and Language Processing 20*, 1 (jan. 2012).
81. OSTENDORF, M., DIGALAKIS, V., AND KIMBALL, O. From HMM's to segment models: A unified view of stochastic modeling for speech recognition. *IEEE Trans. Speech and Audio Proc. 4*, 5 (September 1996).
82. OSTENDORF, M., KANNAN, A., KIMBALL, O., AND ROHLICEK, J. Continuous word recognition based on the stochastic segment model. *Proc. DARPA Workshop CSR* (1992).
83. OZKAN, E., OZBEK, I., AND DEMIREKLER, M. Dynamic speech spectrum representation and tracking variable number of vocal tract resonance frequencies with time-varying dirichlet process mixture models. *IEEE Trans. on Audio, Speech and Language Processing 17*, 8 (2009), 1518 –1532.
84. PASCANU, R., MIKOLOV, T., AND BENGIO, Y. On the difficulty of training recurrent neural networks. In *Proc. ICML* (Atlanta, GA, June 2013).
85. PAVLOVIC, V., FREY, B., AND HUANG, T. Variational learning in mixed-state dynamic graphical models. In *Proc. UAI* (Stockholm, 1999), pp. 522–530.

86. PICONE, J., PIKE, S., REGAN, R., KAMM, T., BRIDLE, J., DENG, L., MA, Z., RICHARDS, H., AND SCHUSTER, M. Initial evaluation of hidden dynamic models on conversational speech. In *Proc. IEEE Intl. Conf. on Acoust., Speech and Signal Proc.* (1999).
87. PUSKORIUS, G., AND FELDKAMP, L. Neurocontrol of nonlinear dynamical systems with kalman filter trained recurrent networks. *Neural Networks, (IEEE) Transactions on 5*, 2 (1998), 279–297.
88. RABINER, L., AND JUANG, B.-H. *Fundamentals of Speech Recognition*. Prentice-Hall, Upper Saddle River, NJ., 1993.
89. RENNIE, S., HERSHEY, J., AND OLSEN, P. Single-channel multitalker speech recognition — graphical modeling approaches. *IEEE Signal Processing Mag. 33* (2010), 66–80.
90. ROBINSON, A. J. An application of recurrent nets to phone probability estimation. *IEEE Transactions on Neural Networks 5*, 2 (August 1994), 298–305.
91. ROSTI, A., AND GALES, M. Rao-blackwellised gibbs sampling for switching linear dynamical systems. In *Proc. IEEE Intl. Conf. on Acoust., Speech and Signal Proc.* (may 2004), vol. 1, pp. I – 809–12 vol.1.
92. RUSSELL, M., AND JACKSON, P. A multiple-level linear/linear segmental HMM with a formant-based intermediate layer. *Computer Speech and Language 19* (2005), 205–225.
93. SAINATH, T., KINGSBURY, B., SOLTAU, H., AND RAMABHADRAN, B. Optimization techniques to improve training speed of deep neural networks for large speech tasks. *IEEE Transactions on Audio, Speech, and Language Processing 21*, 11 (Nov. 2013), 2267–2276.
94. SEIDE, F., G. LI, G., CHEN, X., AND YU, D. Feature engineering in context-dependent deep neural networks for conversational speech transcription. In *Automatic Speech Recognition and Understanding (ASRU), 2011 IEEE Workshop on* (2011), IEEE, pp. 24–29.
95. SHEN, X., AND DENG, L. Maximum likelihood in statistical estimation of dynamical systems: Decomposition algorithm and simulation results. *Signal Processing 57* (1997), 65–79.
96. STEVENS, K. *Acoustic Phonetics*. MIT Press, 2000.
97. STOYANOV, V., ROPSON, A., AND EISNER, J. Empirical risk minimization of graphical model parameters given approximate inference, decoding, and model structure. *Proc. AISTAT* (2011).
98. SUSKEVER, I., MARTENS, J., AND HINTON, G. E. Generating text with recurrent neural networks. In *Proc. 28th International Conference on Machine Learning* (2011).
99. SUTSKEVER, I. *Training Recurrent Neural Networks*. PhD thesis, Ph. D. thesis, University of Toronto, 2013.
100. SUTSKEVER, I., MARTENS, J., AND HINTON, G. E. Generating text with recurrent neural networks. In *Proc. ICML* (Bellevue, WA, June 2011), pp. 1017–1024.
101. TOGNERI, R., AND DENG, L. Joint state and parameter estimation for a target-directed nonlinear dynamic system model. *Signal Processing, IEEE Transactions on 51*, 12 (2003), 3061–3070.
102. TOGNERI, R., AND DENG, L. A state-space model with neural-network prediction for recovering vocal tract resonances in fluent speech from mel-cepstral coefficients. *Speech communication 48*, 8 (2006), 971–988.
103. TRIEFENBACH, F., JALALVAND, A., DEMUYNCK, K., AND MARTENS, J.-P. Acoustic modeling with hierarchical reservoirs. *IEEE Transactions on Audio, Speech, and Language Processing 21*, 11 (Nov. 2013), 2439–2450.
104. WRIGHT, S., KANEVSKY, D., DENG, L., HE, X., HEIGOLD, G., AND LI, H. Optimization algorithms and applications for speech and language processing. *IEEE Transactions on Audio, Speech, and Language Processing 21*, 11 (Nov. 2013), 2231–2243.
105. XING, E., JORDAN, M., AND RUSSELL, S. A generalized mean field algorithm for variational inference in exponential families. In *Proc. UAI* (2003).
106. YU, D., AND DENG, L. Speaker-adaptive learning of resonance targets in a hidden trajectory model of speech coarticulation. *Computer Speech and Language 27* (2007), 72–87.
107. YU, D., AND DENG, L. Discriminative pretraining of deep neural networks. *US Patent 20130138436 A1* (may 2013).

108. YU, D., DENG, L., AND DAHL, G. Roles of pre-training and fine-tuning in context-dependent DBN-HMMs for real-world speech recognition. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning* (2010).
109. YU, D., SEIDE, F., LI, G., AND DENG, L. Exploiting sparseness in deep neural networks for large vocabulary speech recognition. In *Proc. IEEE Intl. Conf. on Acoust., Speech and Signal Proc.* (2012), pp. 4409–4412.
110. YU, D., SINISCALCHI, S., DENG, L., AND LEE, C. Boosting attribute and phone estimation accuracies with deep neural networks for detection-based speech recognition. In *Proc. IEEE Intl. Conf. on Acoust., Speech and Signal Proc.* (2012).
111. ZEN, H., TOKUDA, K., AND KITAMURA, T. An introduction of trajectory model into HMM-based speech synthesis. In *Proc. of ISCA SSW5* (2004), pp. 191–196.
112. ZHANG, L., AND RENALS, S. Acoustic-articulatory modelling with the trajectory HMM. *IEEE Signal Processing Letters 15* (2008), 245248.