A Review of Graphical Model Architectures for Speech Recognition

Jeff Bilmes & Chris Bartels

{bilmes,bartels}@ee.washington.edu

University of Washington, Seattle

Department of Electrical Engineering

Seattle WA, 98195-2500

May 4th, 2005

1 Introduction

Graphs are a two-dimensional visual formalism that can be used to describe many different phenomena and are used in wide variety of fields including computer science, data and control flow, entity relationships and social networks, Petri and neural networks, software/hardware visualization, and parallel computation. The popularity of graphs is owing to their ability represent complex situations in an intuitive and visually appealing way.

Statistical graphical models are a family of graphical abstractions of statistical models, where important aspects (e.g., factorization) of such models are represented using graphs. In recent years and due to a wide range of research, it has become apparent that graphical models offer a mathematically formal but widely flexible means for solving many of the problems in speech and language processing. Graphs are able to represent events at the very high level (such as relationships between linguistic classes), at the very low level (such as correlations between spectral features or acoustic landmarks), and at all levels in between (such as lexical pronunciation). A fundamental advantage of graphical models is *rapidity* — with a graphical model, it is possible to quickly express a novel complicated idea in an intuitive, concise, and mathematically precise way, and to speedily and visually communicate that idea between colleagues. Moreover, with the right software, it is possible to rapidly prototype that idea on a standard desktop workstation.

This article discusses the foundations of the use of graphical models for speech recognition [42, 16, 26, 47, 25], giving detailed accounts of some of the more successful cases. Our discussion will in particular employ dynamic Bayesian networks (DBNs), and a DBN extension using the Graphical Model Toolkit's (GMTK's) basic *template*, a dynamic graphical model representation that is more suitable for speech and language systems. While this article will concentrate on speech recognition, it should be noted that many of the ideas contained here are also applicable to natural language processing and general time-series analysis.

This article assumes some familiarity with basic speech-to-text concepts [42, 16, 26, 46, 47, 25] and the basics of graphical models [41, 28, 32, 29], including notions such as hidden and observed variables, evidence, and factorization and conditional independence. Moreover, we will use the Matlab-like notation $1{:}N$ to denote the set of integers $\{1, 2, \ldots, N\}$. A set of N random variables (RVs) is denoted as $X_{1:N}$. Given any subset $S \subseteq 1{:}N$, where $S = \{S_1, S_2, \ldots, S_{|S|}\}$, then the subset of random variables is denoted as $X_S = \{X_{S_1}, X_{S_2}, \ldots, X_{S_{|S|}}\}$. Lastly, we use upper case letters (such as X, Q) to refer to random variables, and lower case letters (such as x, q) to refer to random variable values.

2 Dynamic Graphical Models

Graphical models [32, 41, 27, 29] are a set of formalisms each of which describes families of probability distributions. There are many different types of graphical models [41, 27, 32, 31] each having its own *semantics* [10] that govern how the graph specifies a set of factorization constraints on multi-variate probability distributions. Of course factorization and conditional independence go hand-in-hand, so factorization constraints typically (but not always) involve conditional independence properties. Bayesian networks are one type of graphical model where the graphs are directed and acyclic (DAG). In a Bayesian network (BN), the probability distribution over a set of variables $X_{1:N}$ factorizes with respect to a directed acyclic graph (DAG) as $p(x_{1:N}) = \prod_i p(x_i | x_{\pi_i})$ where $\pi_i \subset 1{:}N$ are the set of indices of X_i's immediate parents according to the BN's DAG. This factorization is called the directed factorization property [32]. There are many additional (and provably equivalent) characterizations of BNs, including the notion of d-separation [41, 32], but this one suffices for our discussion. It should be clear that because of the strong relationship between factorization and conditional independence, the above factorization implies that a Bayesian network expresses a large number of conditional independence statements to the extent that it has missing edges in the graph. Moreover, it should be clear that it is the common factorization properties of the family of probability distributions that makes for efficient probabilistic inference [41, 32, 28, 29].

Speech is inherently a temporal process, and any graphical model for speech must take this into account.

Accordingly, dynamic graphical models [5] are graphs that represent the temporal evolution of the statistical properties of a speech signal, and ideally in such a way to improve automatic speech recognition (ASR) accuracy. For speech recognition, dynamic Bayesian networks (DBNs) [15, 51, 49, 37, 50, 10] have been most successfully used. DBNs are simply Bayesian networks with a repeated "template" structure over time. Other than this regularity, however, DBNs have exactly the same semantics as Bayesian networks. Specifically, a DBN of length T is a directed acyclic graph $\mathcal{G} = (\mathbf{V}, \mathbf{E}) = (\bigcup_{t=1}^{T} \mathbf{V}_t, \mathbf{E}_T \cup \bigcup_{t=1}^{T-1} \mathbf{E}_t \cup \mathbf{E}_t^{\rightarrow})$ with node set $\mathbf{V}$, and edge set $\mathbf{E}$ comprising pairs of nodes. If $uv \in \mathbf{E}$ for $u, v \in \mathbf{V}$, then uv is an edge of $\mathcal{G}$. The sets $\mathbf{V}_t$ are the nodes at time slice t, $\mathbf{E}_t$ are the *intra-slice* edges between nodes in $\mathbf{V}_t$, and $\mathbf{E}_t^{\rightarrow}$ are the *inter-slice* edges between nodes in $\mathbf{V}_t$ and $\mathbf{V}_{t+1}$. A DBN, however, does not typically have this much flexibility. That is, a DBN is specified using a "rolled up" template giving nodes that are repeated in each slice, the intra-slice edges among those nodes, and the inter-slice edges between nodes of adjacent slices. In other words, $\mathbf{V}_t$ and $\mathbf{V}_{t+\tau}$ have the same set of random variables that are different only in that the time indexes of the variables differ by τ. The same is true for $\mathbf{E}_t$ and $\mathbf{E}_{t+\tau}$ and also $\mathbf{E}_t^{\rightarrow}$ and $\mathbf{E}_{t+\tau}^{\rightarrow}$. The DBN template is then unrolled to any desired length T to yield the DBN $\mathcal{G}$. As in any BN, the collection of edges pointing into a node corresponds to a conditional probability function (CPF). In a DBN, the CPF of a node is shared (or tied) with the CPF of all other nodes that have come from the same underlying node in the DBN template. In other words, if $V_t \in \mathbf{V}_t$ with parents V_{π_t}, then $p(V_t = v | V_{\pi_t} = v_\pi) = p(V_\tau = v | V_{\pi_\tau} = v_\pi)$ for all t, τ and for all scalar values v and vector values v_π. Therefore, it is possible to represent a DBN of unbounded length, but with only a finite description length and a finite number of parameters.

It is well known that the hidden Markov model (HMM) is one type of DBN [44]. Even given its success and flexibility, however, the HMM is only one small model within the enormous family of statistical techniques representable by DBNs. Like an HMM, a DBN makes a temporal Markov assumption, meaning that the future is independent of the past given the present. In fact, it is true that many (but not all, see Section 3.7) DBNs can be "flattened" into a corresponding HMM, but staying within the DBN framework has several advantages. First, in DBN form, there can be exploitable computational advantages since the DBN explicitly represents factorization properties, and factorization is the key to tractable probabilistic inference [29]. These factorizations, however, are lost when the model is flattened. Secondly, the factorization specified by a DBN implies that there are constraints that the model must obey. For example, consider Figure 1 which shows a two-Markov chain DBN with chains (Q_t^1, Q_t^2). A flattened HMM would have one

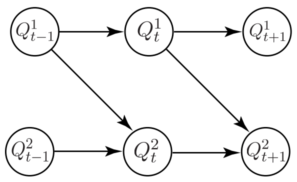


Figure 1: Simple two-stream Markov chain

chain $R_t \equiv (Q_t^1, Q_t^2)$ with transition probabilities set as follows:

$$p(R_t = r_t | R_{t-1} = r_{t-1}) = p(Q_t^1 = q_t^1, Q_t^2 = q_t^2 | Q_{t-1}^1 = q_{t-1}^1, Q_{t-1}^2 = q_{t-1}^2)$$

where $r_t \equiv (q_t^1, q_t^2)$ is the joint HMM state space. Such flattening, however, ignores the factorization constraint expressed by the graph, which is:

$$\begin{aligned} & p(Q_t^1 = q_t^1, Q_t^2 = q_t^2 | Q_{t-1}^1 = q_{t-1}^1, Q_{t-1}^2 = q_{t-1}^2) \\ & = p(Q_t^1 = q_t^1 | Q_{t-1}^1 = q_{t-1}^1) p(Q_t^2 = q_t^2 | Q_{t-1}^1 = q_{t-1}^1, Q_{t-1}^2 = q_{t-1}^2) \end{aligned}$$

In other words, not all possible $p(r_t | r_{t-1})$ CPFs are allowed given the graph due to its conditional independence property — in the above, other things in addition to Q_{t-1}^1 would influence the distribution of Q_t^1 if no assumptions were made. Of course the HMM can represent a distribution designed under these constraints. But when training parameters, we must find the optimal solution within the parameter space subject to the these constraints. Moreover, it is during training (when the amount of training data might be limited) that one wants to reduce the amount of parameter freedom (via a set of constraints on the model) as much as possible. Since a DBN naturally expresses factorization, it is an ideal candidate to train model parameters in this case. A third advantage of DBNs is that they convey structural information about the underlying problem. Such structure might represent anything from the result of data-mining process [3] on the training data to dependencies over high-level knowledge sources, or both. In either case, information about a domain is visually and intuitively portrayed.

Loosely speaking, DBN probabilistic inference (a generalization of the Baum-Welch procedure for HMMs [42]) has a computational cost upper bound (i.e., it is possible to show that this is the worst case) equal to very roughly the joint state space (the number of combined variable assignments that can occur with non-zero probability) of all the variables in two time slices of the graph [5, 37, 49, 50, 51] multiplied by the total number of time slices T. Therefore, one must be careful when adding variables to a DBN that the cost does not become prohibitive. While this article does not get into the specifics of DBN inference, it should be known that this cost often strongly depends on the DBN triangulation method used [5, 1]. In other words, adding variables will often, but not necessarily, cause a significant increase in computational cost.

2.1 The GMTK Dynamic Template

Before exploring various ASR constructs using graphical models, we define the Graphical Models Toolkit's (GMTK) [7, 5] extension of a DBN template. This extension facilitates the expression of graphical models for speech recognition and natural language processing.

A GMTK template extends a standard DBN template in five distinct ways. First, it allows for not only forward but also backward directed time links. This allows for a richer model specification that enables, for example, representations of reverse-time effects such as coarticulation in human speech (see Figure 2). Second, network slices may span multiple time frames, so slices are called *chunks*. This allows for the specification of multi-rate models with rationally related rates. Either of these first two extensions means that the future need not always be independent of the past given the present (directed cycles are of course still disallowed). For example, we might have a variable V_t with parents W_{t-1} and W_{t+1}, so conditioning on V_t does not render W_{t-1} and W_{t+1} independent. Alternatively, V_t might have W_{t-2} as a parent, so conditioning on variables at time $t-1$ does not render V_t and W_{t-2} independent. Third, a GMTK template includes a built-in specification mechanism for switching parents (see Section 3.1). Fourth, parents of a variable may be multiple chunks in the past or into the future, so not only are past and future no longer independent given the present, the temporal Markov property common in HMMs might still not be present even at the chunk level (but it is present if we agglomerate all chunks together inclusively between children and their parents). Fifth, it allows for different special multi-frame structures to occur at both the beginning and the end of the unrolled network. Specifically, a GMTK template consists of a prologue subgraph $\mathcal{P} = (V^p, E^p)$, a chunk subgraph (to be unrolled) $\mathcal{C} = (V^c, E^c)$, an epilogue subgraph $\mathcal{E} = (V^e, E^e)$, and appropriate edges in between. Each of these subgraphs can be any number of time slices long. Letting $T(\mathcal{P})$ denote the number of time frames contained within $\mathcal{P}$ (similarly for $\mathcal{C}$ and $\mathcal{E}$), the number of frames in an unrolled GMTK-DBN $\mathcal{G}^T$ may thus be $T = T(\mathcal{P}) + kT(\mathcal{C}) + T(\mathcal{E})$ for k a positive integer. As we will see in the following sections, using this extended template definition can decrease the time required to express both standard as well as novel statistical models for speech recognition systems.

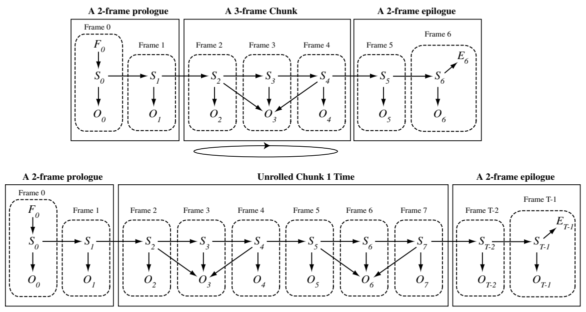


Figure 2: A multi-frame GMTK template (top) with a two-frame prologue $\mathcal{P}$, a 3-frame chunk $\mathcal{C}$, and a 2-frame epilogue $\mathcal{E}$ and unrolled one time (bottom).

3 Graphical Model Speech Architectures

In this section, we describe a number of graphical architectures for various speech recognition decoder constructs. Note that each of the graphs presented are meant only for speech recognition ”decoding” (meaning, producing a sequence of words from the acoustic waveform). In general, the graphical architectures used when training the parameters of a speech recognition system will be slightly different (and will include sub-graph structures for constructs such as counters, sequencers, lattices, and so on). Each of the following graphs falls under one or more of the following broad goals for using graphs for ASR.

The first goal for graphical models in ASR is to design structures that explicitly and efficiently represent typical or novel ASR control constructs. These may include features such as parameter tying, the sequencing of states through a list or lattice, smoothing by mixing, the order n in an n-gram language model, the size of the context in context-dependent phone models (e.g., mono- or tri-phones), the type of this context (within-word or cross-word tri-phones), single or multiple pronunciations per word, and so on. Any given ASR system might or might not implement some of these features (e.g., System X supports bi-grams but not tri-grams). Within the graphical models framework, however, essentially all features are available (given the appropriate general and efficient software). Such models, pioneered by [51], are now called "explicit graph representations" [7], where random variables can exist for such purposes as word identification, numerical word position, phone or phoneme identity, the indication of a phone transition, a count of the number of phones, and direct copies of variables from other time frames. There indeed can be multiple possible representations of the same underlying control structure, but these representations might have different degrees of conceptual simplicity and/or computational requirements. It is therefore imperative to design structures that are both simple to explain and fast to run. Explicit models stand in contrast to the fully "implicit approach", where much of this control information is represented by a single hidden Markov chain with a large sparse transition matrix, and where an integer state encodes all contextual and control information determining the allowable sequencing. The implicit approach, in fact, corresponds roughly to the result of flattening (e.g., composition and minimization [36]).

The implicit and explicit approaches are useful in different circumstances. The explicit approach allows for the representation of an unlimited number of constructs. Moreover, when the underlying set of hidden variables must have factorization constraints, it is natural to use an explicit model. If the underlying model consists of multiple hierarchies, the explicit approach is again quite natural. On the other hand, if no factorization is needed, if the CPFs are often sparse (containing many zeros), and if the underlying operation is always a single hierarchical composition of multiply-nested Markov chains, then the implicit approach can be both natural and very fast [36].

A second goal in forming graphical models for ASR is in latent knowledge modeling. Here, a set of (partially) factored dynamic hidden variables are used to represent any unknown but presumed to exist phenomenon (or "cause") of the speech signal. This includes knowledge from very high linguistic levels to the very low acoustic level. For example, hidden variables can represent dialog act, word or phrase category, pronunciation variant, speaking rate, mode/style/gender, word morphology, vocal-tract or articulatory configuration, acoustic channel condition, noise condition, or Gaussian component. Knowledge modeling using graphical models is a promising area for further research as many believe that high-level knowledge is underutilized in existing state-of-the-art speech recognition systems. Graphical models can provide the infrastructure in which this knowledge can be exploited.

A third goal consists of proper *observation modeling*. Most speech recognition systems represent speech as a sequence of feature vectors corresponding to overlapping windows of speech. The feature vectors (most often Mel-frequency cepstral coefficients, or MFCCs) are then represented by state-conditional multivariate Gaussian mixtures. It is quite easy to extend such a representation by allowing relationships between individual elements of the current feature vector and elements in other feature vectors either before, during, or after the current one. Such a model can more appropriately match the underlying statistics extant in speech as represented by the current series of feature vectors.

Lastly, the most challenging problem (involving all three of the above) is that of automatic structure learning. Here, the problem is to automatically determine a best graph structure (or a set of good structures) using the available training data [10]. This problem is particularly difficult when learning hidden structures, as not only is data for the hidden structure unavailable (making it difficult to learn), the existence of each such hidden variable can be put into question. Therefore, many standard statistical model selection techniques are inappropriate.

In the next several sections, we will see how these four goals arise when we examine a number of different graphs for representing the speech recognition process. We will start with a fairly basic graph (for a bi-gram decoder) and expand on this idea to the much more elaborate.

3.1 Basic phone-based bi-gram decoding structure

Our first model is a simple phone-based decoder that uses bi-gram language models and single-state phones (Figure 3). The figure corresponds to a GMTK template unrolled one time: the first frame is the prologue $\mathcal{P}$, the second and third frames are each a chunk $\mathcal{C}$, and the last frame is the epilogue $\mathcal{E}$. The structure explicitly represents the unflattened and hierarchical constructs that go into building such an ASR system with these attributes even though, when flattened, it can be represented by an HMM. Each such attribute is implemented using a separate temporal layer of random variables in the graph. We define the following random variables at time t: $\mathcal{W}_t$ is the word, $\mathcal{W}_t^{tr}$ is the word transition, $\mathcal{W}_t^{ps}$ is the word position, $\mathcal{P}_t^{tr}$ is the phone transition, $\mathcal{P}_t$ is the phone, and $\mathcal{O}_t$ is the acoustic feature vector. The graph thus specifies the

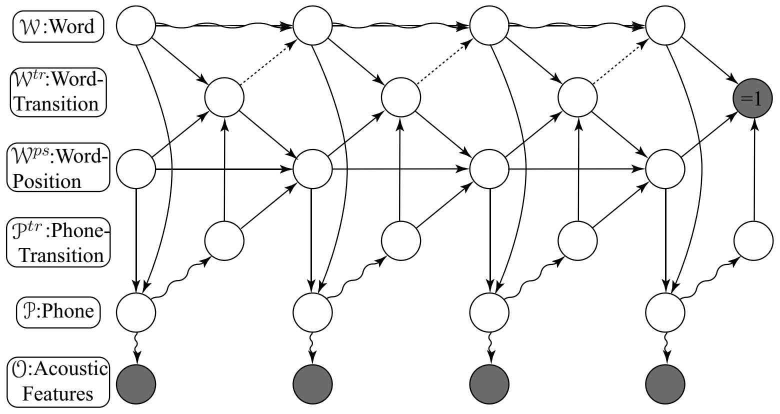

$\mathcal{W}$:Word
$\mathcal{W}^{tr}$:Word-Transition
$\mathcal{W}^{ps}$:Word-Position
$\mathcal{P}^{tr}$:Phone-Transition
$\mathcal{P}$:Phone
$\mathcal{O}$:Acoustic Features
=1

Figure 3: Explicit representation of a phone-based ASR decoder with a bi-gram language model. Observations are depicted as shaded nodes, and hidden variables are unshaded. Also, deterministic dependencies are represented by straight lines, and purely random ones are represented by wavy lines. If a line is both straight and wavy, it switches between deterministic and random based on a switching parent, connected via a finely-dashed edge. Each time frame has (from top to bottom) a variable for: word identity, word transition, word position (position of current phone within a word), phone transition, phone, and acoustic feature vector. Note that the final word transition binary variable is observed to equal the value 1, thus disabling any decodings that do not end in a complete word.

following factorization for a length T utterance:

$$p(\mathcal{W}_{1:T}, \mathcal{W}^{tr}_{1:T}, \mathcal{W}^{ps}_{1:T}, \mathcal{P}^{tr}_{1:T}, \mathcal{P}_{1:T}, \mathcal{O}_{1:T})$$
$$= \prod_t p(\mathcal{O}_t|\mathcal{P}_t)p(\mathcal{P}^{tr}_t|\mathcal{P}_t)p(\mathcal{P}_t|\mathcal{W}^{ps}_t, \mathcal{W}_t)p(\mathcal{W}^{tr}_t|\mathcal{W}_t, \mathcal{W}^{ps}_t, \mathcal{P}^{tr}_t)p(\mathcal{W}^{ps}_t|\mathcal{W}^{tr}_{t-1}, \mathcal{W}^{ps}_{t-1}, \mathcal{P}^{tr}_{t-1})p(\mathcal{W}_t|\mathcal{W}_{t-1}, \mathcal{W}^{tr}_{t-1})$$

Note, however, that it is always the case that $\mathcal{W}^{tr}_T = 1$ to ensure that a proper final word is decoded. Some of the CPFs are deterministic (represented by straight lines in the figure, see two paragraphs below for further explanation) and some are purely random (represented by zig-zagged or wavy lines). To compute the probability of the observations, we must perform the sum:

$$p(\mathcal{O}_{1:T}) = \sum_{\mathcal{W}_{1:T}, \mathcal{W}^{tr}_{1:T}, \mathcal{W}^{ps}_{1:T}, \mathcal{P}^{tr}_{1:T}, \mathcal{P}_{1:T}} p(\mathcal{W}_{1:T}, \mathcal{W}^{tr}_{1:T}, \mathcal{W}^{ps}_{1:T}, \mathcal{P}^{tr}_{1:T}, \mathcal{P}_{1:T}, \mathcal{O}_{1:T})$$

and then exploit the factorization property as specified by the graph to best distribute sums into products to reduce computation. It should be noted that in this graph, many of the terms in the sum will have value zero since one or more of the factors at each time frame will themselves be zero. This is because many of the factors correspond to CPFs that are deterministic. One of the goals of inference in such graphs, therefore, is

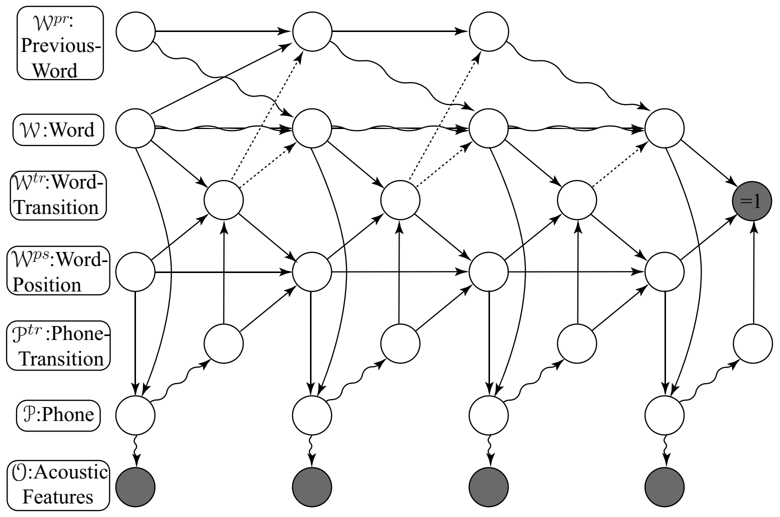

$\mathcal{W}^{pr}$: Previous-Word
$\mathcal{W}$:Word
$\mathcal{W}^{tr}$:Word-Transition
$\mathcal{W}^{ps}$:Word-Position
$\mathcal{P}^{tr}$:Phone-Transition
$\mathcal{P}$:Phone
$\mathcal{O}$:Acoustic Features
=1

Figure 4: Tri-gram language model decoder

to take advantage of this sparsity in order to avoid the unnecessary computation of summing multiple zeros together which takes time but does nothing.

We next describe each individual CPF. First, the acoustic features $\mathcal{O}_t$ are a random function of the phone in the CPF $p(\mathcal{O}_t|\mathcal{P}_t)$. As is typical in an HMM system, each value for $\mathcal{P}_t$ will select a particular Gaussian mixture to be used to produce probability scores for that phone. Next, the phone transition variable $\mathcal{P}_t^{tr}$ is a binary indicator that specifies when the model should advance to the next phone — $\mathcal{P}_t^{tr}$ takes its cue from the phone variable $\mathcal{P}_t$ in the CPF $p(\mathcal{P}_t^{tr}|\mathcal{P}_t)$, meaning that each phone may have its own separate geometric duration distribution. Note that the phone transition variable is also purely random, since for each phone there is a non-zero probability of both staying at that phone state and moving on to the next phone (similar to the state transition probability in the classical HMM).

The phone variable $\mathcal{P}_t$ is a purely deterministic function of its parents $\mathcal{W}_t^{ps}$ and W_t. This means that, given the current word and word position, the phone is known with certainty. Another way of saying this is that $p(\mathcal{P}_t = i|\mathcal{W}_t^{ps} = k, W_t = l) = 1$ if $i = f(k, l)$ for a deterministic function $f(k, l)$ of its arguments k, l, and the probability is zero otherwise. In this model, therefore, each position of a word corresponds to one and only one phone (so multiple pronunciations are not represented, although they easily could be). Another deterministic relationship is the word transition. Here, a word transition $\mathcal{W}_t^{tr} = 1$ occurs only if the model makes a phone transition $\mathcal{P}_t^{tr} = 1$ out of the last position $\mathcal{W}_t^{ps} = k$ of a given word $\mathcal{W}_t = w$ where w is the index of a word that is comprised of k total phones.

At the beginning of the utterance, the word variable $\mathcal{W}_1$ starts out using a unigram distribution over words in the vocabulary. Also, the word position $\mathcal{W}_1^{ps}$ starts out at value 0 with probability 1 (e.g., $p(\mathcal{W}_1^{ps} =$

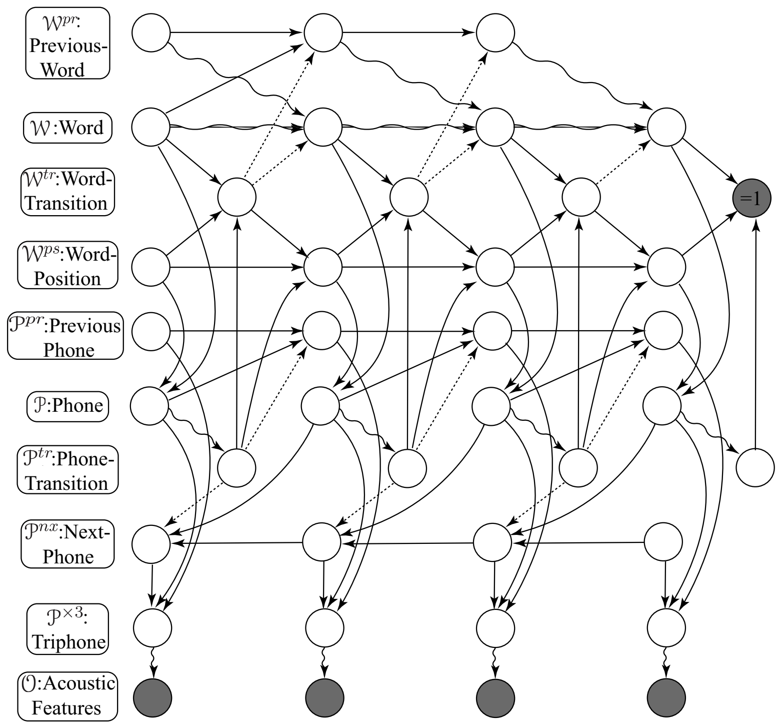

$\mathcal{W}^{pr}$: Previous-Word
$\mathcal{W}$:Word
$\mathcal{W}^{tr}$:Word-Transition
$\mathcal{W}^{ps}$:Word-Position
$\mathcal{P}^{pr}$:Previous Phone
$\mathcal{P}$:Phone
$\mathcal{P}^{tr}$:Phone-Transition
$\mathcal{P}^{nx}$:Next-Phone
$\mathcal{P}^{\times 3}$: Triphone
$\mathcal{O}$:Acoustic Features
=1

Figure 5: Cross-word tri-phone decoder

$0) = 1$). After the first frame, these variables take on more complex distributions. First, the word position variable $\mathcal{W}_t^{ps}$ has essentially three behaviors depending on the value of its parents: 1) it might not change from a frame to the next frame, 2) it might increment in value by one (i.e., $\mathcal{W}_t^{ps} = \mathcal{W}_{t-1}^{ps} + 1$), or 3) it might reset to zero. First, if there is no phone transition ($\mathcal{P}_{t-1}^{tr} = 0$) then the word position does not change (so $p(\mathcal{W}_t^{ps} = i | \mathcal{W}_{t-1}^{ps} = i, \mathcal{W}_{t-1}^{tr} = j, \mathcal{P}_{t-1}^{tr} = 0) = 1$, and this is true for all j, 0 or 1 in this case, but note that when $\mathcal{P}_{t-1}^{tr} = 0$ we will always have that $\mathcal{W}_{t-1}^{tr} = 0$). If there is a phone transition ($\mathcal{P}_{t-1}^{tr} = 1$) and the model is not in the last position of the word, then the word position will increment with probability one. Lastly, if there is a phone transition ($\mathcal{P}_{t-1}^{tr} = 1$) and also the model is in the last position of the word, then that will cause a word transition to occur $\mathcal{W}_{t-1}^{tr} = 1$ which will subsequently cause the next word position to be reset to zero (so $p(\mathcal{W}_t^{ps} = 0 | \mathcal{W}_{t-1}^{ps} = i, \mathcal{W}_{t-1}^{tr} = 1, \mathcal{P}_{t-1}^{tr} = 1) = 1$, for all i). Note that this behavior is entirely deterministic, and could easily be described using a set of if-then rules or a decision tree. The deterministic behavior of the graph is in fact what helps to implement various ASR control structures.

The final variable needing description is the word variable at frames greater than one. The word variable uses the switching parent functionality (see also [9]), where the existence or implementation of an edge can depend on the value of some other variable(s) in the network, referred to as the switching parent(s). The edge from the switching parent to its child is drawn as a finely dashed edge in our graphs. In this case, the

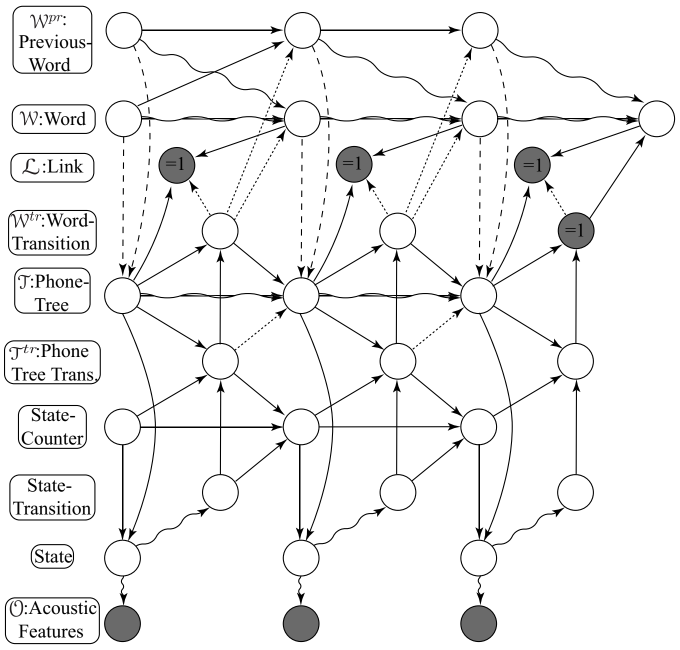
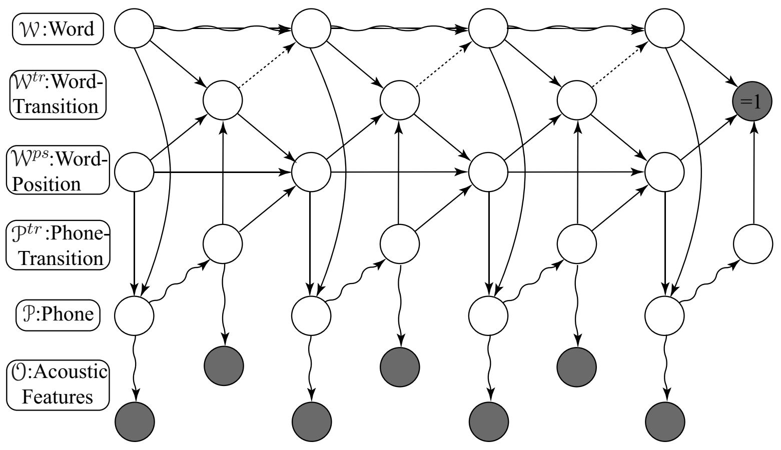

$\mathcal{W}^{pr}$: Previous-Word
$\mathcal{W}$:Word
$\mathcal{L}$:Link
$\mathcal{W}^{tr}$:Word-Transition
$\mathcal{T}$:Phone-Tree
$\mathcal{T}^{tr}$:Phone Tree Trans.
State-Counter
State-Transition
State
$\mathcal{O}$:Acoustic Features
=1

Figure 6: Tree-structured decoder

switching parent is the word transition variable. When the word transition is zero ($\mathcal{W}^{tr}_{t-1} = 0$), it causes the word variable $\mathcal{W}_t$ to copy its previous value, i.e., $\mathcal{W}_t = \mathcal{W}_{t-1}$ with probability one. When a word transition occurs $\mathcal{W}^{tr}_{t-1} = 1$, however, it switches the implementation of the word-to-word edge to use a “bi-gram” language model probability $p(\mathcal{W}_t|\mathcal{W}_{t-1})$. Strictly speaking, this graph does not switch parents, but it does switch implementations of a CPF, from a deterministic function to a random bi-gram language model.

Looking broadly at the graph, we see that most of the variables are hidden (unshaded) but some are observed. The observed variables include the acoustic feature vectors (e.g., MFCCs). As an ASR system typically uses feature *vectors* (modeled say using a Gaussian mixture), here the vector is represented simply using a single node. By doing this, any statistical assumptions made at the acoustic level between individual feature vector elements are not explicitly stated (but see Section 3.7 below). The other observed variable is the final word transition, which is always observed to be one. By insisting that the final word transition is unity, the graph ensures that all decodings end in a word transition. This makes sure that any decodings representing only a partial final word (meaning the final word transition would have value zero) are never considered. The ability of having this special behavior in the last frame is easily enabled by the extended DBN template (Section 2.1).



$\mathcal{W}$:Word
$\mathcal{W}^{tr}$:Word-Transition
$\mathcal{W}^{ps}$:Word-Position
$\mathcal{P}^{tr}$:Phone-Transition
$\mathcal{P}$:Phone
$\mathcal{O}$:Acoustic Features
=1

Figure 7: Transition specific decoder

3.2 Basic phone-based tri-gram decoding architecture

In many ASR systems, moving from a bi-gram to a tri-gram language model can require a significant change in internal system implementation. Under the graphical models framework, however, it is easy to express such a construct. In fact, the graph from the previous section can be extended in only a minor way to represent tri-gram language models. As is well known, going from a bi-gram to a tri-gram will increase the state space of the model by a factor of $|W|$, the vocabulary size [26], and our case is no exception. The ease of going to this new model, however, will be quite apparent.

In this and other graphs, all the variables evolve at the rate of the frame, rather than the rate of the word. Therefore, to implement a tri-gram it is not sufficient to just add an edge in Figure 3 from W_{t-2} to W_t. This is because the word from two frames ago is most often the same as the current word, and is not the same as the unique word that occurred before the previous unique word, which for a given hypothesis might have occurred many frames into the past. Instead, we must explicitly keep track of the identity of the word that was most recently different — i.e., the identity of the word just before the previous word transition, regardless of when in the past that transition took place. We do this with an additional *previous word* variable $\mathcal{W}_t^{pr}$ in Figure 4. When there is no word transition ($\mathcal{W}_t^{tr} = 0$), both the word $\mathcal{W}_t$ and previous word $\mathcal{W}_t^{pr}$ variables do not change when moving to time $t+1$. When a word transition ($\mathcal{W}_t^{tr} = 1$) occurs, the new previous word $\mathcal{W}_{t+1}^{pr}$ gets a copy of the previous current word $\mathcal{W}_t$ with probability one. Moreover, the new current word $\mathcal{W}_{t+1}$ is chosen with the tri-gram probability, but it conditions on the previous current word $\mathcal{W}_t$ and the *previous* previous word $\mathcal{W}_t^{pr}$, as needed by a tri-gram.

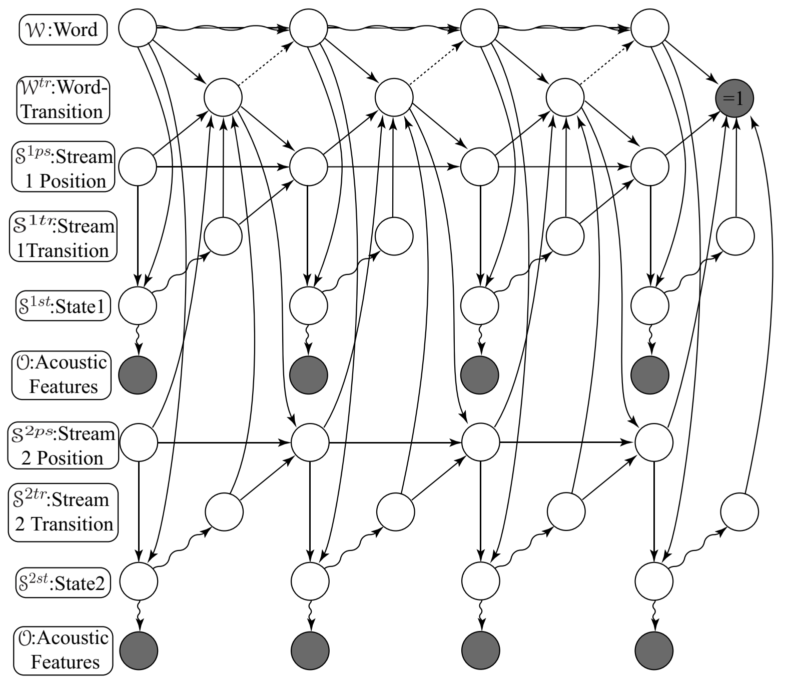
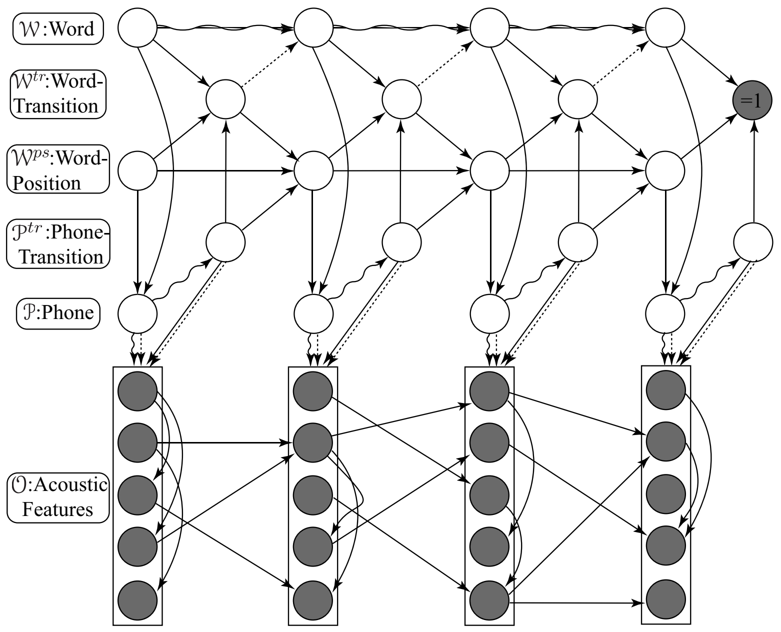

$\mathcal{W}$:Word
$\mathcal{W}^{tr}$:Word-Transition
$\mathcal{S}^{1ps}$:Stream 1 Position
$\mathcal{S}^{1tr}$:Stream 1Transition
$\mathcal{S}^{1st}$:State1
$\mathcal{O}$:Acoustic Features
$\mathcal{S}^{2ps}$:Stream 2 Position
$\mathcal{S}^{2tr}$:Stream 2 Transition
$\mathcal{S}^{2st}$:State2
$\mathcal{O}$:Acoustic Features
=1

Figure 8: Generic multi-stream semi-asynchronous decoder

3.3 Cross-word tri-phone architecture

Another technique that is typically employed by speech recognition systems is that of cross-word tri-phones [46]. Tri-phone models are those where the acoustic observation is conditioned not only on the currently hypothesized phone, but also makes the assumption that the current acoustics are significantly influenced by the preceding and following phonetic context (i.e., coarticulation). Tri-phone models do this by saying that the distribution over the acoustic frame depends on the current, previous, and next phone.

We solve this problem again in a graphical setting. Our solution makes use of a novel feature in the GMTK template, namely the use of backwards time edges. The solution is shown in Figure 5. The top part of the graph shows the graph construct for the tri-gram language model we saw before — in particular, the phone variable is a deterministic function of the word and word position. But in this case the phone layer of the graph has three (rather than one) variables, the previous phone variable $\mathcal{P}_t^{pr}$ and a next phone variable $\mathcal{P}_t^{nx}$ which when combined together with the phone $\mathcal{P}_t$ produce the tri-phone variable $\mathcal{P}_t^{\times 3}$. When a phone transition does not occur $\mathcal{P}_t^{tr} = 0$, the phone variable keeps its same value $\mathcal{P}_{t+1} = \mathcal{P}_t$ from frame to frame (since neither the word changes, $\mathcal{W}_{t+1} = \mathcal{W}_t$, nor does the position increment, $\mathcal{W}_{t+1}^{ps} = \mathcal{W}_t^{ps}$), the *next* previous phone retains its value $\mathcal{P}_{t+1}^{pr} = \mathcal{P}_t^{pr}$, and the *current* next phone $\mathcal{P}_t^{nx}$ simply copies its future value from $\mathcal{P}_{t+1}^{nx}$. When a phone transition does occur ($\mathcal{P}_t^{tr} = 1$), then the current phone $\mathcal{P}_t$ gets copied to the *next* previous phone $\mathcal{P}_{t+1}^{pr}$, a new phone $\mathcal{P}_{t+1}$ is chosen based on the incremented new word position



$\mathcal{W}$:Word
$\mathcal{W}^{tr}$:Word-Transition
$\mathcal{W}^{ps}$:Word-Position
$\mathcal{P}^{tr}$:Phone-Transition
$\mathcal{P}$:Phone
$\mathcal{O}$:Acoustic Features
=1

Figure 9: Transition and phone dependent buried Markov model decoder

$\mathcal{W}^{ps}_{t+1}$, and that new phone is then copied backwards in time into the *current* next phone, i.e., $\mathcal{P}^{nx}_t = \mathcal{P}_{t+1}$ via the backwards time edge for use at time t. The backwards time edges have thus represented anticipatory coarticulation effects in the speech process. Note, however (and similar to the tri-gram language model case), that it is not sufficient to place a link from $\mathcal{P}_{t+1}$ directly to the tri-phone variable $\mathcal{P}^{\times 3}_t$ or observation $\mathcal{O}_t$ because most often $\mathcal{P}_{t+1}$ will just be the same as $\mathcal{P}_t$. That is, $\mathcal{P}_{t+1}$ is the phone of the next frame, but it is not the phone that will next be used — $\mathcal{P}_{t+1}$ is the next used phone only when $\mathcal{P}^{tr}_t = 1$, at which point it is copied back to the current next phone $\mathcal{P}^{nx}_t$, and to earlier current next phones $\mathcal{P}^{nx}_\tau$, for $\tau < t$ as needed until the previous phone transition.

This process also works across word boundaries yielding a true cross-word tri-phone model. When the model starts a new word, the last phone of the previous word is copied into $\mathcal{P}^{pr}_t$. Moreover, when we are at the last phone of a word, $\mathcal{P}^{nx}_t$ will indicate the first phone of whatever the next word will be, regardless of when in the future it starts.

3.4 Tree-structured lexicon architecture

Even for HMM-only large vocabulary speech recognition systems, the decoding search space can be computationally prohibitive. It is most often necessary to arrange and reorganize states in this space in a manner that is efficient to search, and one way is to use a *tree-structured lexicon* [46] where the prefixes of each word are represented explicitly, and the state space probabilistically branches only when extending the prefixes of words that no longer match.

Such a construct can be represented using the graph structure shown in Figure 6. In this graph, a variable is used which corresponds to a phone tree $\mathcal{T}_t$. This variable encodes the prefix-tree for the entire word lexicon in a large but very sparse stochastic matrix. Each state of this variable either corresponds to a "non-terminal" state (which represents multiple words all of which have some prefix in common), and "terminal" states (which represent a single word). There are two operations $\mathcal{T}_t$ might do over a frame depending on the phone tree transition variable $\mathcal{T}^{tr}_{t-1}$ which also acts as a switching parent (dashed edge from $\mathcal{T}^{tr}_{t-1}$ to $\mathcal{T}_t$). If $\mathcal{T}^{tr}_t = 0$ (so there is no phone tree transition), then $\mathcal{T}_{t+1} = \mathcal{T}_t$ with unity probability. If there is a phone tree transition, however, then the next phone tree state is governed by the sparse phone tree probability table $p(\mathcal{T}_{t+1} = j|\mathcal{T}_t = i, \mathcal{T}^{tr}_t = 1, \mathcal{W}^{tr}_t = 0)$. At some point, we will reach a terminal phone tree state that corresponds to the end of a word, lets say $\mathcal{T}_t = k$. When making a phone tree transition out of this state ($\mathcal{T}^{tr}_t = 1$), this will cause a word transition $\mathcal{W}^{tr}_t = 1$ to occur which will choose a new word. The issue, however, is that we have not yet applied the true language model probability score for this new word (since it was not known until now) in the context of the two previous words. Moreover, it is not possible simply to allow the language model structure (top two rows in the graph) to probabilistically choose the next word, since the word at this point is already pre-determined by the terminal state in the phone tree. Therefore, a soft evidence [4, 41] construct is employed. Here, a special and always observed link variable $\mathcal{L}_t = 1$ is used to insist on consistency between the word corresponding to the terminal state of the phone tree variable and the next word chosen by the tri-gram language model, but this consistency is enforced only when a word-transition occurs. This is realized by having $\mathcal{W}^{tr}_t$ be a switching parent of $\mathcal{L}_t$. When there is no word transition ($\mathcal{W}^{tr}_t = 0$), then $\mathcal{L}_t$ is uncoupled from its other parents $\mathcal{W}_{t+1}$ and $\mathcal{T}_t$ (and thus has no effect). When there is a word transition, the event $\mathcal{L}_t = 1$ is explained (with non-zero probability) only when the next word $\mathcal{W}_{t+1} = w$ is the word corresponding to the terminal state of the current phone tree variable $\mathcal{T}_t$. We can moreover extend this model to provide early state pruning by allowing the phone tree probability table to have scores for the most optimistic of the subsequent words [46]. We can also provide for the context dependence of such scores by including edges from $\mathcal{W}_t$ and $\mathcal{W}^{pr}_t$ to $\mathcal{T}_t$ (indicated in the figure by long-dashed edges). Also, we can model the case when certain phone tree states are both non-terminal and terminal, meaning that a valid word might be a prefix of another word. Both cases must be considered, and this can be done by having the word transition $\mathcal{W}^{tr}_t$ be a stochastic function of the phone tree $\mathcal{T}_t$ so that such a state will both indicate a word, but also continue on to cover the words for which it is a prefix.

In summary, using just deterministic and random variables, and the soft-evidence construct, we have re-created a tree-structured speech decoder system, but again all out of the same basic building blocks used to produce other models.

3.5 Transition Explicit Model

One advantage of the graphical modeling approach is that it helps to solve problems that under more conventional approaches are more difficult to deploy. In this section, we describe one possible example. In the past, it has been argued that speech regions corresponding to spectral transitions might include much if not all of the underlying message [20, 6]. By making a simple modification to the graph in Figure 3, it is possible to include a construct that puts different emphasis on the speech signal depending on the current phone-transition condition.

In Figure 7, relative to Figure 3, we have attached to the phone transition $\mathcal{P}_t^{tr}$ another edge and observed variable. The observation could, for example, consist of features that are designed to provide information about spectral transition [20]. One option, say in an HMM system, would be to append this information to the standard feature vector, perform dimensionality reduction or feature selection, and hope for the best. The potential problem, however, is that the novel information might be relevant only part of the time, might not be needed at this level of classification (between different phones), and thus might even introduce noise in the system. Instead, in the figure we have focused this information directly on the part of the speech recognition system most likely to be beneficially influenced, namely the phone transition variable. Therefore, the graphical modeling framework and its machinery have made expressing this novel model very easy.

3.6 Multi-observation & multi-hidden stream semi-asynchronous architecture.

It is also easy to define a generic semi-asynchronous multi-stream and/or multi-modal model for speech recognition as shown in Figure 8. The streams may lie both over the observation space (e.g., multiple streams of feature vectors) and also over the hidden space (multiple semi-synchronous streams of hidden variables).

The word variable, $\mathcal{W}_t$, is at the top of the structure. In the previous models each word was composed of a sequence of phones, but now we generalize this to being composed of two sequences of generic "states." This allows us to have two independent representations of a word. Examples include an audio and a video feature stream [22, 21, 40, 39, 2], differing streams of audio features [48], different articulatory sequences [17, 30, 34, 43, 33, 19], or different spectral sub-band streams [35, 14, 23]. The position variables $\mathcal{S}^{1ps}$ and $\mathcal{S}^{2ps}$ are counters which define the current position in the sequence. The current state, $\mathcal{S}_t^{1st}$ (respectively $\mathcal{S}_t^{2st}$), is calculated from $\mathcal{W}_t$ and $\mathcal{S}_t^{1ps}$ (respectively $\mathcal{S}_t^{2ps}$). The state variables could have unique values for each word and counter value combination, or (as in a phone representation) could have the same values for many word and counter value combinations. The transition variables, $\mathcal{S}_t^{1tr}$ and $\mathcal{S}_t^{2tr}$, are random and determine when the sequences transition into the next state. The word transition variable, $\mathcal{W}_t^{tr}$, determines the end of the word. The state sequences are free to evolve at different rates for the duration of the word, but in order for a word transition to occur, both sequences must transition out of their last state for the current word – this means that for $\mathcal{W}_t^{tr} = 1$ we must have that both $\mathcal{S}_t^{1tr} = 1$ and $\mathcal{S}_t^{2tr} = 1$, and also that $\mathcal{S}_t^{1ps}$ and $\mathcal{S}_t^{2ps}$ are at their last respective value for the current word $\mathcal{W}_t$. Note that there is no requirement that the two sequences use the same number of states per word, nor is there any requirement that the two sequences line-up in any way — effectively, in accumulating the probability of the word, all alignments are considered along with their corresponding alignment probability. Moreover, additional constraints can be placed on the various alignments by including edges directly between the variables $\mathcal{S}_t^{1ps}$, $\mathcal{S}_t^{1st}$ and the variables $\mathcal{S}_t^{2ps}$, $\mathcal{S}_t^{2st}$. Alternatively, a soft evidence [4, 41] mechanism can be used to place symmetric constraints on the stream variables. In either case, certain alignments can be either probabilistically weighted or entirely removed from consideration with non-zero probability.

In this model, words are the points of synchrony because the two streams must start and end together at word boundaries. One can also envision a model where the points of synchrony are not the words but are rather sub-word (syllables, articulatory gestures), or suprasegmental (phrases, sentence boundaries) aspects of speech. Moreover, it is quite easy to generalize to more than two streams.

3.7 Architectures over Observed Variables

In this last example, we see a graph that explicitly represents various factorization properties over the observation vectors themselves (Figure 9). In this graph, the observation explicitly shows the length M *vector* of acoustic features at time t (this is written as $\mathcal{O}_{1:M,t}$), and where each of the scalar observation elements $\mathcal{O}_{j,t}$ might depend on a different scalar observation element $\mathcal{O}_{i,\tau}$ at a different position and/or different time, either earlier or later. Note that this model shows that the dependencies over observations switch from frame to frame, and they switch based on both the current phone $\mathcal{P}_t$ and the current phone transition state $\mathcal{P}_t^{tr}$, variables that are both normal and switching parents, and so both solid and finely-dashed edges are displayed. Note that such models have been called auto-regressive HMMs [45, 11], or for the case with specific element-wise dependencies as depicted in the figure, Buried Markov models (BMMs) [3].

Unlike the models of the earlier sections, BMMs cannot be flattened into an equivalent HMM because the additional edges are between continuous feature vectors, and so this influences the model in non-discrete uncountable ways, something that can not be exactly flattened into a countable (or even finite) state space. Moreover, BMMs provide a promising vehicle for structure learning [8, 18] since the structure to be learned is over sequences of only observed feature vectors (rather than hidden variables). Efficient and provably optimal structure learning methods exist in certain cases when the variables are observed [12, 24, 38]. In past work, in fact, it was shown how to learn the structure in a discriminative fashion over such variables [3].

3.8 Architecture Summary

Looking over the architectures from the previous sections, we see that many of the constructs are already available in ASR systems. A key advantage of GMs is that all of these constructs can be represented in a single, unified framework. In general under this framework, many other modifications to the above given structures can be quickly utilized (Section 3.5 shows one such example), and even slight modifications can lead to quite different ASR systems. While some of these modifications will improve performance and others will not, graphical model machinery can make it easy to both express a new model, and with the right software tools, to rapidly prototype it in a real ASR system. In addition, since the space of possible directed graphical models is so large, it is likely that even radically new ideas can be represented and rapidly prototyped in this framework without having to re-write an entire ASR system. This is the most important promise of the graphical modeling paradigm, but it does require a working toolkit.

4 Graphical Models Software Infrastructure

Building implementations of the aforementioned graphical architectures is greatly facilitated with a graphical models toolkit tuned specifically to speech and language processing. The Graphical Models Toolkit (GMTK), is one such system [7]. GMTK's goal is to provide the researcher with a rich and flexible set of abilities to express models in terms of graphs, but at the same time hide many of the technical details regarding the specifics of graphical algorithms. The process by which a speech researcher uses GMTK is as follows: First, a user specifies a graphical model using GMTK's extended DBN language (described in Section 2.1). Given the graph specification, off-line processing is performed [5] that partitions the graph into three disjoint sections distinct from the prologue, chunk, and epilogue. Next, the three new partitions of the graph are separately triangulated [1, 29], and junction trees for each section are formed and then merged to form a joint triangulated junction tree template. Once the graph is triangulated, it can be used by the inference algorithm for EM training, Viterbi-style decoding, or sampling. The overall goal of this processing is to improve computational efficiency to the extent possible. How precisely inference works and the underlying complexity of the models requires a detailed discussion requiring space much greater than is available in this article, but such a discussion will appear in future publications (for now, the reader may refer to [41, 13, 28, 37]). The researcher can thus concentrate on what is important, namely the model specification according to the four goals in Section 3. More details may be found in [7, 5, 1].

5 Conclusion

In this article, we have described in detail a number of graphical models that can be used to express speech recognition systems. We have demonstrated how graphical models solve problems for the speech scientist for which more conventional methods (e.g., HMMs) would either be unsuitable or at best a poor platform in which to rapidly prototype such a new idea. A key promise of graphical models is that by allowing researchers to experiment quickly, it will be possible to reject ideas that perform poorly, and advance ideas that perform well. This, of course, is not possible without a good software infrastructure. In the future, it indeed may be possible that the speech community will experience a wealth of new ideas, all expressed and implemented in this new graphical modeling paradigm.

6 Acknowledgments

The first version of GMTK was written by Jeff Bilmes and Geoff Zweig. Zweig also co-designed the models in Sections 3.1 and 3.2 as part of the 2001 CLSP-JHU summer workshop, and is an early pioneer in these sorts of representations for speech recognition. Recently, a new version of GMTK has been written by the authors to increase features, flexibility, and speed, and to enable open source. This work was supported by NSF grant IIS-0093430.

References

[1] C. Bartels and J. Bilmes. Elimination is not enough: Non-minimal triangulations for graphical models. Technical Report UWEETR-2004-0010, University of Washington, Dept. of Electrical Engineering, 2004.

[2] M.J. Beal, H. Attias, and N. Jojic. Audio-video sensor fusion with probabilistic graphical models. In *Proc. ECCV*, 2002.

[3] J. Bilmes. Buried Markov models: A graphical modeling approach to automatic speech recognition. *Computer Speech and Language*, 17:213—231, April—July 2003.

[4] J. Bilmes. On soft evidence in bayesian networks. Technical Report UWEETR-2004-0016, University of Washington, Dept. of EE, 2004.

[5] J. Bilmes and C. Bartels. On triangulating dynamic graphical models. In *Uncertainty in Artificial Intelligence: Proceedings of the Nineteenth Conference (UAI-2003)*, pages 47–56. Morgan Kaufmann Publishers, 2003.

[6] J. Bilmes, N. Morgan, S.-L. Wu, and H. Bourlard. Stochastic perceptual speech models with durational dependence. *Intl. Conference on Spoken Language Processing*, November 1996.

[7] J. Bilmes and G. Zweig. The Graphical Models Toolkit: An open source software system for speech and time-series processing. *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing*, 2002.

[8] J. Bilmes, G. Zweig, T. Richardson, K. Filali, K. Livescu, P. Xu, K. Jackson, Y. Brandman, E. Sandness, E. Holtz, J. Torres, and B. Byrne. Discriminatively structured graphical models for speech recognition: JHU-WS-2001 final workshop report. Technical report, CLSP, Johns Hopkins University, Baltimore MD, 2001. `http://www.clsp.jhu.edu/ws2001/groups/gmsr/GMRO-final-rpt.pdf`.

[9] J. A. Bilmes. Graphical models and automatic speech recognition. In R. Rosenfeld, M. Ostendorf, S. Khudanpur, and M. Johnson, editors, *Mathematical Foundations of Speech and Language Processing*. Springer-Verlag, New York, 2003.

[10] J.A. Bilmes. Dynamic Bayesian Multinets. In *Proceedings of the 16th conf. on Uncertainty in Artificial Intelligence*. Morgan Kaufmann, 2000.

[11] P.F. Brown. *The Acoustic Modeling Problem in Automatic Speech Recognition*. PhD thesis, Carnegie Mellon University, 1987.

[12] C.K. Chow and C.N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Trans. on Info. Theory*, 14, 1968.

[13] R. G. Cowell, A. P. Dawid, S. L. Lauritzen, and D. J. Spiegelhalter. *Probabilistic Networks and Expert Systems*. Springer, 1999.

[14] K. Daoudi, D. Fohr, and C. Antoine. A new approach for multi-band speech recognition based on probabilistic graphical models. In *Proc. Int. Conf. on Spoken Language Processing*, Beijing, China, November 2000.

[15] T. Dean and K. Kanazawa. Probabilistic temporal reasoning. *AAAI*, pages 524–528, 1988.

[16] J.R. Deller, J.G. Proakis, and J.H.L. Hansen. *Discrete-time Processing of Speech Signals*. MacMillan, 1993.

[17] L. Deng and H. Sameti. Transitional speech units and their representation by regressive Markov states: Applications to speech recognition. *IEEE Transactions on speech and audio processing*, 4(4):301–306, July 1996.

[18] M. Deviren and Khalid Daoudi. Structure learning of dynamic Bayesian networks in speech recognition. In *European Conf. on Speech Communication and Technology (Eurospeech)*, 2001.

[19] M. Hasegawa-Johnson et. al. Landmark-based speech recognition: Report of the 2004 Johns Hopkins summer workshop. In *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing*, Philadelphia, March 2005.

[20] Sadaoki Furui. On the role of spectral transition for speech perception. *Journal of the Acoustical Society of America*, 80(4):1016–1025, October 1986.

[21] A. Garg, Y. Pavlovic, and J.M. Rehg. Audio-visual speaker detection using dynamic bayesian networks. In *Fourth IEEE International Conference on Automatic Face and Gesture Recognition, 2000.*, pages 384—390, Grenoble, France, 2000.

[22] J. N. Gowdy, A. Subramanya, C. Bartels, and J. Bilmes. DBN-based multi-stream models for audio-visual speech recognition. *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing*, May 2004.

[23] G. Gravier, M. Sigelle, and G. Chollet. A markov random field based multi-band model. In *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing*, 2000.

[24] K-U. Höffgen. Learning and robust learning of product distributions. In *Proceedings of the sixth annual conference on Computational learning theory*, pages 77–83. ACM Press, 1993.

[25] X.D. Huang, A. Acero, and H.-W. Hon. *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*. Prentice Hall, 2001.

[26] F. Jelinek. *Statistical Methods for Speech Recognition*. MIT Press, 1997.

[27] F.V. Jensen. *An Introduction to Bayesian Networks*. Springer, 1996.

[28] F.V. Jensen. *Bayesian Networks and Decision Graphs*. Springer, 2001.

[29] M.I. Jordan and C. M. Bishop, editors. *An Introduction to Graphical Models*. to be published, 200x.

[30] K. Kirchhoff. Syllable-level desynchronisation of phonetic features for speech recognition. In *Proc. Int. Conf. on Spoken Language Processing*, 1996.

[31] F. R. Kschischang, B. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Trans. Inform. Theory*, 47(2):498–519, 2001.

[32] S.L. Lauritzen. *Graphical Models*. Oxford Science Publications, 1996.

[33] K. Livescu and J. Glass. Feature-based pronunciation modeling with trainable asynchrony probabilities. In *Proc. Int. Conf. on Spoken Language Processing*, Jeju, South Korea, October 2004.

[34] K. Livescu, J. Glass, and J. Bilmes. Hidden feature models for speech recognition using dynamic bayesian networks. In *European Conf. on Speech Communication and Technology (Eurospeech)*, 7th, Geneva, Switzerland, 2003.

[35] N. Mirghafori and N. Morgan. Combining connectionist multi-band and full-band probability streams for speech recognition of natural numbers. In *Proceedings of the International Conference on Spoken Language Processing*, pages 743–746, 1998.

[36] M. Mohri, F. C. N. Pereira, and M. Riley. The design principles of a weighted finite-state transducer library. *Theoretical Computer Science*, 231(1):17–32, 2000.

[37] K. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, U.C. Berkeley, Dept. of EECS, CS Division, 2002.

[38] M. Narasimhan and J. Bilmes. PAC-learning bounded tree-width graphical models. In *Uncertainty in Artificial Intelligence: Proceedings of the Twentieth Conference (UAI-2004)*. Morgan Kaufmann Publishers, 2004.

[39] A. Nefian, L. Liang, X. Pi, and K. Murphy. Dynamic bayesian networks for audio-visual speech recognition. *EURASIP, Journal of Applied Signal Processing*, 11:1—15, 2002.

[40] A. Nefian, L. Liang, X. Pi, L. Xiaoxiang, C. Mao, and K. Murphy. A coupled hmm for audio-visual speech recognition. In *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing*, 2002.

[41] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 2nd printing edition, 1988.

[42] L.R. Rabiner and B.-H. Juang. *Fundamentals of Speech Recognition*. Prentice Hall Signal Processing Series, 1993.

[43] M. Richardson, J. Bilmes, and C. Diorio. Hidden-articulator Markov models for speech recognition. *Speech Communication*, 41:511, October 2003.

[44] P. Smyth, D. Heckerman, and M.I. Jordan. Probabilistic independence networks for hidden Markov probability models. Technical Report A.I. Memo No. 1565, C.B.C.L. Memo No. 132, MIT AI Lab and CBCL, 1996.

[45] C.J. Wellekens. Explicit time correlation in hidden Markov models for speech recognition. *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing*, pages 384–386, 1987.

[46] S. Young. A review of large-vocabulary continuous-speech recognition. *IEEE Signal Processing Magazine*, 13(5):45–56, September 1996.

[47] S. Young, J. Jansen, J. Odell, D. Ollason, and P. Woodland. *The HTK Book*. Entropic Labs and Cambridge University, 2.1 edition, 1990's.

[48] Y. Zhang, Q. Diao, S. Huang, W. Hu, C. Bartels, and J. Bilmes. DBN based multi-stream models for speech. In *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing*, Hong Kong, China, April 2003.

[49] G. Zweig. *Speech Recognition with Dynamic Bayesian Networks*. PhD thesis, U.C. Berkeley, 1998.

[50] G. Zweig. Bayesian network structures and inference techniques for automatic speech recognition. *Computer Speech and Language*, 17:173—193, April—July 2003.

[51] G. Zweig and S. Russell. Speech recognition with dynamic Bayesian networks. *AAAI-98*, 1998.