Dynamic Graphical Models

Peter Reinprecht

Signal Processing and Speech Communication Laboratory

May 17, 2011
Outline

Introduction

Static Graphical Models

Dynamic Graphical Models

DGM for Speech Recognition

Conclusion
Outline

Introduction

Static Graphical Models

Dynamic Graphical Models

DGM for Speech Recognition

Conclusion
What is a Graph?

A Graph ...

- "... is a two-dimensional visual formalism."
- "... is used to describe different phenomena."
- "... represent complex situations in an intuitive and visually appealing way."
Components in GM

- Random Variables (vertices)
  - observed RV
- Dependencies (edges)
  - directed
  - undirected
Outline

Introduction

Static Graphical Models

Dynamic Graphical Models

DGM for Speech Recognition

Conclusion
Static Graphical Models

probability distribution

\[ p = p(X_1, X_2, ..., X_N) \]

Number of RV

\[ N = |V| \]

- Model: \( G = (V, E) \)
Markov Random Fields (MRF)

properties: $M$
Graphical Model: $\mathcal{F}(G, M)$

undirected graph

- Independence
- Factorization
- Cliques
Independence

Requires no direct connection between RVs Here:

- $X_1 \perp X_5 | \{X_2, X_3, X_4\}$
- $X_2 \perp X_4 | \{X_1, X_3, X_5\}$
Factorization

factorize $p$ to potential functions $\{\phi(\cdot)\}$

$$p(x_1, x_2, x_3, x_4, x_5) \propto \phi(x_1, x_2, x_3)\phi(x_2, x_3, x_5)$$
$$\phi(x_1, x_3, x_4)\phi(x_3, x_4, x_5)$$
Cliques

fully connected set of vertices

- \{X_1, X_2, X_3\}
- \{X_2, X_3, X_5\}
- \{X_1, X_3, X_4\}
- \{X_3, X_4, X_5\}
Bayesian Network (BN)

properties: $\mathcal{M}^{bn}$
Graphical Model: $\mathcal{F}(G, \mathcal{M}^{bn})$

directed graph

- Factorization
- Independence
BN Example

Factorization

\[ p(x_1, x_2, x_3, x_4, x_5) = p(x_2)p(x_1|x_2)p(x_3|x_1, x_2) \]
\[ p(x_5|x_2, x_3)p(x_4|x_1, x_3, x_5) \]

Independence

- \[ X_2 \perp X_4 | \{X_1, X_3, X_5\} \]
- \[ X_1 \perp X_5 | \{X_2, X_3\} \]
Belief propagation messaging

\[
\phi_{u_1 \rightarrow v}(v) = \sum_{u_1} \psi_{u_1, v}(u_1, v) \phi_{u_1}(u_1)
\]

\[
\phi_{v \rightarrow u}(u) = \sum_{v} \psi_{u, v}(u, v) \phi_{u_1 \rightarrow v}(v) \phi_{u_2 \rightarrow v}(v)
\]

\[
\phi_{u_2 \rightarrow v}(v) = \sum_{u_2} \psi_{u_2, v}(u_2, v) \phi_{u_2}(u_2)
\]
Triangulation

(a) A graph with seven nodes is eliminated in node order.
(b) The triangulated graph with fill-in edges.
(c) The template unrolled two times.
Outline

Introduction

Static Graphical Models

Dynamic Graphical Models

DGM for Speech Recognition

Conclusion
Dynamic Graphical Model (DGM)

DGM consist of

- $G = (V, E)$
- properties $\mathcal{M}$
- integer expansion parameter $T \geq 0$

$$\mathcal{F}(G, \mathcal{M}, T)$$
DGM template

Prologue
- $G^p = (V_p, E_p)$

Chunk
- $G^c = (V_c, E_c)$

Epilogue
- $G^e = (V_e, E_e)$
HMM as DGM

\[ p(\bar{x}_1:T, q_1:T) \propto \prod_{t=1}^{T} \phi(\bar{x}_t, q_t) \prod_{t=2}^{T} \phi(q_{t-1}, q_t) \]

\[ \Delta = \prod_{t=1}^{T} \phi'(q_{t-1}, q_t) \]

(a) An HMM as a DGM. (b) The DGM for the effective state space of the HMM. (c) A junction tree corresponding to the HMM.
The forward recursion is often described using a rectangular trellis graph (not a graphical model) as shown in Figure 5(d). Each recursion corresponds to a column of nodes in this graph—in the figure, the elements corresponding to $a_3^1 q_4^2$ and $a_4^1 q_9^2$ are both marked with arrows.

From the figure, we see that the separators in the junction tree correspond to the columns of the trellis, while the cliques in the junction tree correspond to two successive columns (i.e., the transitions between columns). The reason for the $O(N^2 T)$ memory usage is that the forward computation is only implicitly computing the cliques. Each set of computations for $a_t^1 q_2$ over all $q$ corresponds to a clique expansion (i.e., to an enumeration of all of the values of the variables in the clique) and then immediate reduction. Since the cliques are never stored, it is not necessary to use $O(N^2 T)$ memory. Forward recursion thus corresponds to a form of message in the junction tree between successive separators. Of course, the computation is still $O(N^2 T)$ since each $a_t^1 q_2$ requires $O(N^2)$ steps.

In general, in exact HMM inference, the goal is to “visit” each node in the HMM trellis, and this can be viewed as a form of search procedure \[26\]. The problem of “search” in artificial intelligence corresponds to expanding a very large space of possible elements as efficiently as possible. For example, given a factored function over variables indexed by set $V$, such as $f_1^x V^5 w C f_2^x C_2$, where each $C(\{V, \ldots\})$, the goal might be to identify a maximum element $\arg\max_x f_1^x V^2$. An alternative might be to sum $f$ for all values of $x$. One can imagine doing either of these naively using a set of nested loops, where we first iterate over all values $x_1$, and then $x_2$, and so on, and at the deepest level, when all elements of $x$ are instantiated, we can evaluate $f$. In general, any partial or complete set of variable assignments is called a “hypothesis” and along with each hypothesis, the goal might be to associate a weight $\alpha_i$ with each hypothesis $i$. One can imagine doing either of these naively using a set of nested loops, where we first iterate over all values $x_1$, and then $x_2$, and so on, and at the deepest level, when all elements of $x$ are instantiated, we can evaluate $f$.

HMM as DGM - junction tree

- Time complexity
  - $O(N^2 T)$

- Memory complexity
  - $O(N T)$
HMM as DGM - forward recursion

\[ \alpha_t(q) \triangleq p(\bar{x}_1:T, Q_t = q) = p(\bar{x}_t|q) \sum_r p(Q_t = q|Q_{t-1} = r) \alpha_{t-1}(r) \]
Trellis - synchronous search (Viterbi decoding)

Advantages

- overall efficiency
- simplicity
- performance
Trellis - asynchronous search (Stack decoding)

better for short and fat search problems
DGM Inference

performing inference

- compute marginal probabilities for cliques
- deduce inference algorithm to template $\mathcal{F}(G, \mathcal{M}, 0)$
unroll and compute

- 4 variables per frame (letters)
- 12 frames (numbers)

$48! \approx 10^{61}$ possible elimination orders
unroll and compute (2)

left-to-right (max. clique size: 5)
- A(1), D(1), B(1), C(1), A(2), D(2), B(2), C(2), A(3), ...

right-to-left (max. clique size: 5)
- A(12), C(12), B(12), D(12), A(11), C(11), B(11), ...
unroll and compute (3)

Smallest maximum clique size: 4

- C(1), A(1), B(1), E(1), D(1), C(2), A(2), F(1), B(2), E(2), D(2), C(3), A(3), F(2), ...

[Diagram of a network with nodes labeled A, B, C, D, E, F and numbers 1, 2, 3, 4, 5, 6, 7, 8, showing the structure of the network with edges connecting different nodes.]
finding periodicity

modify chunk

- Elimination order
  - slice by slice
    - left-to-right
    - right-to-left
  - general

- set Interface separator
finding periodicity: left-to-right

memory complexity: \( O(N^3 T) \)
finding periodicity: right-to-left

memory complexity: $O(N^3 T)$
finding periodicity: general

memory complexity: $O(N^2 T)$
finding periodicity: one clique

memory complexity: $O(N^2 T)$
Clique-based limited extent asynchrony

Clique no longer minimal

\[ \phi(b_4, d_4) = \sum_{b_3, c_3, d_3, a_4, b_4, d_4} \phi(b_3, d_3) \psi(b_3, c_3, d_3) \psi(c_3, d_3, d_4) \times \psi(b_3, c_3, a_4, b_4, d_4) \]
Clique-based limited extent asynchrony (2)

time dependencies

▷ $b_3, c_3, d_3 \rightarrow a_4, b_4, d_4$

Solution

▷ expand clique into multiple time slices

$\Rightarrow$ speed up
Outline

Introduction

Static Graphical Models

Dynamic Graphical Models

DGM for Speech Recognition

Conclusion
GMTK Dynamic Template

extend DGM

- allow backward directed links
- chunks span multiple time frame
- built-in specification mechanism for switching parents
- parent of variable may be multiple chunks (to past or future)
- allow multiframe structure at the beginning and the end
GMTK Dynamic Template (2)

A GMTK template extends a standard DBN template in five distinct ways. First, it allows for not only forward but also backward speech recognition and natural language processing. Before exploring various ASR constructs using graphical models, it is important to understand what a GMTK template looks like. GMTK templates are allowed to span multiple time points, and the above figure shows this. Of course, the knowledge sources, or both. In either case, information about a process [3] on the training data to dependencies over high-level domains is visually and intuitively portrayed. GMTK templates are not necessarily always, cause a significant increase in computational cost. Therefore, one must take care when adding variables to a DBN that constraints. Moreover, it is during training (when the amount of training data might be limited) that one wants to reduce the optimal solution within the parameter space subject to these constraints. When training parameters, however, we must find the constraints. When training parameters, however, ignores the factorization constraint expressed by the above, other factors in addition to distribution of effects such as coarticulation in human speech (see Figure 2).

A GMTK template extends a standard DBN template in five distinct ways. First, it allows for not only forward but also backward speech recognition and natural language processing. Before exploring various ASR constructs using graphical models, it is important to understand what a GMTK template looks like. GMTK templates are allowed to span multiple time points, and the above figure shows this. Of course, the knowledge sources, or both. In either case, information about a process [3] on the training data to dependencies over high-level domains is visually and intuitively portrayed. GMTK templates are not necessarily always, cause a significant increase in computational cost. Therefore, one must take care when adding variables to a DBN that constraints. Moreover, it is during training (when the amount of training data might be limited) that one wants to reduce the optimal solution within the parameter space subject to these constraints. When training parameters, however, we must find the constraints. When training parameters, however, ignores the factorization constraint expressed by the above, other factors in addition to distribution of effects such as coarticulation in human speech (see Figure 2).
GM for Automatic Speech Recognition (ASR)

Goals

- explicitly and efficiently represent ASR control constructs
- latent knowledge modeling
- proper observation modeling
- automatic structure learning
Basic Phone-based Bigram decoding structure

To compute the probability of the observations, we must perform the summation over all possible phone sequences that are consistent with the input. This is because the HMM is a generative model, and any sequence of phones that is consistent with the input is a possible path through the model. The probability of a particular path through the model is given by the product of the probabilities of the individual CPFs. This is because the CPFs are independent, and the probability of a sequence of independent events is the product of the probabilities of the individual events.

In the next several sections, we will see how these four goals arise when we examine a number of different graphs used to represent the speech recognition process. We will start with a 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 zero value since one or more of the factors at each time frame will themselves be zero. This is because many of the CPFs at each time frame are unknown, and the sum of unknown CPFs is zero. However, we can still exploit the factorization property as specified by the CPFs.
Trigram language model decoder

\[ W^{pr}: \text{Previous-Word} \]
\[ W: \text{Word} \]
\[ W^{tr}: \text{Word-Transition} \]
\[ W^{ps}: \text{Word-Position} \]
\[ P^{tr}: \text{Phone-Transition} \]
\[ P: \text{Phone} \]
\[ O: \text{Acoustic Features} \]

\( W_{t+1} \) gets a copy of the previous current word \( W_t \) with probability one. Moreover, the new current word \( W_{t+1} \) is chosen with the trigram probability, but it conditions on the previous current word \( W_t \) and the previous previous word \( W^{pr} \), as needed by a trigram.

CROSS-WORD TRIPHONE ARCHITECTURE

Another technique that is typically employed by SRSs is that of cross-word triphones \([46]\). Triphone models are those where the acoustic observation is not only conditioned on the currently hypothesized phone, but it also makes the assumption that the current acoustics are significantly influenced by the preceding and following phonetic context (i.e., coarticulation). Triphone models accomplish this by saying that the distribution over the acoustic frame depends on the current, previous, and next phone.
Cross-word triphone decoder

word $W_{pr}$ gets a copy of the previous current word $W_t$ with probability one. Moreover, the new current word $W_{t+1}$ is chosen with the trigram probability, but it conditions on the previous current word $W_t$ and the previous previous word $W_{pr}$, as needed by a trigram.

CROSS-WORD TRIPHONE ARCHITECTURE

Another technique that is typically employed by SRSs is that of cross-word triphones \[46\]. Triphone models are those where the acoustic observation is not only conditioned on the currently hypothesized phone, but it also makes the assumption that the current acoustics are significantly influenced by the preceding and following phonetic context (i.e., coarticulation). Triphone models accomplish 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 backward time edges. The solution is shown in Figure 5. The top part of the graph shows the graph construct for the trigram 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 $P_{pr}$ and a next phone variable $P_{nx}$, that, when combined together with the phone $P_t$, produce the triphone variable $P_{x3}$.

When a phone transition does not occur, $P_{tr} = 0$, the phone variable keeps its same value $P_{t+1} = P_t$ from frame to frame (since neither the word changes, $W_{t+1} = W_t$, nor does the position increment, $W_{ps} = W_{ps}$), the next previous phone retains its value $P_{pr+1} = P_{pr}$, and the current next phone simply copies its future value from $P_{nx+1}$.

When a phone transition does occur ($P_{tr} = 1$), then the current phone $P_t$ gets copied to the next previous phone $P_{pr+1}$, a new phone $P_{t+1}$ is chosen based on the incremented new word position $W_{ps}$, and that new phone is then copied backwards in time into the current next phone, i.e., $P_{nx} = P_{t+1}$ via the backward time edge for use at time $t$. The backward time edges have thus represented anticipatory coarticulation effects in the speech process. Note, however (and similar to the trigram language model case),
Transition-specific decoder

[Diagram of a transition-specific decoder]
Generic multistream, semi-asynchronous decoder
Transition and phone-dep. buried Markov model decoder

The word variable $W_t$ is at the top of the structure. In the

... FORMAL BUT

WIDELY FLEXIBLE MEANS FOR

SOLVING MANY OF THE

PROBLEMS IN SPEECH

AND LANGUAGE PROCESSING.

...
Outline

Introduction

Static Graphical Models

Dynamic Graphical Models

DGM for Speech Recognition

Conclusion
Conclusion

We talked about

- Graphs
- Templates
- Decoding
- Computing
- Examples for ASR
"A picture is worth a thousand words."

Thank you for your attention!
References

J. Bilmes,
“Dynamic Graphical Models,”
*IEEE Signal Processing Magazine*, vol. 27 (6), pp. 29–42, Nov 2010

J. Bilmes, C. Bartels,
“Graphical Model Architectures for Speech Recognition,”
*IEEE Signal Processing Magazine*, vol. 22 (5), pp. 89–100, Sept 2005

C.M. Bishop,
“Pattern Recognition and Machine Learning,”