Graphical Models (II)

Inference

Eric Xing
Carnegie Mellon University
June 1, 2007

Recap of Basic Prob. Concepts

- Joint probability dist. on multiple variables:
  \[ P(X_1, X_2, X_3, X_4, X_5, X_6) = P(X_1)P(X_2 | X_1)P(X_3 | X_1, X_2)P(X_4 | X_1, X_2, X_3)P(X_5 | X_1, X_2, X_3, X_4)P(X_6 | X_1, X_2, X_3, X_4, X_5) \]

- If \( X_i \)'s are independent: \( P(X_i | \cdot) = P(X_i) \)
  \[ P(X_1, X_2, X_3, X_4, X_5, X_6) = P(X_1)P(X_2)P(X_3)P(X_4)P(X_5)P(X_6) = \prod_i P(X_i) \]

- If \( X_i \)'s are conditionally independent (as described by a GM), the joint can be factored to simpler products, e.g.,
  \[ P(X_1, X_2, X_3, X_4, X_5, X_6) = P(X_1)P(X_2 | X_3)P(X_3 | X_1)P(X_4 | X_3)P(X_5 | X_3)P(X_6 | X_4, X_5) \]
Inference and Learning

- We now have compact representations of probability distributions: **Graphical Models**

- A GM $M$ describes a unique probability distribution $P$

- Typical tasks:
  - Task 1: How do we answer queries about $P$?
    - We use **inference** as a name for the process of computing answers to such queries
  - Task 2: How do we estimate a plausible model $M$ from data $D$?
    - We use **learning** as a name for the process of obtaining point estimate of $M$.
    - But for **Bayesian**, they seek $p(M|D)$, which is actually an **inference** problem.
    - When not all variables are observable, even computing point estimate of $M$ need to do **inference** to impute the missing data.

Inferential Query 1: Likelihood

- Most of the queries one may ask involve **evidence**

  - Evidence $x_v$ is an assignment of values to a set $X_v$ of nodes in the GM over variable set $X=\{X_1, X_2, \ldots, X_n\}$
  - Without loss of generality $X_v=\{X_{v+1}, \ldots, X_n\}$.
  - Write $X_H=X\setminus X_v$ as the set of hidden variables, $X_H$ can be $\emptyset$ or $X$

- Simplest query: compute probability of evidence

  $$P(x_v) = \sum_{X_H} P(X_H, x_v) = \sum_{x_1} \ldots \sum_{x_{v-1}} P(x_1, \ldots, x_{v-1}, x_v)$$

  - this is often referred to as computing the **likelihood** of $x_v$
Inferential Query 2: Conditional Probability

- Often we are interested in the **conditional probability distribution** of a variable given the evidence

\[
P(X_{H} \mid X_{V} = x_{v}) = \frac{P(X_{H}, x_{V})}{P(x_{V})} = \frac{P(X_{H}, x_{V})}{\sum_{x_{H}} P(X_{H}, x_{H}, x_{V})}
\]

- this is the **a posteriori belief** in \(X_{H}\), given evidence \(x_{v}\)

- We usually query a subset \(Y\) of all hidden variables \(X_{H} = \{Y, Z\}\) and "don’t care" about the remaining, \(Z\):

\[
P(Y \mid x_{v}) = \sum_{z} P(Y, Z = z \mid x_{v})
\]

- the process of summing out the "don’t care" variables \(z\) is called **marginalization**, and the resulting \(P(Y \mid x_{v})\) is called a **marginal prob.**

Applications of **a posteriori** Belief

- **Prediction**: what is the probability of an outcome given the starting condition

- the query node is a descendent of the evidence

- **Diagnosis**: what is the probability of disease/fault given symptoms

- the query node an ancestor of the evidence

- **Learning** under partial observation

  - fill in the unobserved values under an "EM" setting (more later)

- The directionality of information flow between variables is not restricted by the directionality of the edges in a GM

  - probabilistic inference can combine evidence form all parts of the network
In this query we want to find the most probable joint assignment (MPA) for some variables of interest.

Such reasoning is usually performed under some given evidence $x_v$, and ignoring (the values of) other variables $Z$:

$$Y^* | x_v = \arg \max_y P(Y | x_v) = \arg \max_y \sum_z P(Y, Z = z | x_v)$$

- this is the maximum a posteriori configuration of $Y$.

### Applications of MPA

- Classification
  - find most likely label, given the evidence
- Explanation
  - what is the most likely scenario, given the evidence

Cautionary note:

- The MPA of a variable depends on its "context"---the set of variables been jointly queried

**Example:**
- MPA of $X$?
- MPA of $(X, Y)$?

<table>
<thead>
<tr>
<th>$x$</th>
<th>$y$</th>
<th>$P(x, y)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0.35</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0.05</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0.3</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0.3</td>
</tr>
</tbody>
</table>
Complexity of Inference

**Thm:**
Computing $P(X_h=x_h|X_v)$ in an arbitrary GM is NP-hard

- Hardness does not mean we cannot solve inference
  - It implies that we cannot find a general procedure that works efficiently for arbitrary GMs
  - For particular families of GMs, we can have provably efficient procedures

Approaches to inference

- **Exact inference algorithms**
  - The sum-product algorithm
  - The junction tree algorithm

- **Approximate inference techniques**
  - Stochastic simulation / sampling methods
  - Markov chain Monte Carlo methods
  - Variational algorithms (later lectures)
The Junction Tree Algorithm

- There are several inference algorithms; some of which operate directly on (special) directed graph
  - Forward-backward algorithm for HMM (we will see it later)
  - Pealing algorithm for trees and phylogenies
- The junction tree algorithm is the most popular and general inference algorithm, it operates on an undirected graph
  - To understand the JT-algorithm, we need to understand how to compile a directed graph into an undirected graph

Moral Graph

- Note that for both directed GMs and undirected GMs, the joint probability is in a product form:
  - **BN**: \( P(X) = \prod_{i \in \text{Ed}} P(X_i | X_{\neg i}) \)
  - **MRF**: \( P(X) = \frac{1}{Z} \prod_{c \in C} \psi_c(x_c) \)
- So let’s convert local conditional probabilities into potentials; then the second expression will be generic, but how does this operation affect the directed graph?
  - We can think of a conditional probability, e.g., \( P(C|A,B) \) as a function of the three variables \( A, B, \) and \( C \) (we get a real number of each configuration):
    - Problem: But a node and its parent are not generally in the same clique in a BN
    - Solution: Marry the parents to obtain the “moral graph”
Moral Graph (cont.)

- Define the potential on a clique as the product over all conditional probabilities contained within the clique.
- Now the product of potentials gives the right answer:

\[ P(X_1, X_2, X_3, X_4, X_5, X_6) = P(X_1)P(X_2)P(X_3 | X_1, X_2)P(X_4 | X_3)P(X_5 | X_4, X_5)P(X_6 | X_4, X_6) \]

\[ \psi(X_1, X_2, X_3) = P(X_1)P(X_2)P(X_3 | X_2) \]
\[ \psi(X_3, X_4, X_5) = P(X_4)P(X_5 | X_3) \]
\[ \psi(X_4, X_5, X_6) = P(X_6 | X_4, X_5) \]

Note that here the interpretation of potential is ambivalent: it can be either marginals or conditionals.

Clique trees

- A clique tree is an (undirected) tree of cliques.

- Consider cases in which two neighboring cliques \( V \) and \( W \) have an overlap \( S \) (e.g., \( (X_1, X_2, X_3) \) overlaps with \( (X_3, X_4, X_5) \)),

\[ \psi(V) \quad \psi(S) \quad \psi(W) \]

- Now we have an alternative representation of the joint in terms of the potentials:
Clique trees

- A clique tree is an (undirected) tree of cliques.

- The alternative representation of the joint in terms of the potentials:

\[
P(X_1, X_2, X_3, X_4, X_5, X_6) = P(X_1)P(X_2)P(X_4 | X_2)P(X_4 | X_3)P(X_6 | X_2, X_3)
\]

\[
P(X_1, X_2, X_3, X_4, X_5, X_6) = \frac{P(X_1, X_2, X_3)P(X_4, X_5, X_6)}{P(X_4, X_5)}
\]

\[
= \psi(X_1, X_2, X_3) \frac{\psi(X_4, X_5, X_6)}{\phi(X_5)}
\]

- Generally:

\[
P(X) = \prod \psi_i(X_i) \prod \phi_j(X_j)
\]

Now each potential is isomorphic to the cluster marginal of the attendant set of variables.

Why this is useful?

- Propagation of probabilities

  - Now suppose that some evidence has been "absorbed" (i.e., certain values of some nodes have been observed). How do we propagate this effect to the rest of the graph?

  - What do we mean by propagate?

Can we adjust all the potentials \( \{\psi\}, \{\phi\} \) so that they still represent the correct cluster marginals (or unnormalized equivalents) of their respective attendant variables?

- Utility?

\[
P(X_i | X_k = x_k) = \sum_{x_{i \neq k}} \psi(X_i, X_k = x_k)
\]

\[
P(X_4 | X_5 = x_5) = \phi(X_4)
\]

\[
P(X_6) = \sum_{x_5} \psi(X_4, X_6 | x_5)
\]

Local operations!
Local Consistency

- We have two ways of obtaining $p(S)$
  
  $P(S) = \sum_{V} \psi(V)$  \hspace{1cm}  $P(S) = \sum_{W} \psi(W)$

  and they must be the same

- The following update-rule ensures this:
  
  Forward update: $\phi^*_S = \sum_{V} \psi'_V$ \hspace{1cm} $\psi'_W = \frac{\phi^*_S}{\phi^*_S}$

  Backward update: $\phi^*_W = \sum_{W} \psi''_W$ \hspace{1cm} $\psi''_V = \frac{\phi^*_W}{\phi^*_W}$

- Two important identities can be proven
  
  $\sum_{V} \psi'_V = \sum_{W} \psi''_W = \phi^*_S$ \hspace{1cm} $\psi'_V \psi''_W = \psi''_V \psi'_W = \psi'_V \psi''_W

  Local Consistency Invariant Joint

Message Passing Algorithm

- This simple local message-passing algorithm on a clique tree defines the general probability propagation algorithm for directed graphs!

  Many interesting algorithms are special cases:
  
  - Forward-backward algorithm for hidden Markov models,
  - Kalman filter updates
  - Peeling algorithms for probabilistic trees

  The algorithm seems reasonable. Is it correct?
A problem

- Consider the following graph and a corresponding clique tree

```
A --|-- B
  |    |
  C   D
```

```
A,B --|-- B,D
     |    |
     A,C --|-- C,D
```

- Note that C appears in two non-neighboring cliques
- Question: with the previous message passage, can we ensure that the probability associated with C in these two (non-neighboring) cliques consistent?
- Answer: No. It is not true that in general local consistency implies global consistency
- What else do we need to get such a guarantee?

Triangulation

- A triangulated graph is one in which no cycles with four or more nodes exist in which there is no chord

```
A --|-- B
  |    |
  C   D
```

- We triangulate a graph by adding chords:

```
A --|-- B
  |    |
  C   D
```

- Now we no longer have our global inconsistency problem.

```
A --|-- B
  |    |
  C   D
```

- A clique tree for a triangulated graph has the running intersection property: If a node appears in two cliques, it appears everywhere on the path between the cliques
- Thus local consistency implies global consistency
**Junction trees**

- A clique tree for a triangulated graph is referred to as a *junction tree*.
- In junction trees, local consistency implies global consistency. Thus the local message-passing algorithms is (provably) correct.
- It is also possible to show that only triangulated graphs have the property that their clique trees are junctions. Thus if we want local algorithms, we *must* triangulate.
- Are we now all set?
  - How to triangulate?
  - The complexity of building a JT depends on how we triangulate!!
  - Consider this network: it turns out that we will need to pay an $O(2^4)$ or $O(2^6)$ cost depending on how we triangulate!

**How to triangulate**

- A graph elimination algorithm
  - Intermediate terms correspond to the *cliques* resulted from elimination
  - "good" elimination orderings lead to *small cliques* and hence reduce complexity (what will happen if we eliminate "e" first in the above graph?)
  - finding the optimum ordering is NP-hard, but for many graph optimum or near-optimum can often be heuristically found
From Elimination to Message Passing

- Our algorithm so far answers only one query (e.g., on one node), do we need to do a complete elimination for every such query?
- Elimination $\equiv$ message passing on a clique tree

- Messages can be reused

Recall this:

$$\phi_j = \sum_{i:j} \psi_i$$
$$\psi_{i'} = \frac{\phi_i}{\phi_{i'}}$$

From Elimination to Message Passing

- Our algorithm so far answers only one query (e.g., on one node), do we need to do a complete elimination for every such query?
- Elimination $\equiv$ message passing on a clique tree
  - Another query ...

- Messages $m_f$ and $m_g$ are reused, others need to be recomputed
Message-passing algorithms

- **Message update**
  - The Hugin update
  - The Shafer-Shenoy update

\[
\phi^*_i = \sum_{\psi_{\bar{V}}} \psi^*_i = \frac{\phi^*_i}{\phi^*_\psi} \psi \quad \psi = \phi^*_i
\]

\[
m_{i \rightarrow j}(S_j) = \sum_{c \in \bar{S}_j} \prod_{i \in j} m_{i \rightarrow j}(S_i)
\]

A Sketch of the Junction Tree Algorithm

- **The algorithm**
  1. Moralize the graph (trivial)
  2. Triangulate the graph (good heuristic exist, but actually NP hard)
  3. Build a clique tree (e.g., using a maximum spanning tree algorithm)
  4. Propagation of probabilities --- a local message-passing protocol

- Results in marginal probabilities of all cliques --- solves all queries in a single run
- A **generic** exact inference algorithm for any GM
- **Complexity**: exponential in the size of the maximal clique --- a good elimination order often leads to small maximal clique, and hence a good (i.e., thin) JT
Case study:

- Hidden Markov Model

Recall definition of HMM

- Transition probabilities between any two states
  \[ p(y'_i = 1 \mid y'_{i-1} = 1) = a_{i,j}, \]
  or
  \[ p(y_i \mid y'_{i-1} = 1) \sim \text{Multinomial}(a_{1,1}, a_{1,2}, \ldots, a_{1,M}) \forall i \in 1. \]

- Start probabilities
  \[ p(y_1) \sim \text{Multinomial}(\pi_1, \pi_2, \ldots, \pi_M) \]

- Emission probabilities associated with each state
  \[ p(x_i \mid y'_i = 1) \sim \text{Multinomial}(b_{1,1}, b_{1,2}, \ldots, b_{1,K}) \forall i \in 1. \]
  or in general:
  \[ p(x_i \mid y'_i = 1) \sim f(\cdot | \theta), \forall i \in 1. \]
Probability of a parse

- Given a sequence $x = x_1, \ldots, x_T$ and a parse $y = y_1, \ldots, y_T$,
- To find how likely is the parse:
  (given our HMM and the sequence)

$$p(x, y) = p(x_1, \ldots, x_T, y_1, \ldots, y_T) \quad \text{(Joint probability)}$$

$$= p(y_1) p(x_1 | y_1) p(x_2 | y_1) p(x_3 | y_1) \ldots p(x_T | y_1)$$

$$= p(y_1) p(y_2 | y_1) p(x_2 | y_2) p(x_3 | y_2) \ldots p(x_T | y_T)$$

$$= p(y_1, \ldots, y_T) p(x_1, \ldots, x_T | y_1, \ldots, y_T)$$

- Marginal probability: $p(x) = \sum_y p(x, y) = \sum_{y_1} \sum_{y_2} \ldots \sum_{y_T} p(x_1) p(y_1) \prod_{i=2}^T p(y_i | y_{i-1}) p(x_i | y_i)$
- Posterior probability: $p(y | x) = p(x, y) / p(x)$

Three main questions on HMMs

1. Evaluation
   GIVEN an HMM $M$, and a sequence $x$,
   FIND $\text{Prob} (x | M)$
   ALGO. Forward

2. Decoding
   GIVEN an HMM $M$, and a sequence $x$,
   FIND the sequence $y$ of states that maximizes, e.g., $P(y | x, M)$, or the most probable subsequence of states
   ALGO. Viterbi, Forward-backward

3. Learning (next lecture)
   GIVEN an HMM $M$, with unspecified transition/emission probs., and a sequence $x$,
   FIND parameters $\theta = (\pi, a, \eta)$ that maximize $P(x | \theta)$
   ALGO. Baum-Welch (EM)
The Forward Algorithm

- We want to calculate $P(x)$, the likelihood of $x$, given the HMM
  - Sum over all possible ways of generating $x$:
    \[
    p(x) = \sum_y p(x, y) = \sum_i \sum_j \ldots \sum_y \pi_1 \prod_{t=2}^{T} a_{y_{t-1}y_t} \prod_{t=1}^{T} p(x_t | y_t)
    \]
  - To avoid summing over an exponential number of paths $y$, define
    \[
    \alpha(y^k_t) = \alpha^k_t \overset{\text{def}}{=} P(x_1, \ldots, x_t, y^k_t = 1)
    \]
    (the forward probability)
  - The recursion:
    \[
    \alpha^k_t = p(x_t | y^k_t = 1) \sum_{i} \alpha^i_{t-1} a_{i,k}
    \]
    \[
    P(x) = \sum_k \alpha^k_T
    \]

The Backward Algorithm

- We want to compute $P(y^k_t = 1 | x)$, the posterior probability distribution on the $t$th position, given $x$
  - We start by computing
    \[
    P(y^k_t = 1, x) = P(x_1, \ldots, x_T, y^k_t = 1, x_{t+1}, \ldots, x_T)
    \]
    \[
    = P(x_1, \ldots, x_t, y^k_t = 1) P(x_{t+1}, \ldots, x_T | x_1, \ldots, x_t, y^k_t = 1)
    \]
    \[
    = P(x_1, \ldots, x_t, y^k_t = 1) P(x_{t+1}, \ldots, x_T | y^k_t = 1)
    \]
  - The recursion:
    \[
    \beta^k_t = \sum_i a_{k,i} p(x_{t+1} | y^i_{t+1} = 1) \beta^i_{t+1}
    \]
Shafer Shenoy for HMMs

- Recap: Shafer-Shenoy algorithm

- Message from clique $i$ to clique $j$:

$$\mu_{i\rightarrow j} = \sum_{C_j|S_{ij}} \psi_{C_j} \prod_{k \in j} \mu_{k \rightarrow i}(S_{ik})$$

- Clique marginal

$$p(C_i) \propto \psi_{C_i} \prod_{k} \mu_{k \rightarrow i}(S_{ik})$$

Shafer Shenoy for HMMs (cont.)

- A junction tree for the HMM

- Rightward pass

$$\mu_{t \rightarrow t+1}(y_{t+1}) = \sum_{y_{t-1}} \psi(y_{t-1}, y_t) \mu_{t-1 \rightarrow t}(y_t) \mu_{t \rightarrow t+1}(y_{t+1})$$

$$= \sum_{y_{t-1}} p(y_{t-1} | y_t) \mu_{t-1 \rightarrow t}(y_t) p(x_{t+1} | y_{t+1})$$

$$= p(x_{t+1} | y_{t+1}) \sum_{y_{t-1}} \psi(y_{t-1}, y_t) \mu_{t-1 \rightarrow t}(y_t)$$

- This is exactly the forward algorithm!

- Leftward pass ...

$$\mu_{t \rightarrow t-1}(y_{t-1}) = \sum_{y_{t+1}} \psi(y_{t+1}, y_t) \mu_{t \rightarrow t+1}(y_{t+1}) \mu_{t \rightarrow t-1}(y_{t-1})$$

$$= \sum_{y_{t+1}} p(y_{t+1} | y_t) \mu_{t \rightarrow t+1}(y_{t+1}) p(x_{t-1} | y_{t-1})$$

- This is exactly the backward algorithm!
Approaches to inference

- Exact inference algorithms
  - The elimination algorithm
  - The junction tree algorithms

- Approximate inference techniques
  - Stochastic simulation / sampling methods
  - Markov chain Monte Carlo methods
  - Variational algorithms (later lectures)

The motif detection problem

Biological background: the transcriptional regulatory machinery

- gene regulatory sequences
- spacer DNA
- gene regulatory proteins
- TATA box
- promoter
- RNA polymerase
- general transcription factors
- start of transcription
**In silico motif detection**

\[
\begin{align*}
5' & \quad - \quad \text{TCTCTCTCCACGGCTAATTAGGTGATCATGAAAAAATGAAAAATTCATGAG} \\
5' & \quad - \quad \text{AAATGACTCA} \\
5' & \quad - \quad \text{TGCGAAC} \\
5' & \quad - \quad \text{GACATCGAAACATACAT} \\
5' & \quad - \quad \text{ACG} \\
5' & \quad - \quad \text{CACATCCAACGAATCACCTCACCGTTATCG} \\
5' & \quad - \quad \text{TTACAACGAGGAAATAGAAGAAAATGAAAAATTTTCGACAAAATGTATAGTCATTTCTATC} \\
5' & \quad - \quad \text{GAGATGAGTCA} \\
5' & \quad - \quad \text{AAATGAGTCA} \\
5' & \quad - \quad \text{AGATGAGTCA} \\
5' & \quad - \quad \text{AAAAGAGTCA} \\
5' & \quad - \quad \text{CCGCGCCACAGTCCGCGTTTGGTTATCCGGC} \\
5' & \quad - \quad \text{TTGGAAAGTGTGGCATGTGCTTCACACA} \\
5' & \quad - \quad \text{AAAAGAGTCA} \\
5' & \quad - \quad \text{AAAAGAGTCA} \\
5' & \quad - \quad \text{AAAAGAGTCA} \\
5' & \quad - \quad \text{AAAAGAGTCA} \\
5' & \quad - \quad \text{AAAAGAGTCA} \\
5' & \quad - \quad \text{AAAAGAGTCA} \\
5' & \quad - \quad \text{AAAAGAGTCA} \\
5' & \quad - \quad \text{AAAAGAGTCA} \\
\end{align*}
\]

Multiple alignment: \( A = \)

\[
\begin{align*}
1: & \quad \text{AAAAGAGTCA} \\
2: & \quad \text{AAAAGAGTCA} \\
\vdots
\end{align*}
\]

Locations: \( \{X\} \)

Background

**A Generative Scheme**

\[
\begin{align*}
\{Y\} =
\end{align*}
\]

Eric Xing 37

Eric Xing 38
The LOGOS motif model

[Xing, Wu, Jordan and Karp, JBCB 2004]

- An integrated LOcal and GIObal motif Sequence model
  - The occurrences of motifs in DNA sequences are governed by a global motif syntax model $p_g(X)$
  - All instances of a specific motif admit a local motif prior model $p_l(A, \theta | X)$
  - Non-motif sequences are modeled by a background model $p_b(Y-A|X)$

The global hidden Markov model (HMM)

1st order Markov grammar
The background model

$k$-th order Markov background

The local prior model

Hidden Markov Dirichlet-multinomial (HMDM)

[Xing, Jordan, Karp and Russell, NIPS 2002]
A modular Bayesian model for motif detection

Joint posterior:

\[ p(X, S, \theta | y) = p(X | \theta, y) p(\theta, S) \]

Inference in LOGOS model

- Joint posterior:

\[ p(X, S, \theta | y) = p(X | \theta, y) p(\theta, S) \]

- Inference on motif locations

\[ p(x_i | y) = \sum_{\theta} \sum_s p(x | \theta, y) p(\theta, s) \]

- State space to be summed (and integrated) over

\[ \mathcal{R}^{4^6 \sum L_j x | \Omega_s | \sum L_j x | \Omega_s} \]

~ \( \mathcal{R}^{120} \times 10^{200} \) for a 1000bp sequence with two motif patterns of length 15bp

- Approximate inference
  - Stochastic approximation: Gibbs sampling
  - Deterministic approximation: Variational inference

\[ \sqrt{\ldots} \]
Variational Methods

- For a distribution \( p(X|\theta) \) associated with a complex graph, computing the marginal (or conditional) probability of arbitrary random variable(s) is intractable.

- Variational methods
  - Formulating probabilistic inference as an optimization problem:

\[
eq f^* = \arg\max_{f \in \mathcal{S}} \left\{ F(f) \right\}
\]

\( f^* \) is a (tractable) probability distribution or, solutions to certain probabilistic queries.

Exponential Family

- Exponential representation of graphical models:

\[
P(X) = \frac{1}{Z} \prod_{c \in C} \psi_c(X_c) \quad \Rightarrow \quad p(X|\theta) = \exp\left\{ \sum_{a} \theta_a \phi_a(X_{D_a}) - A(\theta) \right\}
\]

- Includes discrete models, Gaussian, Poisson, exponential, and many others

\[
E(X) = -\sum_{a} \theta_a \phi_a(X_{D_a}) \text{ is referred to as the energy of state } x
\]

\[
\Rightarrow \quad p(X|\theta) = \exp\left\{ -E(X) - A(\theta) \right\} = \exp\left\{ -E(X_H, x_E) - A(\theta, x_E) \right\}
\]
Example: the Boltzmann distribution on atomic lattice

\[ p(X) = \frac{1}{Z} \exp \left\{ \sum_{i<j} \theta_{ij} X_i X_j + \sum_i \theta_{i0} X_i \right\} \]

Lower bounds of exponential functions

\[ \exp(x) \geq \exp(\mu)(x - \mu + 1) \]
\[ \exp(x) \geq \frac{1}{6} \exp(\mu)(x - \mu)^3 + 3(x - \mu)^2 + 6(x - \mu + 1) \]
Lemma: Every marginal distribution $q(X_H)$ defines a lower bound of likelihood:

$$p(x_E) \geq \int d_x^H \exp\{-E'(x_H)\}$$

$$(1 - A(x_E) - (E(x_H, x_E) - E'(x_H))),$$

where $x_E$ denotes observed variables (evidence).

**Upgradeable to higher order bound** [Leisink and Kappen, 2000]

Lower bounding likelihood

Representing $q(X_H)$ by exp\{-$E'(X_H)$\}:

Lemma: Every marginal distribution $q(X_H)$ defines a lower bound of likelihood:

$$p(x_E) \geq C - \langle E(X_H, x_E) \rangle_{q(x_H)} + \int d_x^H q(x_H) \log q(x_H)$$

$$= C - \langle E \rangle_q - H_q,$$

where $x_E$ denotes observed variables (evidence).

$\langle E \rangle_q$: expected energy  \hspace{1cm} $\langle E \rangle_q + H_q$: Gibbs free energy

$H_q$: entropy
KL and variational (Gibbs) free energy

- Kullback-Leibler Distance:
\[ KL(q \parallel p) = \sum z q(z) \ln \frac{q(z)}{p(z)} \]

- "Boltzmann's Law" (definition of "energy"):
\[ p(z) = \frac{1}{C} \exp[-E(z)] \]

\[ KL(q \parallel p) \equiv \sum z q(z) E(z) + \sum z q(z) \ln q(z) + \ln C \]

Gibbs Free Energy \( G(q) \); minimized when \( q(Z) = p(Z) \)

KL and Log Likelihood

- Jensen's inequality
\[ \mathcal{L}(\theta;x) = \log p(x | \theta) \]
\[ = \log \sum_x p(x, z | \theta) \]
\[ = \log \sum_x q(z | x) \frac{p(x, z | \theta)}{q(z | x)} \]
\[ \geq \sum_x q(z | x) \log \frac{p(x, z | \theta)}{q(z | x)} \]

\[ \Rightarrow \mathcal{L}(\theta;x) \geq \left\{ \mathcal{L}(\theta;x, z) \right\}_q + H_q = \mathcal{L}(q) \]

- KL and Lower bound of likelihood
\[ \mathcal{L}(\theta;x) = \log p(x | \theta) = \sum_x q(z | x) \log \frac{p(x, z | \theta)}{q(z)} \]
\[ = \sum_x q(z | x) \log \frac{p(x, z | \theta)}{q(z)} - \sum_x q(z | x) \log \frac{q(z)}{p(z | x, \theta)} \]
\[ = \sum_x q(z | x) \log \frac{p(x, z | \theta)}{q(z)} + \sum_x q(z | x) \log \frac{q(z)}{p(z | x, \theta)} \]

\[ \Rightarrow \mathcal{L}(\theta;x) = \mathcal{L}(q) + KL(q \parallel p) \]

- Setting \( q() = p(z|x) \) closes the gap (c.f. EM)
A variational representation of probability distributions

\[ q = \arg \max_{q \in Q} \left\{ - \langle E \rangle_q - H_q \right\} \]
\[ = \arg \min_{q \in Q} \left\{ \langle E \rangle_q + H_q \right\} \]

where \( Q \) is the equivalent sets of realizable distributions, e.g., all valid parameterizations of exponential family distributions, marginal polytopes [Winright et al. 2003].

Difficulty: \( H_q \) is intractable for general \( q \)

"solution": approximate \( H_q \), and/or, relax or tighten \( Q \)

Mean field methods

- Optimize \( q(X_H) \) in the space of tractable families

  - *i.e.*, subgraph of \( G_p \) over which exact computation of \( H_q \) is feasible

- Tightening the optimization space

  - exact objective: \( H_q \)
  - tightened feasible set: \( Q \rightarrow T \quad (T \subseteq Q) \)

\[ q^* = \arg \min_{q \in T} \langle E \rangle_q + H_q \]
Belief Propagation

- Do not optimize $q(X_H)$ explicitly, but focus on the set of beliefs
  - e.g., $b = \{ b_{i,j} = \tau(X_i, X_j), \ b_i = \tau(X_i) \}$

- Relax the optimization problem
  - approximate objective: $H_{\text{belief}} = H(b_{i,j}, b_i)$
  - relaxed feasible set: $\mathcal{M}_0 = \{ \tau \geq 0 | \sum_{X_i} \tau(X_i) = 1, \sum_{X_j} \tau(X_i, X_j) = \tau(X_j) \}$
  - $b^* = \arg\min_{b \in \mathcal{M}_0} \{ \langle E \rangle_b + F(b) \}$

- The loopy BP algorithm:
  - a fixed point iteration procedure that tries to solve $b^*$

Mean Field Approximation
Cluster-based approx. to the Gibbs free energy

Exact: \( G[q(X)] \) (intractable)

Clusters: \( G[\{q_c(X_c)\}] \)

Mean field approx. to Gibbs free energy

- Given a disjoint clustering, \( \{C_1, \ldots, C_I\} \), of all variables
- Let \( q(X) = \prod_i q_i(X_{C_i}) \)
- Mean-field free energy
  \[
  G_{MF} = \sum_{i} \sum_{x_{C_i}} \prod_i q_i(x_{C_i}) E(x) + \sum_i \sum_{x_{C_i}} q_i(x_{C_i}) \ln q_i(x_{C_i})
  \]
  e.g., \( G_{MF} = \sum_{i} \sum_{x_{C_i}} q_i(x, y(x, x_i)) + \sum_i \sum_{x_{C_i}} q_i(x) \ln q_i(x, y(x, x_i)) \) (naïve mean field)

- Will never equal to the exact Gibbs free energy no matter what clustering is used, but it does always define a lower bound of the likelihood
- Optimize each \( q_i(x) \)'s.
  - Variational calculus ...
  - Do inference in each \( q_i(x) \) using any tractable algorithm
Theorem: The optimum GMF approximation to the cluster marginal is isomorphic to the cluster posterior of the original distribution given internal evidence and its generalized mean fields:

\[ q_i^* (X_{H,C_i}) = p(X_{H,C_i} | x_{E,C_i}, \langle X_{H,MB_i} \rangle_{q_{rel}}) \]

GMF algorithm: Iterate over each \( q_i \)

---

A generalized mean field algorithm

[xing et al. UAI 2003]
A generalized mean field algorithm

[xing et al. UAI 2003]

Theorem: The GMF algorithm is guaranteed to converge to a local optimum, and provides a lower bound for the likelihood of evidence (or partition function) the model.

Convergence theorem
The naive mean field approximation

- Approximate $p(X)$ by fully factorized $q(X) = P_q(X)$
- For Boltzmann distribution $p(X) = \exp \{ \sum_{i < j} q_{ij} X_i X_j + q_{ii} X_i \}/Z$:

  mean field equation:
  \[
  q_i(X_i) = \exp \left\{ \theta_0 X_i + \sum_{j \in N_i} \theta_{ij} X_i \left\langle X_j \right\rangle_{q_j} + A \right\}
  = p(X_i | \{ \left\langle X_j \right\rangle_{q_j} : j \in N_i \})
  \]

  - $\{ \left\langle X_j \right\rangle_{q_j} : j \in N_i \}$ resembles a “message” sent from node $j$ to $i$
  - $\{ \left\langle X_j \right\rangle_{q_j} : j \in N_i \}$ forms the “mean field” applied to $X_i$ from its neighborhood

Generalized MF approximation to Ising models

Cluster marginal of a square block $C_k$:

\[
q(X_{C_k}) \propto \exp \left\{ \sum_{i,j \in C_k} \theta_{ij} X_i X_j + \sum_{i \in C_k} \theta_{ii} X_i + \sum_{i < j : i \in C_k, j \in C_k} \theta_{ij} \left\langle X_i \right\rangle_{q(X_{C_k})} \right\}
\]

Virtually a reparameterized Ising model of small size.
GMF approximation to Ising models

Attractive coupling: positively weighted
Repulsive coupling: negatively weighted

Cluster-based MF (e.g., GMF)

- a general, iterative message passing algorithm
- clustering completely defines approximation
  - preserves dependencies
  - flexible performance/cost trade-off
  - clustering automatable
- recovers model-specific structured VI algorithms, including:
  - fHMM, LDA
  - variational Bayesian learning algorithms
- easily provides new structured VI approximations to complex models
Example 1: Latent Dirichlet Allocation

- Blei, Jordan and Ng (2003)
- Generative model of documents (but broadly applicable e.g. collaborative filtering, image retrieval, bioinformatics)
- Generative model:
  - choose
  - choose topic
  - choose word

\[ \theta \sim \text{Dir}(\alpha) \]
\[ z_n \sim \text{Mult}(\theta) \]
\[ w_n \sim p(w_n | z_n, \beta) \]

Latent Dirichlet Allocation

- Variational approximation

\[ q(\theta, z) = q_\theta(\theta)q_z(z) \]
\[ = \text{Dir}(\theta | \gamma) \sim f(\alpha, (z)) \times \]
\[ \text{Mult}(z | \phi = f(\beta, \ln \theta)) \]

- Data set:
  - 15,000 documents
  - 90,000 terms
  - 2.1 million words

- Model:
  - 100 factors
  - 9 million parameters

- MCMC could be totally infeasible for this problem
Example 2: Sigmoid belief network

Example 3: Factorial HMM
Example 4: GMF approximation to LOGOS

- Approximate $p(X, S, \theta | y)$ with a tractable distribution $q(X, S, \theta)$
- Variable partition:
  \[
  \{X, S, \theta\} = \{X\} + \{S, \theta\}
  \]
- Let
  \[
  q(X, S, \theta) = q_1(X)q_2(\theta, S)
  \]

GMF for DNA motif prediction

- GMF approximations:
  \[
  q(X, S, \theta) \propto q_1(X)q_2(\theta, S)
  \]
  \[
  q^*_1(X) = p(X | y, \tilde{\theta})
  \]
  \[
  q^*_2(\theta, S) = p(S)p(\theta | S, \tilde{A}_{y|x})
  \]
- GMF algorithm
  - Compute expectation of $\tilde{A}$ w.r.t. local posterior to $HMM$ submodel assuming $\theta$ is given
  - Compute Bayesian est. of $\tilde{\theta}$ w.r.t. local posterior to $HMM$ submodel assuming $\tilde{A}$ is given
Traces of GMF iterations

A single round of FP-iteration

Sequentialized multiple random restarts

GMF vs. Gibbs sampler on motif detection
GMF vs. Gibbs sampler on motif detection

Performance:

Sampling time of Gibbs = 10× the time for GMF

Open Problem

- Idea:
  - $A(\theta)$ is convex
  - Epigraph of $A(\theta)$ can be represented as a pointwise supremum of all affine functions that are global under-estimators of $A(\theta)$
  - Variationally, compute $A(\theta)$ using the following convex optimization:

$$A(\theta) = \sup_{\mu \in \mathbb{R}^d} \{ \langle \theta, \mu \rangle - A^*(\mu) \}$$

- Important consequence
  - Solution also yields the marginal probabilities!

Martin Wainwright and Michael Jordan
IEEE Transactions on Signal Processing, 2006