Lecture 13 - scribbles

Tuesday, October 16, 2018 14:20

Today: finish UGM
- Inference

General themes in this class:

A) representation \(\leftrightarrow\) DGM \(\leftrightarrow\) UGM

- parametrization \(\rightarrow\) exponential family

B) Inference computing \(f(x_0 | x_E)\) today: elimination algo.
- "query" "evidence"

C) statistical estimation \(\rightarrow\) MLE
  - maximum entropy
  - method of moments

Back to UGM

Properties:

- closure with respect to marginalization

Let \(V' = V \setminus \varepsilon n^2\) \(E' =\) edges in \(G \setminus \varepsilon n^2\)

+ connect all neighbors of \(n\) in \(G\) together (new clique)

\(\{\text{marginal on } x_1:n-1 \text{ for } p \in \mathcal{S}(G)\} = \mathcal{S}(G')\)
**DGM vs. UGM**

**def:** Markov blanket for i is the smallest set of nodes M s.t. $X_i \perp \gamma X / M \text{ "rest"}$

- for UGM: $M = \{j : \exists i, j \in E \}$ set of neighbors of i
- for DGM: $M = \pi \cup \text{ children}(i) \cup \bigcup_{j \in \text{ children}(i)} \pi_j$

**Recap:**

<table>
<thead>
<tr>
<th></th>
<th>DGM</th>
<th>UGM</th>
</tr>
</thead>
<tbody>
<tr>
<td>factorization</td>
<td>$p(z) = \prod_{i} p(x_i</td>
<td>\pi_i)$</td>
</tr>
<tr>
<td>cond. indep.</td>
<td>$\pi$-separation</td>
<td>separation</td>
</tr>
<tr>
<td>marginalization</td>
<td>not closed in general</td>
<td>closed</td>
</tr>
<tr>
<td></td>
<td>e.g. $x_2 \perp x_1</td>
<td>x_3$</td>
</tr>
<tr>
<td></td>
<td>but is fine for leaves</td>
<td></td>
</tr>
<tr>
<td>cannot exactly</td>
<td><img src="example.png" alt="Diagram" /></td>
<td><img src="example.png" alt="Diagram" /></td>
</tr>
<tr>
<td>capture some families</td>
<td><img src="example.png" alt="Diagram" /></td>
<td><img src="example.png" alt="Diagram" /></td>
</tr>
</tbody>
</table>

**v-structure**
Moralization

Let $G$ be a DAG; when can we transform to equivalent UGM?

**Def:** for $G$ a DAG, we call $\overline{G}$ the moralized graph of $G$

where $\overline{G}$ is an undirected graph with some $V$

and $\overline{E} = \{ \{i,j\} \mid (i,j) \in E \}$ \text{ undirected version of $E$}

$U \exists \exists k \in \overline{E}$ if $k \in \pi_i \pi_j$ for some $i,j$ \( \exists \) "moralization"

"merging the parents" (\[\overline{G}\] )

\[\begin{array}{c}
e.g., \quad G \\
\quad \longrightarrow \quad \overline{G} \\
\quad \longrightarrow \\
\end{array}\]

\[\begin{array}{c}
\text{I connect all the parents of $i$ with $i$ in } \overline{G}. \quad \text{\"moralizing the parents\")}
\end{array}\]

\[\begin{array}{c}
\text{I only need to add edges when } |\pi_i| > 1 \\
\text{\ie \text{V-structure}}
\end{array}\]

**Prop:** for a DAG $G$ with no V-structure \([\text{forest}]\)

then $\mathcal{M}(G) = \mathcal{M}(\overline{G})$

\[\begin{array}{c}
\text{DAG} \quad \text{DGM} \\
\text{GGM}
\end{array}\]

but in general, can only say that $\mathcal{M}(G) \leq \mathcal{M}(\overline{G})$

\[\begin{array}{c}
\text{[ note that } \overline{G} \text{ is the minimal undirected graph s.t. } \mathcal{M}(\overline{G}) \leq \mathcal{M}(G) \text{ ]}
\end{array}\]
Inference:

Present algo. for UGM

Make DGM \nRightarrow UGM via moralization

i.e. DGM: \[ p(x) = \sum_i p(x_i | \pi_i) \]

Moralize: \[ C = \mathcal{X} \bowtie \mathcal{U} \cup \mathcal{P} \]

\[ \psi_C(x_C) \equiv p(x_i | \pi_i) \]

Inference: want to compute:

a) Marginals: \[ p(x_F) \] for some F \subset V

b) Conditional: \[ p(x_Q | x_F) \]

"Query" "Evidence"

c) For UGM: partition function \[ Z = \sum_{X} \psi_C(x_C) \] \subset V \subset C \subset C

why?

- Missing data
- Prediction (e.g. \( F \{ Q \in \mathcal{Q} \rightarrow \text{diseases} \) )
- $p(x \text{ under } | \text{obs.})$
- prediction
  $p(x \text{ future } | \text{past})$
- "Patient Cause"
  $\propto p(x \text{ cause } | \text{obs.})$

$\text{E} \propto E \text{symptoms}$

$\text{F} \{ \text{diseases} \}
\text{image}$

"infilling task"

* also related principle:

$$\text{argmax }_{x \in \mathcal{F}} \; p(y \mid x_E)$$

$E$ could be huge
(e.g., speech recognition)

* inference is also needed during estimation (parameter fitting MLE)
  (say during E-step $p(z \mid x)$)

* graph elimination algo. (for inference)

  - consider $p \in \mathcal{G}$
  $$p(x) = \frac{1}{Z} \prod_{C \in \mathcal{G}} \mathcal{N}(x_c)$$

  say want to compute $p(y \mid x)$ for $F \subseteq V$ "query nodes"

  **Main trick:** use distributivity of $\oplus$ over $\otimes$ → $\mathbb{C}(a \oplus b) = \mathbb{C}a + \mathbb{C}b$
\[
\prod_{x_1} g(x_2) = \left( \prod_{x_1} f_1(x_1) \right) \left( \prod_{x_2} g(x_2) \right) \quad \text{[convince yourself]}
\]

More generally:
\[
\prod_{x_1} f_i(x_1) = \prod_{x_i} \left( \prod_{x_j} f_i(x_i) \right)
\]

\[
P(x_4) = \frac{1}{Z} \prod_{x_1 \neq x_4} \psi(x_1, x_2) \psi(x_1, x_3) \psi(x_2, x_4) \psi(x_3, x_4)
\]

\[
= \frac{1}{Z} \prod_{x_2 \neq x_4} \psi(x_1, x_3) \psi(x_2, x_4) \psi(x_3, x_4)
\]

\[
= \frac{1}{Z} m_1(x_2, x_3) m_2(x_3, x_4) m_2(x_2, x_4)
\]

\[
= \frac{1}{Z} m_2(x_3, x_4)
\]

last message is proportional to marginal \( p(x_4) \)
\[
\prod_{x_4} m_2(x_4) = Z
\]

**general alg. : graphEliminate**

1. **init.**
   a) choose an elimination ordering such F are the last nodes
   b) put all \( \Psi_i(x_i) \) on "active list"

2. **update**
   c) repeat in order of variables to eliminate
   (say \( x_i \) is variable to eliminate)
"update" c) repeat in order of variables to eliminate
(say x_i is variable to eliminate)

1) remove all factors from active list with x_i in xi \setminus xi
   take product
   i.e. \prod_{\alpha \in \setminus x_i} \phi(x_\alpha)

2) sum x_i to get a new factor M_i(x_S i)
   i.e. S_i all variables in these factors except i
   get \ M_i(x_S i) = \sum_{x_i} \prod_{\alpha \in \setminus x_i} \phi(x_\alpha)
   \[ S_i = \{ U, x_i \} \setminus \{ x_i \} \]
   new clique to
   sum over \[ S_i \cup U \notin \]
   3) put back M_i(x_S i) in active list

"normalise" d) last factor left has only x_i \Rightarrow proportional

memory needed \approx 2^{\max |S_i|}
(\# factors)

computational cost \approx 2^{\max |S_i| + 1} \cdot n

graph of cliques...

cliques formed during graph eliminate

5 4 2

\( \Theta \) running graphEum + keep track of all edges added
Running graphElimiEq and keep track of all edges added yields a **triangulated graph**

**Def.** A graph with no cycle of size 4 or more that cannot be broken by a "chord".

- not triangulated
- triangulated

**Treewidth of a graph**

\[
\text{treewidth of a graph} \leq \min_{\text{all elimination orderings}} \left\{ \text{size of largest clique} \right\} - 1 \]

\[
\leq \text{convention treewidth (tree)} = 1
\]

Both memory and running time of graphElimiEq is dominated by \(2^{\text{treewidth} + 2}\) best ordering gives \(2^{\text{treewidth}}\).

Not all orderings are good.

**Bad news:**

a) NP hard to compute treewidth (or find best ordering)

b) NP hard to do (exact) inference in general OGM
Example: treewidth of a grid 

\[
\text{treewidth} = \sqrt{\text{side of grid}} 
\]

\(\Rightarrow\) need approx. methods

Good news:

\(\Rightarrow\ |V| + |E|\)

- Inference in linear time for trees (treewidth = 1)
  (HMM, Markov chain)
- Efficient for "small treewidth graph"
  ("sum-product algo")

Use junction tree algo

14135