CPSC 540: Machine Learning

Undirected Graphical Models

Mark Schmidt

University of British Columbia

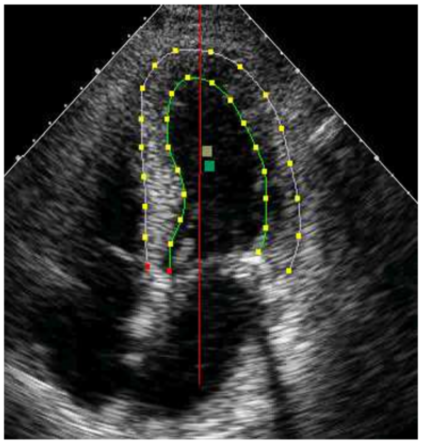
Winter 2020

Last Time: Learning and Inference in DAGs

- Learning in DAG models:
    - Given a graph structure, parameter estimation is modeling $p(x_j \mid x_{\text{pa}(j)})$.
        - We can use counting, or any method for supervised learning.
    - If we don't have the graph structure, common to use greedy "search and score".
- Inference in DAG models:
    - Inference tasks (decoding/marginalization/conditioning) are easy in trees.
        - Where we have at most one parent.
    - In non-trees, dynamic programming can be much more expensive.
        - We'll discuss approximations soon.
- We motivated looking at undirected graphical models (UGMs):
    - Can make more sense if the variables don't have a natural "ordering".

Multi-Label Classification

- Consider automated heart wall abnormality detection:



- Want to model if any of 16 areas of the heart are not moving properly.
  - Can potentially improve predictions by modeling correlations.

Ising Models from Statistical Physics

- The Ising model for binary x_i is defined by

$$p(x_1, x_2, \ldots, x_d) \propto \exp\left(\sum_{i=1}^{d} x_i w_i + \sum_{(i,j)\in E} x_i x_j w_{ij}\right),$$

  where E is the set of edges in an undirected graph.
  - Called a log-linear model, because $\log p(x)$ is linear plus a constant.

- Consider using $x_i \in \{-1, 1\}$:
  - If $w_i > 0$ it encourages $x_i = 1$.
  - If $w_{ij} > 0$ it encourages neighbours i and j to have the same value.
    - E.g., neighbouring pixels in the image receive the same label ("attractive" model)

- We're modeling dependencies, but haven't assumed an "ordering".
  - We often learn the w_i and w_{ij} from data.
  - Later, we'll see how these could be output by a neural network.

Undirected Graphical Models

- Pairwise undirected graphical models (UGMs) assume $p(x)$ has the form

$$p(x) \propto \left(\prod_{j=1}^{d} \phi_j(x_j)\right)\left(\prod_{(i,j)\in E} \phi_{ij}(x_i, x_j)\right).$$

- The ϕ_j and ϕ_{ij} functions are called potential functions:
  - They can be any non-negative function.
  - Ordering doesn't matter: more natural for things like pixels of an image.

- Ising model is a special case where

$$\phi_i(x_i) = \exp(x_i w_i), \quad \phi_{ij}(x_i, x_j) = \exp(x_i x_j w_{ij}).$$

- Bonus slides generalize Ising to non-binary case.

Gaussians as Undirected Graphical Models

- Multivariate Gaussian can be written as

$$p(x) \propto \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right) \propto \exp\left(-\frac{1}{2}x^T\Sigma^{-1}x + x^T\underbrace{\Sigma^{-1}\mu}_{v}\right),$$

and writing it in summation notation we can see that it's a pairwise UGM:

$$p(x) \propto \exp(\left(-\frac{1}{2}\sum_{i=1}^{d}\sum_{j=1}^{d}x_ix_j(\Sigma^{-1})_{ij} + \sum_{i=1}^{d}x_iv_i\right)$$

$$= \left(\prod_{i=1}^{d}\prod_{j=1}^{d}\underbrace{\exp\left(-\frac{1}{2}x_ix_j(\Sigma^{-1})_{ij}\right)}_{\phi_{ij}(x_i,x_j)}\right)\left(\prod_{i=1}^{d}\underbrace{\exp\left(x_iv_i\right)}_{\phi_i(x_i)}\right)$$

- Above we include all edges. You can "remove" edges by setting $(\Sigma^{-1})_{ij} = 0$.
- "Gaussian graphical model" (GGM) or "Gaussian Markov random field" (GMRF).

Label Propagation as a UGM

- Consider modeling the probability of a vector of labels $\bar{y} \in \mathbb{R}^t$ using

$$p(\bar{y}^1, \bar{y}^2, \dots, \bar{y}^t) \propto \exp\left(-\sum_{i=1}^{n}\sum_{j=1}^{t} w_{ij}(y^i - \bar{y}^i)^2 - \frac{1}{2}\sum_{i=1}^{t}\sum_{j=1}^{t} \bar{w}_{ij}(\bar{y}^i - \bar{y}^j)^2\right).$$

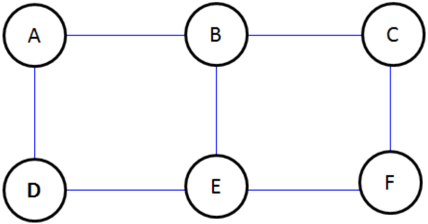
- Decoding in this model is the label propagation problem.

- This is a pairwise UGM:

$$\phi_j(\bar{y}^j) = \exp\left(-\sum_{i=1}^{n} w_{ij}(y^i - \bar{y}^j)^2\right), \quad \phi_{ij}(\bar{y}^i, \bar{y}^j) = \exp\left(-\frac{1}{2}\bar{w}_{ij}(\bar{y}^i - \bar{y}^j)^2\right).$$

Conditional Independence in Undirected Graphical Models

- It's easy to check conditional independence in UGMs:
  - $A \perp B \mid C$ if C blocks all paths from any A to any B.
- Example:



  - $A \not\perp C$.
  - $A \not\perp C \mid B$.
  - $A \perp C \mid B, E$.
  - $A, B \not\perp F \mid C$
  - $A, B \perp F \mid C, E$.

Independence in Gaussians

- Independence in multivariate Gaussian:
  - In Gaussians, marginal independence is determined by covariance:

$$x_i \perp x_j \Leftrightarrow \Sigma_{ij} = 0,$$

    (we previously saw diagonal Σ means all x_i independent).

- Gaussian conditional independence is determined by precision matrix sparsity.
  - Diagonal Θ gives disconnected graph: all variables are independent.
  - Full Θ gives fully-connected graph: there are no independences.

- Gaussians are pairwise UGMs with $\phi_{ij}(x_i, x_j) = \exp\left(-\frac{1}{2}x_i x_j \Theta_{ij}\right)$,
  - Where Θ_{ij} is element (i, j) of Σ^{-1}.

- If $\Theta_{ij} \neq 0$ we have an edge in the UGM (direct dependency between x_i and x_j).
  - Related to partial correlation which us $-\Theta_{ij}/\sqrt{\Theta_{ii}\Theta_{jj}}$.
    - The "correlation after controlling for other variables".

Independence in GGMs

- Consider a Gaussian with the following covariance matrix:

$$\Sigma = \begin{bmatrix} 0.0494 & -0.0444 & -0.0312 & 0.0034 & -0.0010 \\ -0.0444 & 0.1083 & 0.0761 & -0.0083 & 0.0025 \\ -0.0312 & 0.0761 & 0.1872 & -0.0204 & 0.0062 \\ 0.0034 & -0.0083 & -0.0204 & 0.0528 & -0.0159 \\ -0.0010 & 0.0025 & 0.0062 & -0.0159 & 0.2636 \end{bmatrix}$$

- $\Sigma_{ij} \neq 0$ so all variables are dependent: $x_1 \not\perp x_2$, $x_1 \not\perp x_5$, and so on.
  - This would show up in graph: you would be able to reach any x_i from any x_j.
- The inverse is given by a tri-diagonal matrix:

$$\Sigma^{-1} = \begin{bmatrix} 32.0897 & 13.1740 & 0 & 0 & 0 \\ 13.1740 & 18.3444 & -5.2602 & 0 & 0 \\ 0 & -5.2602 & 7.7173 & 2.1597 & 0 \\ 0 & 0 & 2.1597 & 20.1232 & 1.1670 \\ 0 & 0 & 0 & 1.1670 & 3.8644 \end{bmatrix}$$

- So conditional independence is described by a Markov chain:

$$p(x_1 \mid x_2, x_3, x_4, x_5) = p(x_1 \mid x_2).$$

Graphical Lasso

- Conditional independence in Gaussians is described by sparsity in $\Theta = \Sigma^{-1}$.
  - Setting a Θ_{ij} to 0 removes an edge from the graph.

- Recall fitting multivariate Gaussian with L1-regularization,

$$\underset{\Theta \succ 0}{\text{argmin}}\ \text{Tr}(S\Theta) - \log|\Theta| + \lambda\|\Theta\|_1,$$

which is called the graphical Lasso because it encourages a sparse graph.

- Graphical Lasso is a convex approach to structure learning for GGMs.
  - Examples: https://normaldeviate.wordpress.com/2012/09/17/high-dimensional-undirected-graphical-models.

Higher-Order Undirected Graphical Models

- In UGMs, we can also define potentials on higher-order interactions.
  - A three-variable generalization of Ising potentials is:

$$\phi_{ijk}(x_i, x_j, x_k) = w_{ijk} x_i x_j x_k.$$

    - If $w_{ijk} > 0$ and $x_j \in \{0, 1\}$, encourages you to set all three to 1.
    - If $w_{ijk} > 0$ and $x_j \in \{-1, 1\}$, encourages odd number of positives.
- In the general case, a UGM just assumes $p(x)$ factorizes over subsets c,

$$p(x_1, x_2, \ldots, x_d) \propto \prod_{c \in \mathcal{C}} \phi_c(x_c),$$

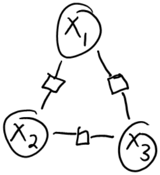
  from among a collection of subsets of $\mathcal{C}$.

- In this case, graph has edge (i, j) if i and j are together in at least one c.
  - Conditional independences are still given by graph separation.

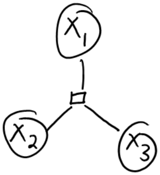
Factor Graphs

- Factor graphs are a way to visualize UGMs that distinguishes different orders.
  - Use circles for variables, squares to represent dependencies.

- Factor graph of $p(x_1, x_2, x_3) \propto \phi_{12}(x_1, x_2)\phi_{13}(x_1, x_3)\phi_{23}(x_2, x_3)$:



- Factor graph of $p(x_1, x_2, x_3) \propto \phi_{123}(x_1, x_2, x_3)$:



Outline

Tractability of UGMs

- Without using $\propto$, a UGM probability would be

$$p(x) = \frac{1}{Z}\prod_{c \in \mathcal{C}} \phi_c(x_c),$$

where Z is the constant that makes the probabilites sum up to 1.

$$Z = \sum_{x_1}\sum_{x_2}\cdots\sum_{x_d}\prod_{c \in \mathcal{C}} \phi_c(x_c) \quad \text{or} \quad Z = \int_{x_1}\int_{x_2}\cdots\int_{x_d}\prod_{c \in \mathcal{C}} \phi_c(x_c) dx_d dx_{d-1} \dots dx_1.$$

- Whether you can compute Z depends on the choice of the ϕ_c:
  - Gaussian case: $O(d^3)$ in general, but $O(d)$ for forests (no loops).
  - Continuous non-Gaussian: usually requires numerical integration.
  - Discrete case: #P-hard in general, but $O(dk^2)$ for forests (no loops).

Discrete DAGs vs. Discrete UGMs

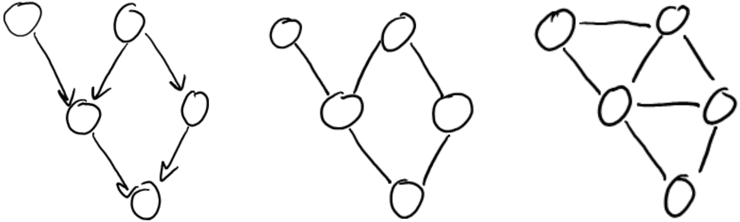
- Common inference tasks in graphical models:
  1. Compute $p(x)$ for an assignment to the variables x.
  2. Generate a sample x from the distribution.
  3. Compute univariate marginals $p(x_j)$.
  4. Compute decoding $\text{argmax}_x\, p(x)$.
  5. Compute univariate conditional $p(x_j \mid x_{j'})$.
- With discrete x_i, all of the above are easy in tree-structured graphs.
  - For DAGs, a tree-structured graph has at most one parent.
  - For UGMs, a tree-structured graph has no cycles.
- With discrete x_i, the above may be harder for general graphs:
  - In DAGs the first two are easy, the others are NP-hard.
  - In UGMs all of these are NP-hard.

Moralization: Converting DAGs to UGMs

- To address the NP-hard problems, DAGs and UGMs use same techniques.
- We'll focus on UGMs, but we can convert DAGs to UGMs:

$$p(x_1, x_2, \dots, x_d) = \prod_{j=1}^{d} p(x_j | x_{\text{pa}(j)}) = \prod_{j=1}^{d} \underbrace{\phi_j(x_j, x_{\text{pa}(j)})}_{=p(x_j \mid x_{\text{pa}(j)})},$$

  which is a UGM with $Z = 1$.
- Graphically: we drop directions and "marry" parents (moralization).

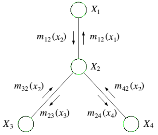


- May lose some conditional independences, but doesn't change computational cost.

Easy Cases: Chains, Trees and Forests

- The forward-backward algorithm still works for chain-structured UGMs:
  - We compute the forward messages M and the backwards messages V.
  - With both M and V we can [conditionally] decode/marginalize/sample.

- Belief propagation generalizes this to trees:
  - Pick an arbitrary node as the "root", and order the nodes going away from the root.
    - Pass messages starting from the "leaves" going towards the root.
  - "Root" is like the last node in a Markov chain.
    - Backtrack from root to leaves to do decoding/sampling.
    - Send messages from the root going to the leaves to compute all marginals.



https://www.quora.com/
Probabilistic-graphical-models-what-are-the-relationships-between-sum-product-algorithm-belief-propagation-and-junction-tree-

Easy Cases: Chains, Trees and Forests

- Recall the CK equations in Markov chains:

$$M_c(x_c) = \sum_{x_p} p(x_c \mid x_p) M_p(x_p).$$

- For chain-structure UGMs we would have:

$$M_c(x_c) \propto \sum_{x_p} \phi(x_p)\phi(x_p, x_c) M_p(x_p).$$

- In tree-structured UGMs, parent p in the ordering may have multiple parents.
- Message coming from "parent" p that has parents j and k would be

$$M_{pc}(x_c) \propto \sum_{x_p} \phi_i(x_p)\phi_{pc}(x_p, x_c) M_{jp}(x_p) M_{kp}(x_p),$$

- Univariate marginals are proportional to $\phi_i(x_i)$ times all "incoming" messages.
  - The "forward" and "backward" Markov chain messages are a special case.
  - Replace $\sum_{x_i}$ with $\max_{x_i}$ for decoding.
    - "Sum-product" and "max-product" algorithms.

Exact Inference in UGMs

- Message passing is also efficient in some non-tree graphs.

- For example, computing Z in a simple 4-node cycle could be done using:

$$
\begin{aligned}
Z &= \sum_{x_4}\sum_{x_3}\sum_{x_2}\sum_{x_1} \phi_{12}(x_1,x_2)\phi_{23}(x_2,x_3)\phi_{34}(x_3,x_4)\phi_{14}(x_1,x_4) \\
&= \sum_{x_4}\sum_{x_3} \phi_{34}(x_3,x_4)\sum_{x_2}\phi_{23}(x_2,x_3)\sum_{x_1}\phi_{12}(x_1,x_2)\phi_{14}(x_1,x_4) \\
&= \sum_{x_4}\sum_{x_3} \phi_{34}(x_3,x_4)\sum_{x_2}\phi_{23}(x_2,x_3)M_{24}(x_2,x_4) \\
&= \sum_{x_4}\sum_{x_3} \phi_{34}(x_3,x_4)M_{34}(x_3,x_4) = \sum_{x_4} M_4(x_4).
\end{aligned}
$$

- Message-passing cost depends on graph structure and the order of the sums.

Exact Inference in UGMs

- To see the effect of the order, consider Markov chain inference with bad ordering:

$$p(x_5) = \sum_{x_5}\sum_{x_4}\sum_{x_3}\sum_{x_2}\sum_{x_1} p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_2)p(x_4 \mid x_3)p(x_5 \mid x_4)$$

$$= \sum_{x_5}\sum_{x_1}\sum_{x_4}\sum_{x_3}\sum_{x_2} p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_2)p(x_4 \mid x_3)p(x_5 \mid x_4)$$

$$= \sum_{x_5}\sum_{x_1} p(x_1) \sum_{x_3}\sum_{x_4} p(x_4 \mid x_3)p(x_5 \mid x_4) \underbrace{\sum_{x_2} p(x_2 \mid x_1)p(x_3 \mid x_2)}_{M_{13}(x_1,x_3)}$$

- So even though we have a chain, we have an M with k^2 values instead of k.
  - Inference can be exponentially more expensive with the wrong ordering.

Variable Order and Treewidth

- So cost of message passing depends on
  1. Graph structure.
  2. Variable order.

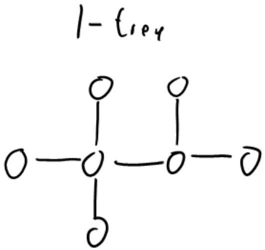
- Cost of message passing is given by $O(dk^{\omega+1})$.
  - Here, ω is the size of the largest message.
  - For trees, $\omega = 1$ so we get our usual cost of $O(dk^2)$.

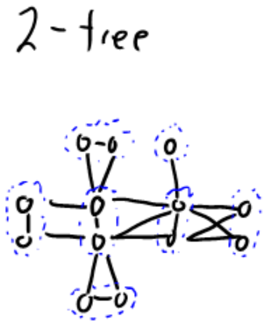
- The minimum value of ω across orderings for a given graph is called treewidth.
  - In terms of graph: "minimum size of largest clique, minus 1, over all triangulations".
    - Also called "graph dimension" or "ω-tree".

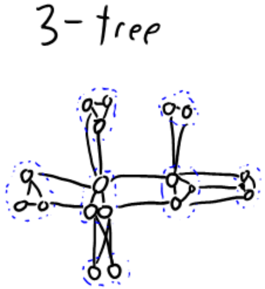
  - Intuitively, you can think of low treewidth as being "close to a tree".

Treewidth Examples

- Examples of k-trees:



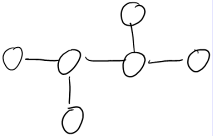




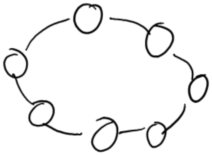
- 2-tree and 3-tree are trees if you use dotted circles to group nodes.

Treewidth Examples

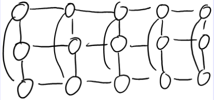
- Trees have $\omega = 1$, so with the right order inference costs $O(dk^2)$.



- A big loop has $\omega = 2$, so cost with the right ordering is $O(dk^3)$.



- The below grid-like structure has $\omega = 3$, so cost is $O(dk^4)$.



Variable Order and Treewidth

- Junction trees generalize belief propagation to general graphs (requires ordering).
- Computing ω and the optimal ordering is NP-hard.
  - But various heuristic ordering methods exist.

- An m_1 by m_2 lattice has $\omega = \min\{m_1, m_2\}$.
  - So you can do exact inference on "wide chains" with Junction tree.
  - But for 28 by 28 MNIST digits it would cost $O(784 \cdot 2^{29})$.
- Some links if you want to read about treewidth:
  - https://www.win.tue.nl/~nikhil/courses/2015/2W008/treewidth-erickson.pdf
  - https://math.mit.edu/~apost/courses/18.204-2016/18.204_Gerrod_Voigt_final_paper.pdf

- For some graphs $\omega = (d-1)$ so there is no gain over brute-force enumeration.
  - Many graphs have high treewidth so we need approximate inference.

Summary

- Undirected graphical models factorize probability into non-negative potentials.
  - Gaussians are a special case.
  - Log-linear models (like Ising) are a common choice.
  - Simple conditional independence properties.
- Moralization of DAGs to do decoding/inference/sampling as a UGM.
- Message passing can be used for inference in UGMs.
  - Belief propagation for trees.
  - Cost might be exponential for unfavourable graphs/ordering.
- Next time: our first visit to the wild world of approximate inference.

General Pairwise UGM

- For general discrete x_i a generalization of Ising models is

$$p(x_1, x_2, \ldots, x_d) = \frac{1}{Z} \exp\left(\sum_{i=1}^{d} w_{i,x_i} + \sum_{(i,j)\in E} w_{i,j,x_i,x_j}\right),$$

  which can represent any "positive" pairwise UGM (meaning $p(x) > 0$ for all x).

- Interpretation of weights for this UGM:
  - If $w_{i,1} > w_{i,2}$ then we prefer $x_i = 1$ to $x_i = 2$.
  - If $w_{i,j,1,1} > w_{i,j,2,2}$ then we prefer $(x_i = 1, x_j = 1)$ to $(x_i = 2, x_j = 2)$.

- As before, we can use parameter tieing:
  - We could use the same w_{i,x_i} for all positions i.
  - Ising model corresponds to a particular parameter tieing of the w_{i,j,x_i,x_j}.

Decomposable Graphical Models

- Probabilities whose conditional independences that can be represented as DAGs *and* UGMs are called decomposable.
    - Includes chains, trees, and fully-connected graphs.

- These models allow some efficient operations in UGMs by writing them as DAGs:
    - Computing $p(x)$.
    - Ancestral sampling.
    - Fitting parameters independently.

Other Graphical Models

- Factor graphs: we use a square between variables that appear in same factor.
  - Can distinguish between a 3-way factor and 3 pairwise factors.
- Chain-graphs: DAGs where each block can be a UGM.
- Ancestral-graph:
  - Generalization of DAGs that is closed under conditioning.
- Structural equation models (SEMs): generalization of DAGs that allows cycles.