CPSC 440: Advanced Machine Learning

More DAGs 3

Mark Schmidt

University of British Columbia

Winter 2021

Last Time: Learning in DAG Models

- Learning in DAGs involves fitting each $p(x_j \mid x_{\text{pa}(j)})$:
  1. Set $\bar{y}^i = x_j^i$ and $\bar{x}^i = x_{\text{pa}(j)}^i$.
  2. Solve a supervised learning problem using $\{\bar{X}, \bar{y}\}$.
     - Gives you a model of $p(x_j \mid x_{\text{pa}(j)})$.
- Combine the d regression/classification models as the density estimator.
  - We've turned unsupervised learning into supervised learning.

- We can use our usual tricks:
  - Linear models, non-linear bases, regularization, kernel trick, random forests, etc.
  - With least squares for continuos x_j it's called a Gaussian belief network.
  - With logistic regression for binary x_j it's called a sigmoid belief networks.
  - Don't need Markov assumptions to tractably fit these models.

DAGs: Big Picture

- Setting the parameters of a DAG model:
  - Get the graph from an expert, or learn the graph (later).
  - Given the conditional probabilities from an expert, or learn them from data.
    - Counting if you use general discrete distribution for conditionals.
    - Supervised learning for conditions.
    - Combine either of the above with EM if you have hidden/missing values.

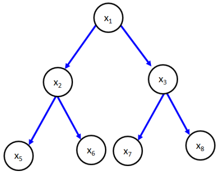
- Inference in DAG models:
  - Can use Monte Carlo approximations with ancestral sampling:
    - Sample x_1 from $p(x_1)$, x_2 from $p(x_2 \mid x_{\text{pa}(2)})$, x_3 from $p(x_3 \mid x_{\text{pa}(3)})$,...
  - Can use dynamic programming for exact inference with discrete x_j.
    - Also works if all $p(x_j \mid x_{\text{pa}(j)})$ are Gaussian.
    - But dynamic programming may be too expensive (today).

Inference in Forest DAGs ("Belief Propagation")

- If we try to generalize the CK equations to DAGs we obtain

$$p(x_j = s) = \sum_{x_{\text{pa}(j)}} p(x_j = s, x_{\text{pa}(j)}) = \sum_{x_{\text{pa}(j)}} \underbrace{p(x_j = s \mid x_{\text{pa}(j)})}_{\text{given}} p(x_{\text{pa}(j)}).$$

  which works if each node has at most one parent.
  - Such graphs are called trees (connected), or forests (disconnected).
    - Also called "singly-connected".



  - Forests allow efficient dynamic programming methods as in Markov chains.
    - In particular, decoding and univariate marginals/conditionals in $O(dk^2)$.
    - Forward-backward applied to tree-structured graphs is called belief propagation.

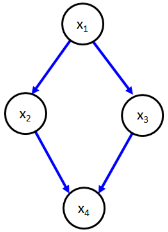
Inference in General DAGs

- If we try to generalize the CK equations to DAGs we obtain

$$p(x_j = s) = \sum_{x_{\text{pa}(j)}} p(x_j = s, x_{\text{pa}(j)}) = \sum_{x_{\text{pa}(j)}} \underbrace{p(x_j = s \mid x_{\text{pa}(j)})}_{\text{given}} p(x_{\text{pa}(j)}).$$

- What goes wrong if nodes have multiple parents?
  - The expression $p(x_{\text{pa}(j)})$ is a joint distribution depending on multiple variables.

- Consider the non-tree graph:



Inference in General DAGs

- We can compute $p(x_4)$ in this non-tree using:

$$\begin{aligned} p(x_4) &= \sum_{x_3}\sum_{x_2}\sum_{x_1} p(x_1, x_2, x_3, x_4) \\ &= \sum_{x_3}\sum_{x_2}\sum_{x_1} p(x_4 \mid x_2, x_3)p(x_3 \mid x_1)p(x_2 \mid x_1)p(x_1) \\ &= \sum_{x_3}\sum_{x_2} p(x_4 \mid x_2, x_3) \underbrace{\sum_{x_1} p(x_3 \mid x_1)p(x_2 \mid x_1)p(x_1)}_{M_{23}(x_2,x_3)} \end{aligned}$$

- Dependencies between $\{x_1, x_2, x_3\}$ mean our message depends on two variables.

$$\begin{aligned} p(x_4) &= \sum_{x_3}\sum_{x_2} p(x_4 \mid x_2, x_3) M_{23}(x_2, x_3) \\ &= \sum_{x_3} M_{34}(x_3, x_4), \end{aligned}$$

Inference in General DAGs

- With 2-variable messages, our cost increases to $O(dk^3)$.

- If we add the edge $x_1 -> x_4$, then the cost is $O(dk^4)$.

  (the same cost as enumerating all possible assignments)

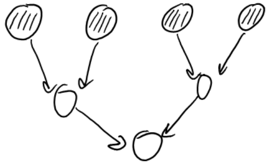
- Unfortunately, cost is not as simple as counting number of parents.
  - Even if each node has 2 parents, we may need huge messages.
  - Decoding is NP-hard and computing marginals is $\#$P-hard in general.

  - We'll see later that maximum message size is "treewidth" of a particular graph.

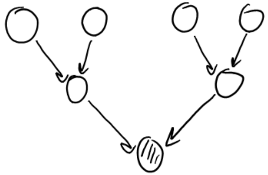
- On the other hand, ancestral sampling is easy:
  - We can obtain Monte Carlo estimates of solutions to these NP-hard problems.

Conditional Sampling in DAGs

- What about conditional sampling in DAGs?
  - Could be easy or hard depending on what we condition on.
- For example, easy if we condition on the first variables in the order:
  - Just fix these and run ancestral sampling.



- Hard to condition on the last variables in the order:
  - Conditioning on descendent makes ancestors dependent.

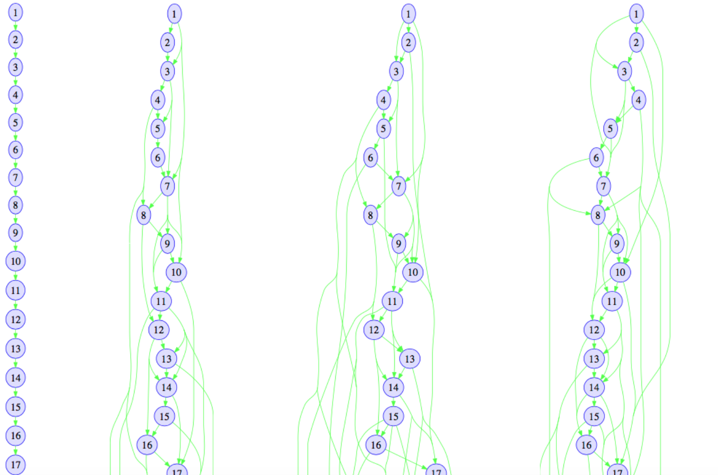


DAG Structure Learning

- Structure learning is the problem of choosing the graph.
  - Input is data X.
  - Output is a graph G.
- The "easy" case is when we're given the ordering of the variables.
  - So the parents of j must be chosen from $\{1, 2, \dots, j-1\}$.
- Given the ordering, structure learning reduces to feature selection:
  - Select features $\{x_1, x_2, \dots, x_{j-1}\}$ that best predict "label" x_j.
  - We can use any feature selection method to solve these d problems.

Example: Structure Learning in Rain Data Given Ordering

- Structure learning in rain data using L1-regularized logistic regression.
  - For different λ values, assuming chronological ordering.

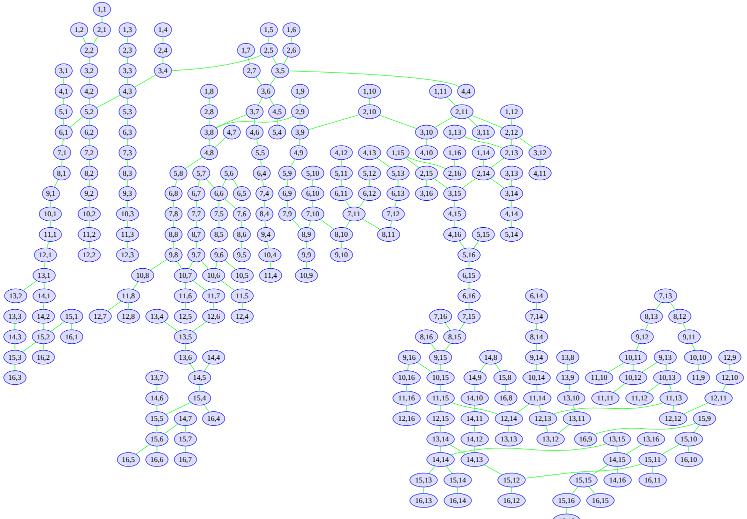


DAG Structure Learning without an Ordering

- Without an ordering, a common approach is "search and score"
  - Define a score for a particular graph structure (like BIC or other L0-regularizers).
  - Search through the space of possible DAGs.
    - "DAG-Search": at each step greedily add, remove, or reverse an edge.
- May have equivalent graphs with the same score (don't trust edge direction).
  - Do not interpret causally a graph learned from data.
- Structure learning is NP-hard in general, but finding the optimal tree is poly-time:
  - For symmetric scores, can be found by minimum spanning tree ("Chow-Liu").
    - Score is symmetric if score($x_j \rightarrow x_{j'}$) is the same as score($x_{j'} \rightarrow x_j$).
  - For asymetric scores, can be found by minimum spanning arborescence.

Structure Learning on USPS Digits

An optimal tree on USPS digits (16 by 16 images of digits).



20 Newsgroups Data

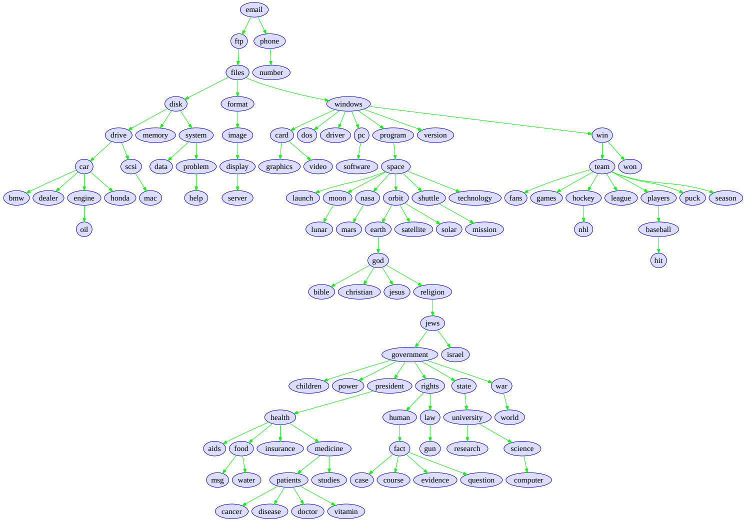
- Data containing presence of 100 words from newsgroups posts:

car	drive	files	hockey	mac	league	pc	win
0	0	1	0	1	0	1	0
0	0	0	1	0	1	0	1
1	1	0	0	0	0	0	0
0	1	1	0	1	0	0	0
0	0	1	0	0	0	1	1

- Structure learning should give some relationship between word occurrences.

Structure Learning on News Words

Optimal tree on newsgroups data:



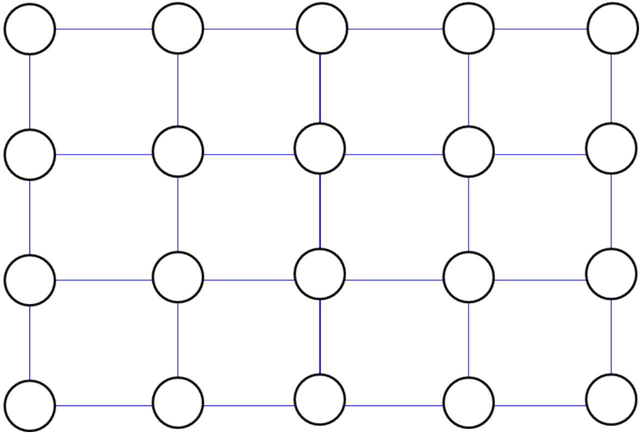
Outline

Directed vs. Undirected Models

- In some applications we have a natural ordering of the x_j.
  - In the "rain" data, the past affects the future.
- In some applications we don't have a natural order.
  - E.g., pixels in an image.
- In these settings we often use undirected graphical models (UGMs).
  - Also known as Markov random fields (MRFs) and originally from statistical physics.
    - Another name is "Markov networks".

Directed vs. Undirected Models

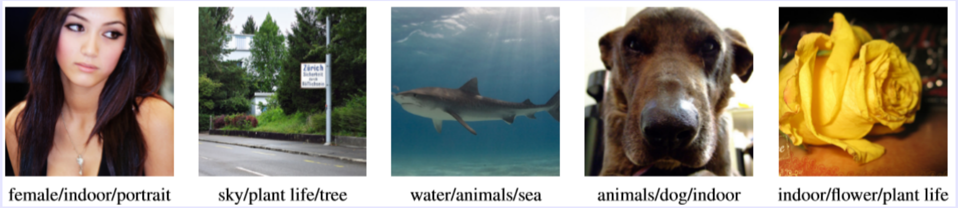
- Undirected graphical models are based on undirected graphs:



- They are a classic way to model dependencies in images:
    - Can capture dependencies between neighbours without imposing an ordering.

Multi-Label Classification

- Consider multi-label classification:

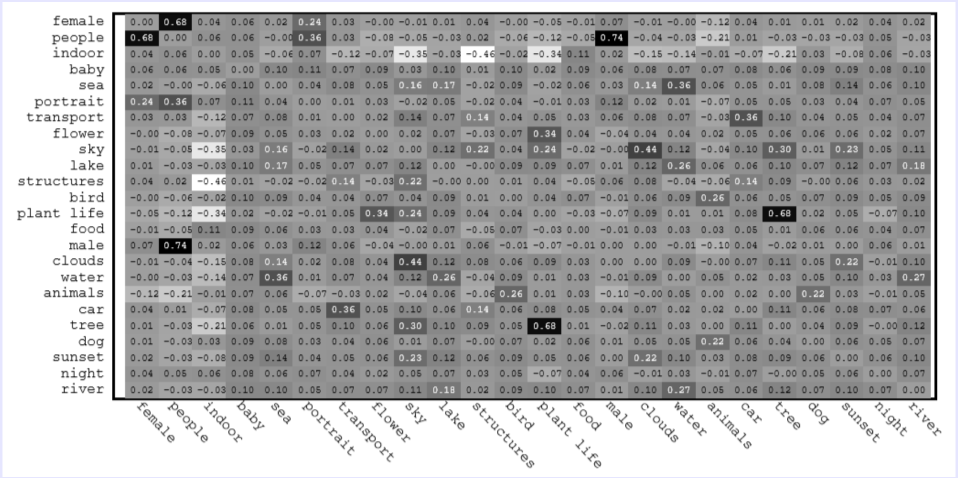


`http://proceedings.mlr.press/v37/chenb15.pdf`

- Flickr dataset: each image can have multiple labels (out of 38 possibilities).
- Use neural networks to generate “factors” in an undirected model.
  - Decoding undirected model makes predictions accounting for label correlations.
  - We’ll discuss how neural networks and density models fit together later.

Multi-Label Classification

- Learned correlation matrix:



	female	people	indoor	baby	sea	portrait	transport	flower	sky	lake	structures	bird	plant life	food	male	clouds	water	animals	car	tree	dog	sunset	night	river
female	0.00	0.68	0.04	0.06	0.02	0.24	0.03	-0.00	-0.01	0.01	0.04	-0.00	-0.05	-0.01	0.07	-0.01	-0.00	-0.12	0.04	0.01	0.01	0.02	0.04	0.02
people	0.68	0.00	0.06	0.06	-0.00	0.36	0.03	-0.08	-0.05	-0.03	0.02	-0.06	-0.12	-0.05	0.74	-0.04	-0.03	-0.21	0.01	-0.03	-0.03	-0.03	0.05	-0.03
indoor	0.04	0.06	0.00	0.05	-0.06	0.07	-0.12	-0.07	-0.35	-0.03	-0.46	-0.02	-0.34	0.11	0.02	-0.15	-0.14	-0.01	-0.07	-0.21	0.03	-0.08	0.06	-0.03
baby	0.06	0.06	0.05	0.00	0.10	0.11	0.07	0.09	0.03	0.10	0.01	0.10	0.02	0.09	0.06	0.08	0.07	0.07	0.08	0.06	0.09	0.09	0.08	0.10
sea	0.02	-0.00	-0.06	0.10	0.00	0.04	0.08	0.05	0.16	0.17	-0.02	0.09	-0.02	0.06	0.03	0.14	0.36	0.06	0.05	0.01	0.08	0.14	0.06	0.10
portrait	0.24	0.36	0.07	0.11	0.04	0.00	0.01	0.03	-0.02	0.05	-0.02	0.04	-0.01	0.03	0.12	0.02	0.01	-0.07	0.05	0.05	0.03	0.04	0.07	0.05
transport	0.03	0.03	-0.12	0.07	0.08	0.01	0.00	0.02	0.14	0.07	0.14	0.04	0.05	0.03	0.06	0.08	0.07	-0.03	0.36	0.10	0.04	0.05	0.04	0.07
flower	-0.00	-0.08	-0.07	0.09	0.05	0.03	0.02	0.00	0.02	0.07	-0.03	0.07	0.34	0.04	-0.04	0.04	0.04	0.02	0.05	0.06	0.06	0.06	0.02	0.07
sky	-0.01	-0.05	-0.35	0.03	0.16	-0.02	0.14	0.02	0.00	0.12	0.22	0.04	0.24	-0.02	-0.00	0.44	0.12	-0.04	0.10	0.30	0.01	0.23	0.05	0.11
lake	0.01	-0.03	-0.03	0.10	0.17	0.05	0.07	0.07	0.12	0.00	-0.00	0.09	0.09	0.07	0.01	0.12	0.26	0.06	0.06	0.10	0.07	0.12	0.07	0.18
structures	0.04	0.02	-0.46	0.01	-0.02	-0.02	0.14	-0.03	0.22	-0.00	0.00	0.01	0.04	-0.05	0.06	0.08	-0.04	-0.06	0.14	0.09	-0.00	0.06	0.03	0.02
bird	-0.00	-0.06	-0.02	0.10	0.09	0.04	0.04	0.07	0.04	0.09	0.01	0.00	0.04	0.07	-0.01	0.06	0.09	0.26	0.06	0.05	0.07	0.09	0.05	0.09
plant life	-0.05	-0.12	-0.34	0.02	-0.02	-0.01	0.05	0.34	0.24	0.09	0.04	0.04	0.00	-0.03	-0.07	0.09	0.01	0.01	0.08	0.68	0.02	0.05	-0.07	0.10
food	-0.01	-0.05	0.11	0.09	0.06	0.03	0.03	0.04	-0.02	0.07	-0.05	0.07	-0.03	0.00	-0.01	0.03	0.03	0.03	0.05	0.01	0.06	0.06	0.04	0.07
male	0.07	0.74	0.02	0.06	0.03	0.12	0.06	-0.04	-0.00	0.01	0.06	-0.01	-0.07	-0.01	0.00	0.00	-0.01	-0.10	0.04	-0.02	0.01	0.00	0.06	0.01
clouds	-0.01	-0.04	-0.15	0.08	0.14	0.02	0.08	0.04	0.44	0.12	0.08	0.06	0.09	0.03	0.00	0.00	0.09	-0.00	0.07	0.11	0.05	0.22	-0.01	0.10
water	-0.00	-0.03	-0.14	0.07	0.36	0.01	0.07	0.04	0.12	0.26	-0.04	0.09	0.01	0.03	-0.01	0.09	0.00	0.05	0.02	0.03	0.05	0.10	0.03	0.27
animals	-0.12	-0.21	-0.01	0.07	0.06	-0.07	-0.03	0.02	-0.04	0.06	-0.06	0.26	0.01	0.03	-0.10	-0.00	0.05	0.00	0.02	0.00	0.22	0.03	-0.01	0.05
car	0.04	0.01	-0.07	0.08	0.05	0.05	0.36	0.05	0.10	0.06	0.14	0.06	0.08	0.05	0.04	0.07	0.02	0.02	0.00	0.11	0.06	0.08	0.07	0.06
tree	0.01	-0.03	-0.21	0.06	0.01	0.05	0.10	0.06	0.30	0.10	0.09	0.05	0.68	0.01	-0.02	0.11	0.03	0.00	0.11	0.00	0.04	0.09	-0.00	0.12
dog	0.01	-0.03	0.03	0.09	0.08	0.03	0.04	0.06	0.01	0.07	-0.00	0.07	0.02	0.06	0.01	0.05	0.05	0.22	0.06	0.04	0.00	0.06	0.05	0.07
sunset	0.02	-0.03	-0.08	0.09	0.14	0.04	0.05	0.06	0.23	0.12	0.06	0.09	0.05	0.06	0.00	0.22	0.10	0.03	0.08	0.09	0.06	0.00	0.06	0.10
night	0.04	0.05	0.06	0.08	0.06	0.07	0.04	0.02	0.05	0.07	0.03	0.05	-0.07	0.04	0.06	-0.01	0.03	-0.01	0.07	-0.00	0.05	0.06	0.00	0.07
river	0.02	-0.03	-0.03	0.10	0.10	0.05	0.07	0.07	0.11	0.18	0.02	0.09	0.10	0.07	0.01	0.10	0.27	0.05	0.06	0.12	0.07	0.10	0.07	0.00

Summary

- Inference in DAGs:
  - Ancestral sampling and Monte Carlo methods work as before.
  - Message-passing message sizes depend on graph structure.
- Structure learning is the problem of learning the graph structure.
  - Hard in general, but easy for trees and L1-regularization gives fast heuristic.
- Undirected graphical models do not require an ordering of the variables.
- Next time: easy conditional dependence and hard "everything else" in UGMs.

“Constraint-Based” DAG Structure Learning

- Another common structure learning approach is “constraint-based”:
  - Based on performing a sequence of conditional independence tests.
  - Prune edge between x_i and x_j if you find variables S making them independent,

$$x_i \perp x_j \mid x_S.$$

  - Challenge is considering exponential number of sets x_S (heuristic: “PC algorithm”).
  - Assumes “faithfulness” (all independences are reflected in graph).
    - Otherwise it’s weird (a duplicated feature would be disconnected from everything.)