

Learning Graphical Models

Aayush Bajaj

August 31, 2026

everything so far assumed the network was handed to us. in practice an expert sketches the graph at best, and the numbers — sometimes the graph too — must come from data. learning splits along two axes: *parameters vs structure*, and *complete vs incomplete data*. the four quadrants ascend steeply in difficulty: counting, then optimisation, then search over graphs, then all three at once.

this is the pipeline of the comp9418 assignment: learn the outcome spaces and cpts from an icu dataset, then classify with complete and with missing evidence

Contents

1	parameter learning from complete data	1
1.1	the zero problem and additive smoothing	2
1.2	worked microcosm: a text classifier	2
2	missing data	2
2.1	classification with missing evidence	2
2.2	learning with missing data: em	2
3	structure learning	3
3.1	scoring: likelihood needs a leash	3
3.2	the tree case is solved: chow–liu	3
3.3	structure is not causality	3
4	results	4
5	see also	4

1 parameter learning from complete data

fix the dag; estimate the cpts from a dataset of complete instantiations. the log-likelihood of the data decomposes, by the chain rule of bayesian networks, into one independent term per cpt row — so the global maximum-likelihood estimate is just local **counting**:

$$\hat{\theta}_{x|\mathbf{u}} = \frac{c(x, \mathbf{u})}{c(\mathbf{u})}, \quad (1)$$

the fraction of the $\mathbf{u}$ -rows of the data in which $X = x$. one pass over the data fills every table. (a small practicality that bites in real datasets: even the *outcome space* of each variable may need learning — take the set of values observed in the data, and be aware a rare value may be missing from it entirely.)

1.1 the zero problem and additive smoothing

counting has a sharp edge: an event unseen in the data gets probability exactly zero, and a single zero factor annihilates the whole chain-rule product — one novel word and a naive bayes spam filter scores the email impossible under *both* classes, dividing by zero at normalisation. **additive (laplace) smoothing** pads every count by a pseudo-count α :

$$\hat{P}_L(x) = \frac{c(x) + \alpha}{N + \alpha |X|}, \quad \hat{\theta}_{x|\mathbf{u}} = \frac{c(x, \mathbf{u}) + \alpha}{c(\mathbf{u}) + \alpha |X|}, \quad (2)$$

shrinking the estimate towards uniform; $\alpha = 1$ is laplace's rule of succession, and the bayesian reading is exact — smoothing is the posterior mean under a dirichlet prior with α pseudo-observations per outcome. the deeper the table (more parents), the thinner each $c(\mathbf{u})$ slice of data, and the more the smoothing matters — parameter estimation is the hidden tax on structure complexity.

overfitting in a bayesian network is precisely this: tables faithful to the sample and brittle about the world. smoothing is the l2 of cpts

1.2 worked microcosm: a text classifier

a naive bayes ham/spam filter is parameter learning end to end: outcome space = vocabulary; $\hat{P}(C)$ = class frequencies; $\hat{P}(w | C)$ = smoothed word counts per class; classification = a map query computed in log-space because a thousand word-probabilities multiplied together underflow long before they inform.

2 missing data

real records have holes — sensors fail, patients skip tests. two distinct problems hide here.

2.1 classification with missing evidence

if the *model* is known, missing attributes at prediction time are not a problem but an inference call: marginalise them out. for naive bayes this is deliciously simple — an unobserved attribute's factor sums to one and drops out, so you classify on the attributes you have. this graceful degradation is a genuine advantage of generative classifiers over discriminative ones, which need imputation or retraining.

2.2 learning with missing data: em

if the *training data* is incomplete, the likelihood no longer decomposes and counting is undefined. **expectation-maximisation** restores the counting picture by iterating:

- **e-step** — with current parameters $\theta^{(k)}$, run inference on each incomplete record to compute *expected counts*: each record contributes fractional counts $P(x, \mathbf{u} | \text{observed part}; \theta^{(k)})$ to every completion consistent with it;
- **m-step** — re-estimate cpts from the expected counts exactly as if they were data (smoothing applies unchanged).

each iteration provably does not decrease the likelihood; convergence is to a local maximum, so initialisation matters and restarts are standard. latent-variable models — clustering as a bayesian network with an unobserved class node, hmm training (baum-welch) — are em with a variable that is missing in *every* record.

3 structure learning

now the graph itself is unknown. the space of dags is super-exponential, so everything hinges on how the problem is scored and constrained.

3.1 scoring: likelihood needs a leash

maximum likelihood alone always prefers denser graphs — an extra edge can only fit better, terminating in the useless complete dag. usable scores charge for capacity:

$$\text{score}(G) = \log \hat{L}(G) - \frac{\log N}{2} \dim(G), \quad (3)$$

the bic/mdl form ($\dim(G)$ = number of free cpt parameters); fully bayesian scores (bde) integrate the parameters out. all decompose per-family, so local search — add, delete, reverse an edge; greedy hill-climbing with tabu lists and restarts — evaluates candidates cheaply. the alternative school is *constraint-based*: run conditional-independence tests, reconstruct whatever graph is consistent with them (the pc algorithm), recovering structure only up to markov equivalence.

3.2 the tree case is solved: chow–liu

restrict to trees (each node at most one parent) and the optimal structure is exact and cheap: the maximum-likelihood tree is the **maximum spanning tree** of the complete graph weighted by pairwise mutual information

$$I(X; Y) = \sum_{x,y} \hat{P}(x, y) \log \frac{\hat{P}(x, y)}{\hat{P}(x) \hat{P}(y)}, \quad (4)$$

because a tree’s log-likelihood is a constant plus the sum of edge mutual informations. **tan** — tree-augmented naive bayes — is chow–liu run with conditional mutual information $I(A_i; A_j \mid C)$ on top of the class arrows:

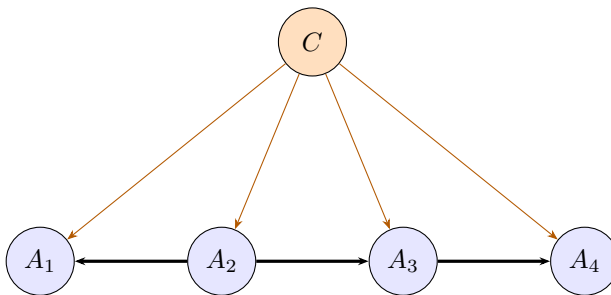


Figure 1: tree-augmented naive bayes: the class points at every attribute (orange arrows) and the attributes carry a chow–liu tree among themselves (dark arrows) — each attribute gets at most one attribute parent.

3.3 structure is not causality

a learned dag is an i-map of the data’s independencies, nothing more: within a markov-equivalence class, edge directions are unidentifiable from observational data alone (all three of $X \rightarrow W \rightarrow Y$, $X \leftarrow W \leftarrow Y$, $X \leftarrow W \rightarrow Y$ encode the same single independence; only the collider $X \rightarrow W \leftarrow Y$ stands out). learning *causal* direction needs interventional data, temporal ordering, or assumptions well beyond likelihood. resist reading arrows in a learned network as mechanisms.

4 results

- **mle for cpts** — with complete data the likelihood decomposes; $\hat{\theta}_{x|\mathbf{u}} = c(x, \mathbf{u})/c(\mathbf{u})$, computable in one pass.
- **additive smoothing** — $(c + \alpha)/(N + \alpha|X|)$; equals the posterior mean under a symmetric dirichlet prior; mandatory wherever a zero count would meet the chain rule.
- **missing evidence at test time** — marginalise, don't impute: inference already defines the answer.
- **em** — expected counts (inference) alternating with counting (mle); monotone in likelihood; local maxima; baum-welch and model-based clustering as instances.
- **penalised scores** — bic/mdl subtract $\frac{\log N}{2} \dim(G)$; decomposability enables local search over edge edits.
- **chow-liu** — the optimal tree structure is the mst under pairwise mutual information; $O(n^2)$ statistics + mst; tan is its class-conditional variant.
- **markov equivalence** — observational data identifies structure only up to equivalence class; colliders are the identifiable feature.

5 see also

- bayesian networks — the representation being learned, and naive bayes/tan as classifiers
- exact inference — the subroutine inside em and inside every classification query
- approximate inference — when the e-step itself needs approximating
- markov chains and hidden markov models — baum-welch is em on the hmm