Bayesuvius,
a small visual dictionary of Bayesian Networks

Robert R. Tucci
www.ar-tiste.xyz

August 7, 2020

Figure 1: View of Mount Vesuvius from Pompeii

Figure 2: Mount Vesuvius and Bay of Naples

Contents

0.2 Notational Conventions

bnet=B net=Bayesian Network

Define $\mathbb{Z}, \mathbb{R}, \mathbb{C}$ to be the integers, real numbers and complex numbers, respectively.

For $a < b$, define $\mathbb{Z}_I$ to be the integers in the interval I, where $I = [a,b], [a,b), (a,b], (a,b)$ (i.e, I can be closed or open on either side).

$A_{>0} = \{k \in A : k > 0\}$ for $A = \mathbb{Z}, \mathbb{R}$.

Random Variables will be indicated by underlined letters and their values by non-underlined letters. Each node of a bnet will be labelled by a random variable. Thus, $\underline{x} = x$ means that node $\underline{x}$ is in state x.

$P_{\underline{x}}(x) = P(\underline{x} = x) = P(x)$ is the probability that random variable $\underline{x}$ equals $x \in S_{\underline{x}}$. $S_{\underline{x}}$ is the set of states (i.e., values) that $\underline{x}$ can assume and $n_{\underline{x}} = |S_{\underline{x}}|$ is the size (aka cardinality) of that set. Hence,

$$\sum_{x \in S_{\underline{x}}} P_{\underline{x}}(x) = 1 \tag{1}$$

$$P_{\underline{x},\underline{y}}(x,y) = P(\underline{x} = x, \underline{y} = y) = P(x,y) \tag{2}$$

$$P_{\underline{x}|\underline{y}}(x|y) = P(\underline{x} = x|\underline{y} = y) = P(x|y) = \frac{P(x,y)}{P(y)} \tag{3}$$

Kronecker delta function: For x, y in discrete set S,

$$\delta(x,y) = \begin{cases} 1 \text{ if } x = y \\ 0 \text{ if } x \neq y \end{cases} \tag{4}$$

Dirac delta function: For $x, y \in \mathbb{R}$,

$$\int_{-\infty}^{+\infty} dx\ \delta(x-y) f(x) = f(y) \tag{5}$$

Transition probability matrix of a node of a bnet can be either a discrete or a continuous probability distribution. To go from continuous to discrete, one replaces integrals over states of node by sums over new states, and Dirac delta functions by Kronecker delta functions. More precisely, consider a function $f : S \to \mathbb{R}$. Let $S_{\underline{x}} \subset S$ and $S \to S_{\underline{x}}$ upon discretization (binning). Then

$$\int_S dx\ P_{\underline{x}}(x) f(x) \to \frac{1}{n_{\underline{x}}} \sum_{x \in S_{\underline{x}}} f(x)\ . \tag{6}$$

Both sides of last equation are 1 when $f(x) = 1$. Furthermore, if $y \in S_{\underline{x}}$, then

$$\int_S dx\ \delta(x-y) f(x) = f(y) \to \sum_{x \in S_{\underline{x}}} \delta(x,y) f(x) = f(y)\ . \tag{7}$$

Indicator function (aka Truth function):

$$\mathbb{1}(\mathcal{S}) = \begin{cases} 1 \text{ if } \mathcal{S} \text{ is true} \\ 0 \text{ if } \mathcal{S} \text{ is false} \end{cases} \tag{8}$$

For example, $\delta(x,y) = \mathbb{1}(x=y)$.

$$\vec{x} = (x[0], x[1], x[2] \ldots, x[nsam(\vec{x})-1]) = x[:] \tag{9}$$

$nsam(\vec{x})$ is the number of samples of $\vec{x}$. $\underline{x}[i]$ are i.d.d. (independent identically distributed) samples with

$$x[i] \sim P_{\underline{x}} \text{ (i.e. } P_{\underline{x[i]}} = P_{\underline{x}}) \tag{10}$$

$$P(\underline{x} = x) = \frac{1}{nsam(\vec{x})} \sum_i \mathbb{1}(x[i] = x) \tag{11}$$

If we use two sampled variables, say $\vec{x}$ and $\vec{y}$, in a given bnet, their number of samples $nsam(\vec{x})$ and $nsam(\vec{y})$ need not be equal.

$$P(\vec{x}) = \prod_i P(x[i]) \tag{12}$$

$$\sum_{\vec{x}} = \prod_i \sum_{x[i]} \tag{13}$$

$$\partial_{\vec{x}} = [\partial_{x[0]}, \partial_{x[1]}, \partial_{x[2]}, \ldots, \partial_{x[nsam(\vec{x})-1]}] \tag{14}$$

$$\begin{aligned} P(\vec{x}) &\approx [\prod_x P(x)^{P(x)}]^{nsam(\vec{x})} && (15)\\ &= e^{nsam(\vec{x}) \sum_x P(x) \ln P(x)} && (16)\\ &= e^{-nsam(\vec{x}) H(P_{\underline{x}})} && (17) \end{aligned}$$

$$f^{[1,\partial_x,\partial_y]}(x,y) = [f, \partial_x f, \partial_y f] \tag{18}$$

$$f^+ = f^{[1,\partial_x,\partial_y]} \tag{19}$$

For probabilty distributions $p(x), q(x)$ of $x \in S_{\underline{x}}$

- Entropy:

$$H(p) = -\sum_x p(x) \ln p(x) \geq 0 \tag{20}$$

- Kullback-Liebler divergence:

$$D_{KL}(p \parallel q) = \sum_x p(x) \ln \frac{p(x)}{q(x)} \geq 0 \tag{21}$$

- Cross entropy:

$$\begin{aligned} CE(p \to q) &= -\sum_x p(x) \ln q(x) && (22)\\ &= H(p) + D_{KL}(p \parallel q) && (23) \end{aligned}$$

Normal Distribution: $x, \mu, \sigma \in \mathbb{R}$, $\sigma > 0$

$$\mathcal{N}(\mu, \sigma^2)(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{24}$$

Uniform Distribution: $a < b$, $x \in [a, b]$

$$\mathcal{U}(a, b)(x) = \frac{1}{b-a} \tag{25}$$

Expected Value

Given a random variable $\underline{x}$ with states $S_{\underline{x}}$ and a function $f : S_{\underline{x}} \to \mathbb{R}$, define

$$E_{\underline{x}}[f(\underline{x})] = E_{x \sim P(x)}[f(x)] = \sum_x P(x) f(x) \tag{26}$$

Conditional Expected Value

Given a random variable $\underline{x}$ with states $S_{\underline{x}}$, a random variable $\underline{y}$ with states $S_{\underline{y}}$, and a function $f : S_{\underline{x}} \times S_{\underline{y}} \to \mathbb{R}$, define

$$E_{\underline{x}|\underline{y}}[f(\underline{x}, \underline{y})] = \sum_x P(x|\underline{y}) f(x, \underline{y}) \ , \tag{27}$$

$$E_{\underline{x}|\underline{y}=y}[f(\underline{x}, y)] = E_{\underline{x}|y}[f(\underline{x}, y)] = \sum_x P(x|y) f(x, y) \ . \tag{28}$$

Note that

$$\begin{aligned} E_{\underline{y}}[E_{\underline{x}|\underline{y}}[f(\underline{x}, \underline{y})]] &= \sum_{x,y} P(x|y) P(y) f(x, y) && (29) \\ &= \sum_{x,y} P(x, y) f(x, y) && (30) \\ &= E_{\underline{x},\underline{y}}[f(\underline{x}, \underline{y})] \ . && (31) \end{aligned}$$

Sigmoid function: For $x \in \mathbb{R}$,

$$\mathrm{sig}(x) = \frac{1}{1 + e^{-x}} \tag{32}$$

$\mathcal{N}(!a)$ will denote a normalization constant that does not depend on a. For example, $P(x) = \mathcal{N}(!x) e^{-x}$ where $\int_0^\infty dx \ P(x) = 1$.

A **one hot** vector of zeros and ones is a vector with all entries zero with the exception of a single entry which is one. A **one cold** vector has all entries equal to one with the exception of a single entry which is zero. For example, if $x^n = (x_0, x_1, \ldots, x_{n-1})$ and $x_i = \delta(i, 0)$ then x^n is one hot.

Chapter 8

Expectation Maximization

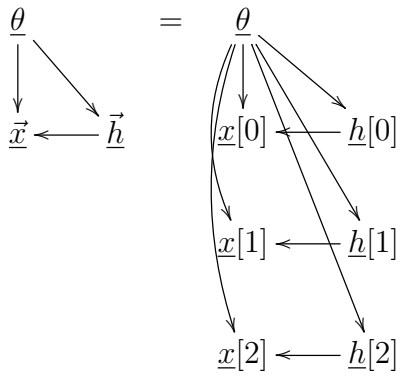

Figure 8.1: bnet for EM with $nsam = 3$.

This chapter is based on Wikipedia Ref.[3].

The bnet for Expectation Maximization (EM) is given by Fig.8.1 for $nsam = 3$. Later on in this chapter, we will give the node transition prob matrices for this bnet for the special case in which $P(x[i] \mid \theta)$ is a mixture (i.e., weighted sum) of Gaussians.

Note that if we erase the $\underline{h}[i]$ nodes from Fig.8.1, we get the bnet for naive Bayes, which is used for classification into the states of $\underline{\theta}$. However, there is one big difference. With naive Bayes, the leaf nodes have different transition prob matrices. Here, we will assume they are i.i.d. Naive Bayes is used for classification: i.e., given the states of the leaf nodes, we infer the state of the root node. EM is used for clustering; i.e., given many i.i.d. samples, we fit their distribution by a weighted sum of prob distributions, usually Gaussians.

Let

$\mathcal{L}$ =likelihood function.

$nsam$ = number of samples.

$\vec{x} = (x[0], x[1], \ldots, x[nsam-1])$ = **observed data**. $x[i] \in S_{\underline{x}}$ for all i.

$\vec{h} = (h[0], h[1], \ldots, h[nsam-1])$ = **hidden or missing data**. $h[i] \in S_{\underline{h}}$ for all i.

We assume that the samples $(x[i], h[i])$ are i.i.d. for different i at fixed θ. What this means is that there are probability distributions $P_{\underline{x}|\underline{h},\underline{\theta}}$ and $P_{\underline{h}|\underline{\theta}}$ such that

$$P(\vec{x}, \vec{h}|\theta) = \prod_i \left[P_{\underline{x}|\underline{h},\underline{\theta}}(x[i] \mid h[i], \theta) P_{\underline{h}|\underline{\theta}}(h[i] \mid \theta) \right] \ . \tag{8.1}$$

Definition of likelihood functions:

$$\underbrace{P(\vec{x}|\theta)}_{\mathcal{L}(\theta;\vec{x})} \ = \ \sum_{\vec{h}} \underbrace{P(\vec{x}, \vec{h}|\theta)}_{\mathcal{L}(\theta;\vec{x},\vec{h})} \tag{8.2}$$

θ^* = maximum likelihood estimate of θ (no prior $P(\theta)$ assumed):

$$\theta^* = \underset{\theta}{\operatorname{argmax}}\, \mathcal{L}(\theta; \vec{x}) \tag{8.3}$$

The EM algorithm:

1. **Expectation step:**
$$Q(\theta|\theta^{(t)}) = E_{\vec{h}|\vec{x},\theta^{(t)}} \ln P(\vec{x}, \vec{h}|\theta) \tag{8.4}$$

2. **Maximization step:**
$$\theta^{(t+1)} = \underset{\theta}{\operatorname{argmax}}\, Q(\theta|\theta^{(t)}) \tag{8.5}$$

Claim: $\lim_{t\to\infty} \theta^{(t)} = \theta^*$.

Motivation

$$\begin{aligned} Q(\theta|\theta) &= E_{\vec{h}|\vec{x},\theta} \ln P(\vec{x}, \vec{h}|\theta) && (8.6)\\ &= E_{\vec{h}|\vec{x},\theta}[\ln P(\vec{h}|\vec{x}, \theta) + \ln P(\vec{x}|\theta)] && (8.7)\\ &= -H[P(\underline{\vec{h}}|\vec{x}, \theta)] + \ln P(\vec{x}|\theta) && (8.8) \end{aligned}$$

$$\begin{aligned} \partial_\theta Q(\theta|\theta) &= -\sum_{\vec{h}} \partial_\theta P(\underline{\vec{h}}|\vec{x}, \theta) + \partial_\theta \ln P(\vec{x}|\theta) && (8.9)\\ &= \partial_\theta \ln P(\vec{x}|\theta) && (8.10) \end{aligned}$$

So if $\theta^{(t)} \to \theta$ and $Q(\theta|\theta)$ is max at $\theta = \theta^*$, then $\ln P(\vec{x}|\theta)$ is max at $\theta = \theta^*$ too.

For a more rigorous proof that $\lim_{t\to\infty} \theta^{(t)} = \theta^*$, see Wikipedia article Ref.[3] and references therein.

EM for Gaussian mixture

$x[i] \in \mathbb{R}^d = S_{\underline{x}}$. $S_{\underline{h}}$ discrete and not too large. $n_{\underline{h}} = |S_{\underline{h}}|$ is number of Gaussians that we are going to fit the samples with.

Let

$$\theta = [w_h, \mu_h, \Sigma_h]_{h \in S_{\underline{h}}} \ , \tag{8.11}$$

where $[w_h]_{h \in S_{\underline{h}}}$ is a probability distribution of weights, and where $\mu_h \in \mathbb{R}^d$ and $\Sigma_h \in \mathbb{R}^{d \times d}$ are the mean value vector and covariance matrix of a d-dimensional Gaussian distribution.

The transition prob matrices, printed in blue, for the nodes of Fig.8.1, for the special case of a mixture of Gaussians, are as follows:

$$P(x[i] \mid h[i] \mid \theta) = \mathcal{N}_d(x[i]; \mu_{h[i]}, \Sigma_{h[i]}) \tag{8.12}$$

$$P(h[i] \mid \theta) = w_{h[i]} \tag{8.13}$$

Note that

$$\begin{aligned} P(x[i] \mid \theta) &= \sum_h P(x[i] \mid h[i] = h, \theta) P(h[i] = h \mid \theta) && (8.14) \\ &= \sum_h w_h \mathcal{N}_d(x[i]; \mu_h, \Sigma_h) && (8.15) \end{aligned}$$

$$\begin{aligned} P(\vec{x}, \vec{h} | \theta) &= \prod_i \left[w_{h[i]} \mathcal{N}_d(x[i]; \mu_{h[i]}, \Sigma_{h[i]}) \right] && (8.16) \\ &= \prod_i \prod_h \left[w_h \mathcal{N}_d(x[i]; \mu_h, \Sigma_h) \right]^{\mathbb{1}(h = h[i])} && (8.17) \end{aligned}$$

Old Faithful: See Wikipedia Ref.[3] for an animated gif of a classic example of using EM to fit samples with a Gaussian mixture. Unfortunately, could not include it here because pdflatex does not support animated gifs. It shows samples in a 2 dimensional space (eruption time, delay time) from the Old Faithful geyser. In that example, $d = 2$ and $n_{\underline{h}} = 2$. Two clusters of points in a plane are fitted by a mixture of 2 Gaussians.

K-means clustering is often presented as the main competitor to EM for doing **clustering (non-supervised learning)**. In K-means clustering, the sample points are split into K mutually disjoint sets $S_0, S_1, \ldots, S_{K-1}$. The algorithm is easy to describe:

1. Initialize by choosing at random K data points $(\mu_k)_{k=0}^{K-1}$ called means or centroids and placing μ_k in S_k for all k.

2. **STEP 1:** For each data point, add it to the S_k whose centroid μ_k is closest to it.

3. **STEP 2:** Recalculate the centroids. Set μ_k equal to the mean value of set S_k.

4. Repeat steps 1 and 2 until the centroids stop changing by much.

Step 1 is analogous to the expectation step in EM, and Step 2 to the maximization step in EM (θ estimation versus μ_k estimation). We won't say anything further about K-means clustering because it isn't related to bnets in any way, and this is a book about bnets. For more info about K-means clustering, see Ref.[4].

Bibliography

[1] Robert R. Tucci. Bell's inequalities for Bayesian statisticians. blog post in blog Quantum Bayesian Networks, https://qbnets.wordpress.com/2008/09/19/bells-inequaties-for-bayesian-statistician/.

[2] Wikipedia. Binary decision diagram. https://en.wikipedia.org/wiki/Binary_decision_diagram.

[3] Wikipedia. Expectation maximization. https://en.wikipedia.org/wiki/Expectation%E2%80%93maximization_algorithm.

[4] Wikipedia. k-means clustering. https://en.wikipedia.org/wiki/K-means_clustering.

[5] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, David Warde-Farley Bing Xu, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. https://arxiv.org/abs/1406.2661.

[6] Wikipedia. Hidden Markov model. https://en.wikipedia.org/wiki/Hidden_Markov_model.

[7] Gregory Nuel. Tutorial on exact belief propagation in Bayesian networks: from messages to algorithms. https://arxiv.org/abs/1201.4724.

[8] Wikipedia. Kalman filter. https://en.wikipedia.org/wiki/Kalman_filter.

[9] Wikipedia. Markov blanket. https://en.wikipedia.org/wiki/Markov_blanket.

[10] Judea Pearl. *Probabilistic Inference in Intelligent Systems*. Morgan Kaufmann, 1988.

[11] Nitish Srivastava, G E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. http://www.cs.toronto.edu/~rsalakhu/papers/srivastava14a.pdf.

[12] Wikipedia. Non-negative matrix factorization. https://en.wikipedia.org/wiki/Non-negative_matrix_factorization.

[13] Andrew Ng. Lecture at deeplearning.ai on recurrent neural networks. http://www.ar-tiste.com/ng-lec-rnn.pdf.

[14] Wikipedia. Long short term memory. https://en.wikipedia.org/wiki/Long_short-term_memory.

[15] Wikipedia. Gated recurrent unit. https://en.wikipedia.org/wiki/Gated_recurrent_unit.

[16] Charles Fox, Neil Girdhar, and Kevin Gurney. A causal bayesian network view of reinforcement learning. https://www.aaai.org/Papers/FLAIRS/2008/FLAIRS08-030.pdf".

[17] Sergey Levine. Course CS 285 at UC Berkeley, Deep reinforcement learning. http://rail.eecs.berkeley.edu/deeprlcourse/.

[18] Robert R. Tucci. Simpson's paradox, the bane of clinical trials. blog post in blog Quantum Bayesian Networks https://qbnets.wordpress.com/2020/07/09/simpsons-paradox-the-bane-of-clinical-trials/.

[19] Robert J. McEliece, David J. C. MacKay, and Jung-Fu Cheng. Turbo decoding as an instance of Pearls belief propagation algorithm. http://authors.library.caltech.edu/6938/1/MCEieeejstc98.pdf.