Graphical Models

November 24, 2020

Abstract

Topics: Bayesian networks and Markov random fields; conditional independence; inference algorithms (sum-product and max-sum); hidden Markov models (HMM); conditional random fields (CRF).

Readings: Chapters 8 and 13.2 of PRML.

1 Motivations and overview

We have discussed probabilistic models in the lecture of generative models. For example, the naive Bayes classifier is a probabilistic classifier that estimates the conditional class probability by

$$\Pr(Y = y|\mathbf{x}) = \frac{\Pr(Y = y) \prod_{i=1}^{n} \Pr(X_i = x_i|Y = y)}{\Pr(\mathbf{x})} \tag{1}$$

In this model, the conditional independence between any two features X_i and X_j given the class label $Y = y$ avoids the estimation of covariance between any two features X_i and X_j and reduces the number of parameters to be estimated (the general joint distribution $\Pr(X_1, \ldots, X_n|Y)$ needs $O(K^n)$ parameters if each X_i is a discrete random variable and takes K possible values).

We wish to develop more sophisticated and general probabilistic models with hidden variables and parameters, while allowing more general conditional independence structures, and to exploit the conditional independence to enable efficient computation for the models. Graphical models is a general representation and computation tool to attain these goals. A graphical model contains nodes representing random variables and edges representing relationships between the variables. Depending on the Graphical models can be divided into two families: directed graphical models (Bayesian networks) and undirected graphical models (Markov random fields, or MRF for short). Bayesian networks are more suitable for modeling causal relationships between random variables, while MRF are more suitable for expressing soft constraints between variables.

Many probabilistic supervised learning and unsupervised learning models can be expressed as graphical models.

2 Bayesian networks

Bayesian networks are directed graphical models, with nodes being random variables and directed edges represent local conditional distributions. We don't specify the particular type and law of the distribution of the random variables, which can be discrete or continuous, following different distributions.

Let's start with a simple probability distribution: $\Pr(X, Y)$. The distribution can be factorized into $\Pr(X)\Pr(Y|X)$, which can be represented by the graphical model in Figure 1(a). The node X is called the *parent* of the node Y, which is a child of X. The joint distribution $\Pr(X, Y, Z)$ can be factorized into $\Pr(X)\Pr(Y|X)\Pr(Z|X, Y)$ and the corresponding graphical model is shown in Figure 1(b).

Note that a graphical model encodes how a joint distribution is factorized, and the structure of the graphical model depends that factorization. If $\Pr(X, Y, Z)$ is factorized into $\Pr(Z)\Pr(Y|Z)\Pr(X|Z, Y)$, the graphical model will be different.

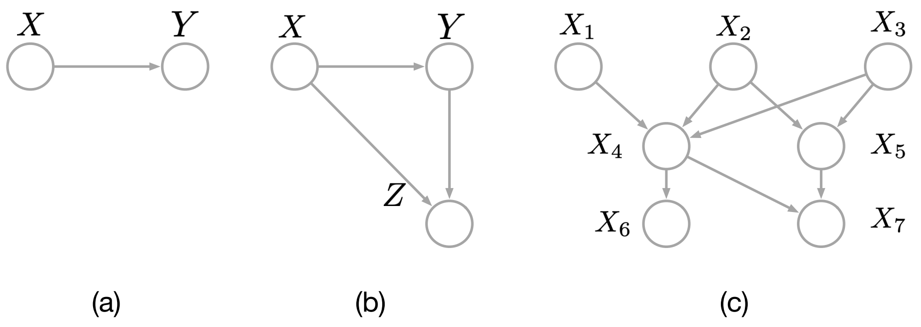


Figure 1: Example Bayesian networks.

In general, the joint distribution $\Pr(X_1, \ldots, X_n)$ for n random variables can be factorized into $\Pr(X_1) \prod_{i=2}^{n} \Pr(X_i | X_1, \ldots, X_{i-1})$ and the corresponding graphical models will have edges pointing from all nodes with lower numbers to each X_i, except the first node X_1. This graphical model is not interesting as it does not say anything about conditional independence that is more useful for computation.

Given a graphical model, a factorization of a joint distribution can be read out from the graphical model. For example, the graphical models in Figure 1(c) gives the factorization

$$\Pr(X_1, \ldots, X_7) = \Pr(X_1)\Pr(X_2)\Pr(X_3)\Pr(X_4|X_1, X_2, X_3)\Pr(X_5|X_2, X_3)\Pr(X_6|X_4)\Pr(X_7|X_4, X_5)$$

An important class of directed graphs is called the "acyclic directed graphs" (DAG), meaning that there is no loop that starts and ends in the same node by following the direction of some edges on the graph. DAG makes computation easier and the semantics of the graphical models valid.

In general, the joint distribution $\Pr(X_1, \ldots, X_n)$ for n random variables can be factorized according to a Bayesian network so that

$$\Pr(X_1, \ldots, X_n) = \prod_{i=1}^{n} \Pr(X_i | \mathrm{Pa}_i), \tag{2}$$

where $\mathrm{Pa}_i \subset \{X_1, \ldots, X_n\} \setminus \{X_i\}$ is the parents of node X_i on the DAG. It can be proved that the right hand side does sum up to 1.

Next we will see some machine learning models that can be represented as Bayesian networks, along which we introduce more notation and definition of graphical models.

Example 2.1 (**Bayesian linear regression**). In the regular linear regression, we treat the parameters $\mathbf{w} = [w_0, \ldots, w_n]$ as unknown constants to be estimated from training data $\{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{m}$, using the technique MLE (Maximum Likelihood Estimation). With a Bayesian treatment, now $\mathbf{w}$ is considered as a random vector that follows the prior distribution $\Pr(\mathbf{w}|\alpha) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \alpha I)$, where α is the precision parameter of the zero-mean Gaussian for $\mathbf{w}$. A target $y^{(i)}$ is assumed to follow the single-variate Gaussian distribution $\Pr(y^{(i)}|\mathbf{w}, \mathbf{x}^{(i)}; \sigma) = \mathcal{N}(\mathbf{w}^\top \mathbf{x}^{(i)}; \sigma^2)$ where σ^2 is the variance of the Gaussian and $\mathbf{w}^\top \mathbf{x}^{(i)}$ is the mean. We assume that the hyperparameters α and σ^2 are constants. Then the joint distribution of $y^{(1)}, \ldots, y^{(m)}, \mathbf{w}$ is

$$\Pr(y^{(1)}, \ldots, y^{(m)}, \mathbf{w}|\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}; \alpha, \sigma) = \Pr(\mathbf{w}|\alpha) \prod_{i=1}^{m} \Pr(y^{(i)}|\mathbf{w}, \mathbf{x}^{(i)}; \sigma). \tag{3}$$

You can easily read from the joint distribution that, the m targets are independent given $\mathbf{w}$, $\mathbf{x}^{(i)}$, and σ. Indeed, the graphical model in Figure 2 expresses such conditional independence.

The posterior distribution of $\mathbf{w}$ given the training set is

$$\Pr(\mathbf{w}|\{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{m}; \alpha, \sigma) \propto \Pr(\mathbf{w}|\alpha) \prod_{i=1}^{m} \Pr(y^{(i)}|\mathbf{w}, \mathbf{x}^{(i)}; \sigma), \tag{4}$$

where the $\propto$ sign means that the left hand side is proportional to the right hand side, with the difference of a scaling factor constant with respect to $\mathbf{w}$ (the inverse of $\Pr(\{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{m})$ in this example).

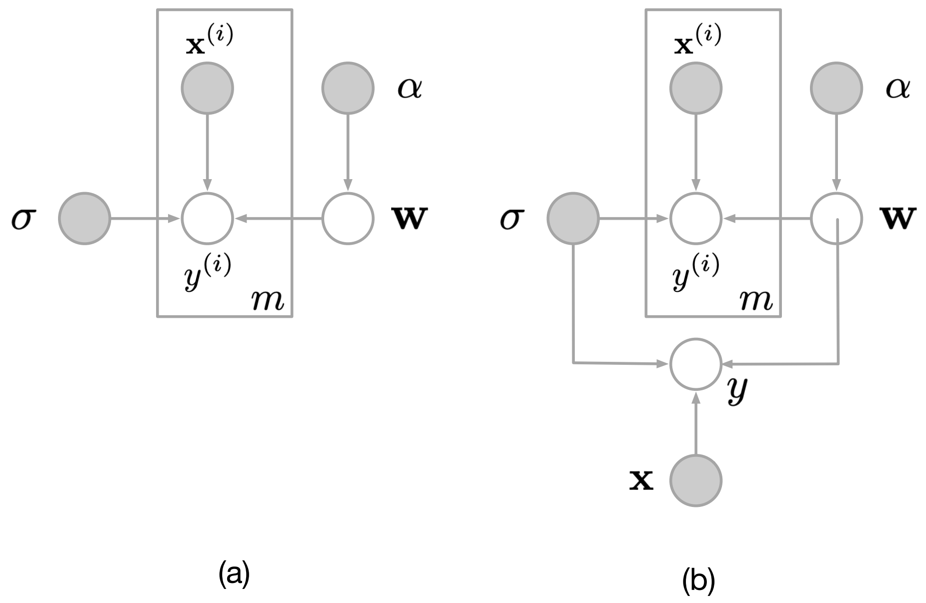


Figure 2: Bayesian linear regression. (a) A graphical model for the joint distribution of the training labels and $\mathbf{w}$. (b) A graphical model for the joint distribution of the training, a test label, and $\mathbf{w}$. Shaded nodes are observed constants and the empty ones are hidden variables.

To make prediction on a test example $\mathbf{x}$, the joint distribution of $y^{(1)}, \ldots, y^{(m)}$, y, and $\mathbf{w}$ needs to be made first

$$\Pr(y^{(1)}, \ldots, y^{(m)}, y, \mathbf{w} | \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}, \mathbf{x}; \alpha, \sigma) = \Pr(\mathbf{w}|\alpha) \prod_{i=1}^{m} \Pr(y^{(i)}|\mathbf{x}^{(i)}, \mathbf{w}; \sigma) \Pr(y|\mathbf{x}, \mathbf{w}; \sigma). \quad (5)$$

Then $\mathbf{w}$ is integrated out since it is a hidden variable

$$\Pr(y, y^{(1)}, \ldots, y^{(m)} | \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}, \mathbf{x}; \alpha, \sigma) = \int_{\mathbf{w}} \Pr(y^{(1)}, \ldots, y^{(m)}, y, \mathbf{w} | \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}, \mathbf{x}; \alpha, \sigma) d\mathbf{w} \quad (6)$$

But then $y^{(1)}, \ldots, y^{(m)}$ are observed constants, so that the predictive distribution of y given $\{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{m}$ is $\Pr(y|\{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{m}, \mathbf{x}; \alpha, \sigma)$, which can be found by normalizing

$$\Pr(y, y^{(1)}, \ldots, y^{(m)} | \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}, \mathbf{x}; \alpha, \sigma). \quad (7)$$

In this example graphical model, we observe two new symbols:

- since there are multiple targets $y^{(i)}$, $i = 1, \ldots, m$ and input feature vectors $\mathbf{x}^{(i)}$, $i = 1, \ldots, m$, for convenience, a *plate*, denoted by a box surrounding the multiple variables and constants is drawn in the graphical model. The plate is labeled by an integer, in this example m, to denote the number of the surrounded variables/constants.
- We allow one node to represent a vector or even matrix of random variables or constants. This further reduce the number of the symbols on the graphical model.
- The symbols for constants and variables are different. We let any fixed observed constant be denoted by a shaded circle, rather than a empty circle that represents a random variable/vector/matrix. This is slightly different from the notation in PRML.
- A graphical model can be seen as a module, such as the model for the joint distribution $\Pr(y^{(1)}, \ldots, y^{(m)}, \mathbf{w}|\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}; \alpha, \sigma)$ for the training data, and the one $\Pr(y|\mathbf{x}, \mathbf{w}; \sigma)$. These two modules can be integrated into a single graphical model to obtain the joint distribution of more random variables, with the shared hidden variable $\mathbf{w}$ and the hyperparameters α and σ.

Example 2.2 (Generative models). Graphical models can represent data generation mechanisms, namely, generative models. An important unsupervised learning model for text data is

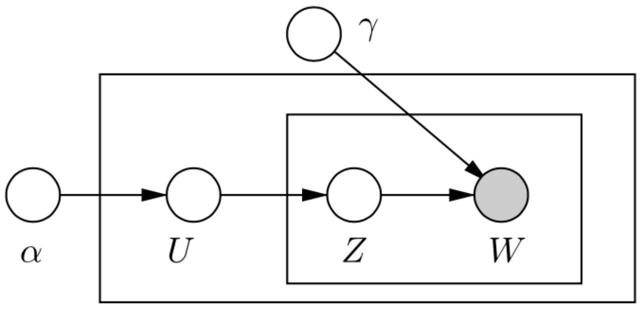


Figure 3: A graphical model for LDA. α is the hyperparameter for the Dirichlet distribution of topics U, the random variable Z represent a topic and follows a multinomial distribution over the K topics. The random variable γ is a matrix with K rows and V columns, with the k-th row representing a multinomial distribution over the V words for the k-th topic. Lastly, the integer constant $W \in \{0, \ldots, V-1\}$ represents a word observed in a document. The outer plate represents multiple documents and the inner plate represents multiple words in a document.

Latent Dirichelet Allocation (LDA). A corpora of text documents are given, with each document represented by a bag-of-words (a set of words appearing in the document). Furthermore, each document is assumed to be a mixture of K topics, and the words in a document is generated by first sampling a topic and then sample the word according to the distribution of words under that topic. For instance, a piece of news can be discussing how sport events are affected by COVID-19, then a mixture of sport and health topics will be used for this piece of news, with different words used for each of the two topics.

Without labels of the topics of the documents, LDA learns a topic distribution of each document, and a word distribution of each topic, using the graphical model shown in Figure 3. We can see that a node can represent a matrix, such as the topic-word matrix γ, and a plate can be nested inside another plate.

The distribution of U is the Dirichlet distribution of the topics

$$\Pr(U|\alpha) \propto \prod_{k=1}^{K} U_k^{\alpha_k - 1} \tag{8}$$

This distribution models the prior distribution of how likely a topic will be used for a particular document.

Then for the document, the random vector Z is associated with a word in that document and represents the topic distribution of that word.

$$\Pr(Z|U) \propto \prod_{k=1}^{K} U_k^{Z_k}. \tag{9}$$

This is a multinomial distribution, with the probability of sampling the k-th topic being U_k.

Lastly, once Z is sampled ($Z_k = 1$ and $Z_{k'} = 0$ for $k' \neq k$), the word corresponding to Z is sampled from the k-th row of γ, which is distribution over V words for the topic k.

2.1 Reducing parameters via removing edges

Graphical models can represent a wide spectrum of joint distributions of the same set of random variables $\{X_1, \ldots, X_n\}$.

When the graphical models of $\Pr(X_1, \ldots, X_n)$ has no edge, then all random variables are independent and the factorization $\Pr(X_1, \ldots, X_n) = \Pr(X_1) \ldots \Pr(X_n)$ holds. Assuming each X_k takes K discrete values, then there are $n(K-1)$ parameters need to be specified for this distribution.

If the the graphical models of $\Pr(X_1, \ldots, X_n)$ is a fully connected graph, namely, any pair of two random variables are connected by a directed edge (in just one of the two directions), then $\Pr(X_1, \ldots, X_n)$ is factorized as $\Pr(X_1) \prod_{i=2}^{n} \Pr(X_i|X_1, \ldots, X_{i-1})$. There will be $(K-1) + K(K-1) + K^2(K-1) + \cdots + K^{n-1}(K-1) = (K-1)(\sum_{i=0}^{n-1} K^i) = K^n - 1$ parameters that are required to fully specify the joint distribution.

Between these two extremes, we can have a linear chain of $\{X_1, \ldots, X_n\}$, with an edge pointing from X_{i-1} to X_i, $i = 2, \ldots, n$. Then there will be $K - 1 + (n-1)K(K-1)$ parameters to be specified. We can see that the number of parameters of a joint distribution is related to the number of edges of a graphical model for that joint distribution, because a graphical model correspond to a specific factorization of the joint distribution.

2.2 Conditional independence

A Bayesian network specifies conditional independence between any two random variables from the collection $\{X_1, \ldots, X_n\}$. There are three simple scenarios with three random variables, and we will generalize that to more random variables. The three cases are demonstrated in Figure 4

- Tail-to-tail: the graphical model in Figure 4(a) represents the factorization

$$\Pr(X, Y, Z) = \Pr(Z)\Pr(X|Z)\Pr(Y|Z). \tag{10}$$

  X and Y are not independent since, in general[1],

$$\Pr(X, Y) = \sum_z \Pr(X, Y, Z = z) \neq \Pr(X)\Pr(Y). \tag{11}$$

  However, if the value of Z is observed, X and Y are conditional independent given Z (denoted by $X \perp\!\!\!\perp Y|Z$), since

$$\Pr(X, Y|Z) = \frac{\Pr(X, Y, Z)}{\Pr(Z)} = \frac{\Pr(Z)\Pr(X|Z)\Pr(Y|Z)}{\Pr(Z)} = \Pr(X|Z)\Pr(Y|Z). \tag{12}$$

- Tail-to-head: the graphical model in Figure 4(b) represents the factorization

$$\Pr(X, Y, Z) = \Pr(X)\Pr(Z|X)\Pr(Y|Z). \tag{13}$$

  Similar to the head-to-head case, in general,

$$\Pr(X, Y) = \sum_z \Pr(X, Y, Z = z) \neq \Pr(X)\Pr(Y). \tag{14}$$

  If $Z = z$ is given, $X \perp\!\!\!\perp Y|Z$, since

$$\begin{aligned} \Pr(X, Y|Z) &= \frac{\Pr(X, Y, Z)}{\Pr(Z)} = \frac{\Pr(X)\Pr(Z|X)\Pr(Y|Z)}{\Pr(Z)} & (15) \\ &= \frac{\Pr(X, Z)\Pr(Y|Z)}{\Pr(Z)} = \Pr(X|Z)\Pr(Y|Z). & (16) \end{aligned}$$

- Head-to-head: this is very different case from the previous two. Figure 4(c) represents the factorization

$$\Pr(X, Y, Z) = \Pr(X)\Pr(Y)\Pr(Z|X, Y). \tag{17}$$

  $X \perp\!\!\!\perp Y$ since

$$\Pr(X, Y) = \sum_{Z=z} \Pr(X)\Pr(Y)\Pr(Z = z|X, Y) = \Pr(X)\Pr(Y). \tag{18}$$

  However, if $Z = z$ is observed, we have $X \not\perp\!\!\!\perp Y|Z$, as in general,

$$\Pr(X, Y|Z) = \frac{\Pr(X)\Pr(Y)\Pr(Z|X, Y)}{\Pr(Z)} \neq \Pr(X|Z)\Pr(Y|Z). \tag{19}$$

The last case can be counter-intuitive, but it is demonstrated by the real world example, shown in Figure 8.21 of PRML.

[1] There can be some specific assignments of probabilities to the random variables that make X and Y independent.

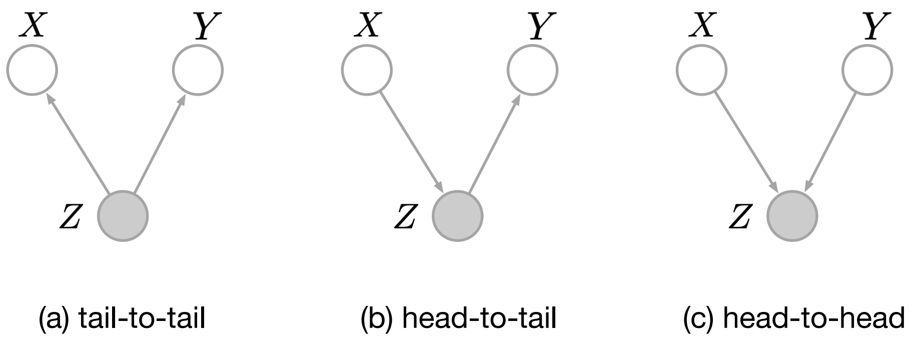


Figure 4: Three simple example graphical models demonstrating conditional independence.

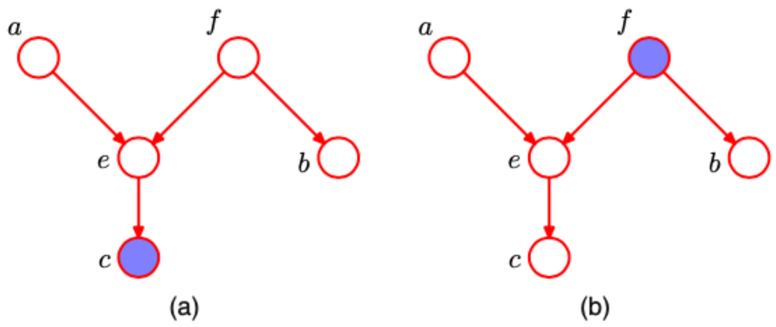


Figure 5: Examples of d-separations. Figure taken from PRML.

2.3 D-separation

The conditional independence test can be generalized to two sets of random variables A and B, conditioned on a set of random variables C. Given a graphical model containing A, B, and C (there can be other random variables), we want to check if $A \perp\!\!\!\perp B | C$. Considering any path from one random variable in A to a random variable in B, we say the path is "blocked" by C if any of the following two conditions hold:

- the path contains a node from C where the arrows on the path meet either tail-to-tail or tail-to-head; or
- the path contains a node where the arrows on the path meet head-to-head, and neither the node or any of its descendants is from C.

If all path from A to B are blocked, then $A \perp\!\!\!\perp B | C$. Figure 5 is from Figure 8.22 of PRML, showing examples of d-separation tests.

- In Figure 5(a), the path from a to b is not blocked by f since f is not observed. The path has the node e where two arrows meet head-to-head, but since the descendant of e, $c \in C$, is observed, the path is not blocked by e either. Therefore, $A \not\perp\!\!\!\perp B | C$.
- In Figure 5(b), the path from a to b is blocked by f. The path is also blocked by e or c, since two arrows on the path meet at e, and both c and e are not from the observed set $C = \{f\}$. Either one of these blockings wiill lead to the claim $A \perp\!\!\!\perp B | C$.

3 Markov Random Fields

Another family of graphical models is the undirected graphical models, or Markov random fields (MRF). Similar to Bayesian networks, MRF can represent conditional independence and factorization of a joint distribution $\Pr(X_1, \ldots, X_n)$. Nodes in an MRF represent random variables and any undirected edge represents dependencies between the two connected random variables. There are fundamental difference between Bayesian networks and MRF. There are conditional independence that can be represented by MRF but can't be represented by a Bayesian network, and vice versa (see Figure 8.35 and 8.36 of PRML).

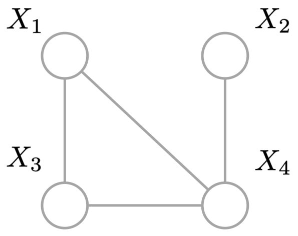


Figure 6: The path from X_1 to X_2 is blocked if and only if X_4 is observed. However, the path from X_1 to X_4 can not be blocked by any observed variables on the graph.
There are 5 cliques: $\{X_1, X_3\}$, $\{X_1, X_4\}$, $\{X_3, X_4\}$, $\{X_2, X_4\}$, and $\{X_1, X_3, X_4\}$. There are only two maximal cliques: $\{X_2, X_4\}$ and $\{X_1, X_3, X_4\}$. Note that the cliques $\{X_1, X_3\}$, $\{X_1, X_4\}$ and $\{X_3, X_4\}$ can be subsumed into the maximal clique $\{X_1, X_3, X_4\}$.

3.1 Conditional independence

MRF represents conditional independence using connectivities on the graphs, in contrast to Bayesian network that use d-separation. In particular, two sets of random variables A and B are independent conditioned on another set of random variables C, if all paths from A to B are blocked by C when the variables in C are observed. In Figure 6, we have one example of conditional independence ($X_1 \not\perp\!\!\!\perp X_2 | X_4$) and one example of non-conditional independence ($X_1 \not\perp\!\!\!\perp X_4 | (X_2, X_3)$). Since there is no direction over an edge in MRF, we can avoid the discussion of different connection patterns (e.g., head-to-head).

Then we ask the question: when are two random variables X_1 and X_2 are not blocked by *any* observed variables? The only answer is when the two random variables are directly connected on the MRF, namely, X_1 and X_2 are neighbors of each other. In other words, so long as they are not neighbors, there exists one configuration of the MRF so that some observed nodes block all paths between X_1 and X_2. By default, if there is no path between X_1 and X_2, $X_1 \perp\!\!\!\perp X_2$ under all conditions. To formalize conditional independence using a graph notion, we introduce the following definition.

Definition 3.1. Clique and maximal cliques A clique C on a graph G is a set of nodes of the graph so that any two nodes are directly connected. A maximal clique C is a clique that adding any node outside C makes the augmented set of nodes not a clique.

Example 3.1. Figure 6 shows example cliques and maximal cliques.

Any two random variables in a clique will not be conditional independent as there is no way to block the paths connecting the two random variables by conditioning on any other random variables.

Maximal cliques of an MRF identify all conditional independence of a joint distribution: variables from one maximal clique will be conditional independent of variables from another clique conditioned on all the remaining variables:

$$\Pr(X_i, X_j | X_{\setminus i,j}) = \Pr(X_i | X_{\setminus i,j}) \Pr(X_j | X_{\setminus i,j}), \tag{20}$$

where $X_{\setminus i,j}$ are the set of random variables except X_i and X_j.

3.2 Factorization using cliques

Given the set $\mathcal{C}$ of all maximal cliques on an MRF, the joint distribution $\Pr(X_1, \ldots, X_n)$ can be factorized as follows:

$$\Pr(X_1, \ldots, X_n) = \frac{1}{Z} \prod_{c \in \mathcal{C}} \psi_c(X_c), \tag{21}$$

where X_c is a subset of $\{X_1, \ldots, X_n\}$ corresponding to the nodes of the maximal clique c. Each $\psi_c(X_c) \geq 0$ is a multi-variate function mapping from some evaluation of the random variables $X_c = \{X_i : X_i \in c\}$ to a non-negative real value, and is the factor corresponding to the clique c. Sometimes such a factor is also called a *potential*. The normalization factor Z is defined as

$$Z = \sum_{x_1, \ldots, x_n} \prod_{c \in \mathcal{C}} \psi_c(X_c), \tag{22}$$

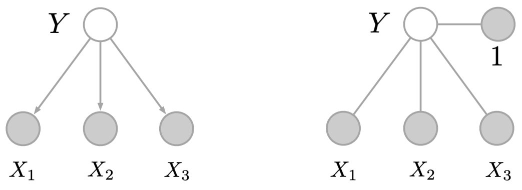


Figure 7: The Bayesian network on the left shows the graphical model for a naive Bayes classifier. The MRF on the right shows the graphical model for a logistic regression classifier.

so that $\frac{1}{Z}\prod_{c\in\mathcal{C}}\psi_c(X_c)$ is indeed a distribution.

While a subset c' of a maximal clique c is also a clique, we only consider maximal cliques, as the factor $\psi_{c'}(X_{c'})$ corresponding to c' can always be absorbed into the factor $\psi_c(X_c)$ corresponding to c:

$$\psi_{c'}(X_{c'})\psi_c(X_c)=\tilde{\psi}_c(X_c) \tag{23}$$

One can replace $\psi_{c'}(X_{c'})\psi_c(X_c)$ with $\tilde{\psi}_c(X_c)$ so that the factorization Eq. (21) involves only maximal cliques.

MRF are not locally normalized, meaning that $\psi_c(X_c)$ is not a joint distribution of the set of random variables X_c. That why the normalization factor Z in Eq. (21) needs to be calculated. Note that calculating Z in general is not computationally feasible, leading to one of the restrictions of the applications of MRFs. On the other hand, Bayesian networks are locally normalized, meaning that each random variable is associated with a conditional probability distribution $\Pr(X_i|\mathrm{Pa}_i)$, which sums to one.

If each factor $\psi_c(X_c)$ is strictly greater than zero, then $\psi_c(X_c)=\exp(-E_c(X_c))$, where $E_c(X_c)$ is the so-called energy function of the variables in c, mapping from the values of X_c to some real value (can be positive or negative). The joint distribution $\Pr(X_1,\dots,X_n)$ can then be factorized in a different manner:

$$\Pr(X_1,\dots,X_n)=\frac{1}{Z}\exp\left(-\sum_{c\in\mathcal{C}}E_c(X_c)\right). \tag{24}$$

This equation shows that MRF in fact defines an exponential family, with natural parameters being an all -1 vector of length $|\mathcal{C}|$ and the sufficient statistics being the vector consisting of elements $\{E_c(X_c):c\in\mathcal{C}\}$. Given this connection, it is not surprising that the distribution of naive Bayes and logistic regression can be represented by a MRF.

Example 3.2 (**Naive Bayes vs. logistic regression**). A graphical model can be used to represent the naive Bayes model and logistic regression models, respectively, as shown in Figure 7. The two models have the same set of random variables and connections, with the subtle difference in the directions of the edges. The naive Bayes is a generative model and the conditional probability of generating each features conditioned on the class label Y is $\Pr(X_i|Y)$ for the i-th feature, $i=1,\dots,n$. The joint probability is then factorized as

$$\Pr(X_1,\dots,X_n,Y)=\Pr(Y)\prod_{i=1}^{n}\Pr(X_i|Y)=\exp\left(-\log(1/\Pr(Y))-\sum_{i=1}^{n}\log(1/\Pr(X_i|Y))\right). \tag{25}$$

$Z=1$ since the Bayesian network is locally normalized. For logistic regression, the joint probability is

$$\Pr(Y=y|X_1,\dots,X_n)=\frac{1}{Z}\psi_y(Y=y)\prod_{i=1}^{n}\psi_i(X_i,Y=y)=\frac{1}{Z}\exp\left(\theta_y+\sum_{i=1}^{n}\theta_{i,y}X_i\right). \tag{26}$$

Note that this MRF has parametrized factors $\psi_y(Y=y)=\exp(\theta_y)$ and $\psi_i(X_i,Y=y)=\exp(\theta_{i,y}X_i)$. We allow the parameters θ to depend on y so that multi-class logistic regression can be represented by this MRF. The normalization factor $Z=\sum_y\psi_y(Y=y)\prod_{i=1}^{n}\psi_i(X_i,Y=y)$ and does not depends on a particular y.

4 Inference algorithms

With the representation of a joint distribution using a graphical model, we can efficiently calculate some quantities of interest from the joint distribution using the graph. In particular, the conditional

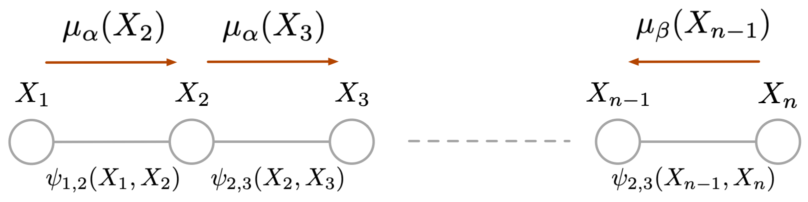


Figure 8:

independence encoded by the graph structure will allow us to find:

- Marginal distribution, such as
$$\Pr(X_i = x_i) = \sum_{\backslash X_i} \Pr(X_1, \dots, X_n). \tag{27}$$
This probability is important in learning the parameters of a parametrized graphical model, such as logistic regression.
- Maximum of a joint distribution
$$\max_{X_1,\dots,X_n} \Pr(X_1, \dots, X_n). \tag{28}$$
- Most likely prediction that maximizes the joint distribution
$$(X_1^*, \dots, X_n^*) = \arg\max_{X_1,\dots,X_n} \Pr(X_1, \dots, X_n). \tag{29}$$

We will see all these tasks when we discuss hidden Markov model and conditional random fields.

4.1 Inference on linear chain

The simplest and yet interesting graphical model[2] is the linear chain, shown in Figure 8. This chain represents the factorization

$$\Pr(X_1, \dots, X_n) = \frac{1}{Z} \prod_{i=1}^{n-1} \psi_{i,i+1}(X_i, X_{i+1}). \tag{30}$$

Note that a Bayesian network can be represented in the same manner:

$$\begin{aligned} \Pr(X_1, \dots, X_n) &= \Pr(X_1) \prod_{i=2}^{n} \Pr(X_i | X_1, \dots, X_{i-1}) & (31) \\ &= \psi_{1,2}(X_1, X_2)\psi_{2,3}(X_2, X_3) \dots \psi_{n-1,n}(X_{n-1}, X_n), & (32) \end{aligned}$$

where $\psi_{1,2}(X_1, X_2) = \Pr(X_1)\Pr(X_1|X_2)$ and $\psi_{i,i+1}(X_i, X_{i+1}) = \Pr(X_i|X_{i+1})$ for $2 \leq i \leq n-1$.

We aim to calculate the marginal distribution of X_i, defined as

$$\Pr(X_i) = \sum_{\backslash X_i} \Pr(X_1, \dots, X_n) = \frac{1}{Z} \sum_{\backslash X_i} \prod_{j=1}^{i} \psi_{j,j+1}(X_j, X_{j+1}) \prod_{j=i+1}^{n-1} \psi_{j,j+1}(X_j, X_{j+1}). \tag{33}$$

where the subscript under the summation sign $\backslash X_i$ means that the summation is taken over all possible evaluations of the random variables except X_i, which remains a random variable.

A brute-force calculation of $\Pr(X_i = k)$ involves enumerating all K^{n-1} possible combinations of the $n-1$ random variables, each of which is assumed to take one of the K discrete values. Since X_i can take K values, the marginal distribution requires $(K-1)K^{n-1}$ summations, each involves a product $\prod_{j=1}^{i} \psi_{j,j+1}(X_j, X_{j+1}) \prod_{j=i+1}^{n-1} \psi_{j,j+1}(X_j, X_{j+1})$. This is computational infeasible as n grows.

By exploiting the conditional independence, the above summations of products (sum-product) can be reduced to a series of products of summation (product-sum). The product-sum algorithm is a dynamic programming algorithm, and it will be generalized to trees later.

We can expand the sum-product calculation as follows.

[2]A simpler model is a graph without any edge, but that's not interesting

- First, the summation over all values of X_n involves the factor $\psi_{n-1,n}(X_{n-1}, X_n)$ only. The remaining factors can be pull out of the summation. We can evaluate $\mu_\beta(X_{n-1}) = \sum_{X_n} \psi_{n-1,n}(X_{n-1}, X_n)$, which is a function of X_{n-1} since only X_n is marginalized out. By multiplying $\mu_\beta(X_{n-1})$ with $\psi_{n-2,n-1}(X_{n-2}, X_{n-1})$, we obtain a function of (X_{n-2}, X_{n-1}).
- Next, we will conduct the summation over X_{n-1}. This summation only involves the product $\psi_{n-2,n-1}(X_{n-2}, X_{n-1})\mu_\beta(X_{n-1})$ and results in a function of X_{n-2}, denoted by $\mu_\beta(X_{n-2})$. Such multiplication of $\mu_\beta(X_j)$, called a *backward message*, and a potential $\psi_{j,j+1}(X_j, X_{j+1})$, and the marginalization of X_{j+1} continues up to the point of $\mu_\beta(X_i)$.
- We can also start from X_1 to compute the *forward message* $\mu_\alpha(X_2) = \sum_{X_1} \psi_{1,2}(X_1, X_2)$ by marginalizing out X_1. The multiplication of $\mu_\alpha(X_2)$ and $\psi_{1,2}(X_2, X_3)$ results in a function in (X_2, X_3), which will be marginalized over X_2 to obtain $\mu_\alpha(X_3)$. This continues until $\mu_\alpha(X_i)$ is calculated.
- Given $\mu_\beta(X_i)$ and $\mu_\alpha(X_i)$, we have marginalized out all random variables except X_i, and $\mu_\beta(X_i)\mu_\alpha(X_i)$ is just

$$\sum_{\setminus X_i} \prod_{j=1}^{i} \psi_{j,j+1}(X_j, X_{j+1}) \prod_{j=i+1}^{n-1} \psi_{j,j+1}(X_j, X_{j+1}). \tag{34}$$

How about the normalization factor Z? Since it is constant, it won't affect the evaluation of the messages. But Z is just

$$\sum_{X_1,\ldots,X_n} \prod_{j=1}^{n-1} \psi_{j,j+1}(X_j, X_{j+1}) = \sum_{X_i} \mu_\alpha(X_i)\mu_\beta(X_i), \tag{35}$$

since $\mu_\alpha(X_i)$ and $\mu_\beta(X_i)$ have done the difficult work of marginalizing out the remaining $n-1$ random variables.

As the above steps involves computing the forward and backward messages over the chain, the algorithm is known as "forward-backward" algorithm in the literature of hidden Markov models. It is also a special case of the "belief propagation" algorithm.

Compared to the sum-product algorithm, the time complexity of the product-sum algorithm is only $O((n-1)K^2)$ for computing $n-1$ messages, each involves the summation over K values of one random variable for each value of the other random variable. The reduction from exponential time complexity to linear time complexity is due to the messages caching the marginalization results that do not have to be recomputed over and over again.

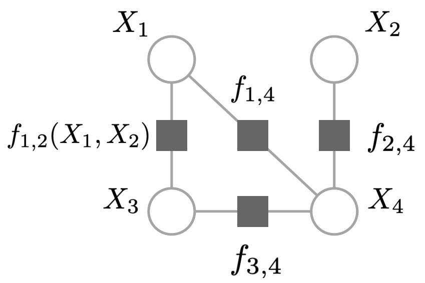
4.2 Inference over trees

Here we only consider the case when the tree is a undirected acyclic graph and any two nodes are connected by one and only one path. The directed graphs (or Bayesian networks) can be first be converted into an undirected graph using the so-called "moralization" step (Section 8.3.4. of PRML) and make inference from there.

4.2.1 Factor graphs

To make the inference more explicit, a new concept called "factor graphs" should be introduced. Note that factor graphs are only a computational tool and do not alternate the factorization nor the conditional independence properties of a joint distribution.

Given a MRF with the factorization of $\Pr(X)$ as in Eq. (21), the factor graph is a bipartite graph consisting of variable nodes and factor nodes. The variable nodes are the same as those in the MRF, and a factor node represents on factor/potential function in Eq. (21). There is no edge running between nodes of the same type, and factor nodes only connect to variable nodes, and vice versa. The factor node f_c for a potential function $\psi_c(X_c)$ is connected to all variables nodes in the clique c, and a variable node X_i is connected to all factor nodes f_c for any clique c that X_i participates (one random variable can be in multiple cliques. See Figure 6 for some examples).



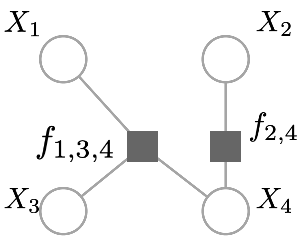


Figure 9: Two possible factor graphs for the MRF in Figure 6, depending on what factor/potential to used for the factorization of the joint distribution. *Left:* this factor graph corresponds to the factorization $\Pr(X_1, \ldots, X_4) = f_{1,3}(X_1, X_3) f_{1,4}(X_1, X_4) f_{3,4}(X_3, X_4) f_{2,4}(X_2, X_4)$, with Z being absorbed into one of the potentials. There are four factor nodes corresponding to the four potential functions. Note that this factor graph has a loop, as the corresponding MRF. *Right:* the factor graph corresponds to the factorization $\Pr(X_1, \ldots, X_4) = f_{1,3,4}(X_1, X_3, X_4) f_{2,4}(X_2, X_4)$ with two maximal cliques. The number of factor nodes is reduced to only two, but the factor node $f_{1,3,4}$ involves three variable nodes X_1, X_3, and X_4. This factor graph is a tree and has no cycle.

The normalization factor Z can be absorbed into one of the potential functions so that we just need a factor graph for the product of factors:

$$\Pr(X_1, \ldots, X_n) = \prod_c f_c(X_c), \tag{36}$$

where we let f_c to denote a potential function ψ_c.

Example 4.1. The MRF in Figure 6 can be turned into the factor graphs in Figure 9 for inference computation.

4.2.2 Inference on trees using belief propagation

We generalize the forward-backward algorithm to trees. Let's say we are going to evaluate $\Pr(X_i)$. X_i is connected to its neighboring factor nodes $\mathcal{N}(i) = \{s : X_i \text{ is connected to } s\}$ on the factor graph. The factor nodes in $\mathcal{N}(i)$ then partitions all the remaining factor and variables nodes into $|\mathcal{N}(i)|$ groups, each of which contains the remaining factor and variables nodes in the sub-tree $\mathcal{T}_s$ rooted at the factor node $s \in \mathcal{N}(i)$. See Figure 10, left panel. The potential functions in the factorization of $\Pr(X_1, \ldots, X_n)$ can be grouped based on this partition accordingly:

$$\Pr(X_1, \ldots, X_n) = \prod_{s \in \mathcal{N}(i)} \prod_{c \in \mathcal{T}_s} f_c(X_c), \tag{37}$$

The variable nodes in different sub-trees are independent of each other conditioned on X_i, since there is no other path going from one sub-tree to another sub-tree, except the path going through X_i. The marginalization over all random variables except X_i can be conducted within each sub-tree. The results are $|\mathcal{N}(i)|$ messages $\mu_{s \to i}(X_i)$ from $\mathcal{T}_s$:

$$\mu_{s \to i}(X_i) = \sum_{X_j \in \mathcal{T}_s \setminus \{X_i\}} \prod_{c \in \mathcal{T}_s} f_c(X_c) \tag{38}$$

The summation in the above equation is to enumerate all possible combinations of values of the random variables in the sub-tree $\mathcal{T}_s$, except X_i. The message $\mu_{s \to i}(X_i)$ is a function of X_i as it is the marginalization of every random variables in $\mathcal{T}_s$ except X_i. Intuitively, $\mu_{s \to i}(X_i)$ represents the "belief" of the sub-tree $\mathcal{T}_s$ about the distribution of X_i. Given the messages, the marginal $\Pr(X_i)$ is then the following simple product

$$\Pr(X_i) = \prod_{s \in \mathcal{N}(i)} \mu_{s \to i}(X_i). \tag{39}$$

This is similar to the forward-backward algorithm where the messages $\mu_\alpha(X_i)$ and $\mu_\beta(X_i)$ meet at the node X_i and help evaluate $\Pr(X_i)$.

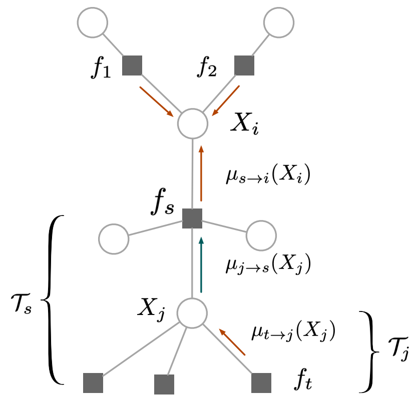


Figure 10: This factor graph demonstrates the derivation of the belief propagation algorithm from Eq. (39) to Eq. (44).
While here only X_i is selected as the root, the marginal distribution of all nodes can be computed by passing messages from X_i back to all other variable nodes.

Next we evaluate the message $\mu_{s\to i}(X_i)$. Let $\mathcal{N}(s)$ be the set of neighboring variable nodes of the factor node s on the factor graph. The variables nodes $X_j \in \mathcal{N}(s)$, excluding X_i, partition the factor and variable nodes in $\mathcal{T}_s$ into $|\mathcal{N}(s)|-1$ sub-trees, each of which roots at X_j. The partition leads to the following way of evaluating $\mu_{s\to i}(X_i)$:

$$\begin{aligned}
\mu_{s\to i}(X_i) &= \sum_{X_j\in\mathcal{T}_s\setminus\{X_i\}} \prod_{c\in\mathcal{T}_s} f_c(X_c) && (40)\\
&= \sum_{X_s\setminus\{X_i\}} f_s(X_s) \sum_{X_j\in\mathcal{T}_s\setminus X_s} \prod_{c\in\mathcal{T}_s\setminus\{s\}} f_c(X_c) && (41)\\
&= \sum_{X_s\setminus\{X_i\}} f_s(X_s) \prod_{j\in\mathcal{T}_s\setminus\{X_i\}} \left[\sum_{X_k\in\mathcal{T}_j\setminus\{X_j\}} \prod_{t\in\mathcal{T}_j} f_t(X_t)\right] && (42)\\
&= \sum_{X_s\setminus\{X_i\}} f_s(X_s) \prod_{j\in\mathcal{T}_s\setminus\{X_i\}} \mu_{j\to s}(X_j) && (43)\\
&= \sum_{X_s\setminus\{X_i\}} f_s(X_s) \prod_{j\in\mathcal{T}_s\setminus\{X_i\}} \left[\prod_{t\in\mathcal{N}(j)\setminus\{s\}} \mu_{t\to j}(X_j)\right] && (44)
\end{aligned}$$

- Going from Eq. (40) to Eq. (41), we use the fact that the potential function f_s is one of the f_c for $c \in \mathcal{T}_s$ in the product of Eq. (40) and we pull f_s out. This is just the same as $(ax + ay + bx + by) = a(x + y) + b(x + y)$. The second summation is to marginalize those random variables not taken care of in the first summation to complete the summation in Eq. (40).

- Going from Eq. (41) to Eq. (42), we partition the marginalization of the random variables $X_j \in \mathcal{T}_s \setminus X_s$ (the second summation in Eq. (41)) according to the sub-trees rooted at $X_j \in \mathcal{T}_s \setminus \{X_i\}$. In the bracket, the summation is to marginalize the random variables in the sub-tree rooted at X_j, except X_j.

- Eq. (43) is to introduce the definition of the message going from the variable node X_j to the factor node f_s: we just denote the summation in the bracket in Eq. (42) by $\mu_{j\to s}(X_j)$. If the set $\mathcal{T}_s \setminus \{X_i\}$ is empty, then the product is just the constant 1.

- Going from Eq. (43) to Eq. (44), we use the fact that marginalization over all random variables in the sub-tree $\mathcal{T}_j$, except X_j, is the product of the summations of variables within each branch of the root of $\mathcal{T}_j$. This is the same when we define $\mu_{s\to i}(X_i)$ in Eq. (39).

Based on the derivation, we can compute $\mu_{s\to i}(X_i)$ recursively: starting from the leaf nodes of $\mathcal{T}_s$, we can assume that all leaf nodes are factor nodes, since if a leave node is a variable node, the variable node can only pass a constant 1 message to its parent factor node, as in Eq. (43) when the set $\mathcal{T}_s \setminus \{X_i\}$ is empty.

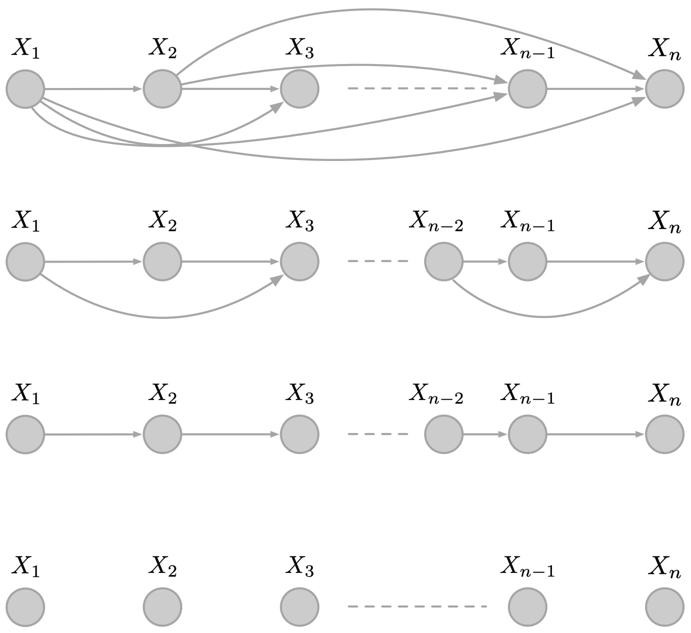


Figure 11: The distribution of sequential data $X_1, \ldots, X_n$ can be modeled using different graphical models, with different order of dependencies and number of parameters. The top one is a fully connected graphical model, with an exponential number of parameters to specify the distribution. The second one is a second-order Markov model, with X_i depends on the previous two variables X_{i-1} and X_{i-2}. The third one is a first-order Markov model (or just Markov model), with X_i depending on X_{i-1} only. The last model assumes that all n variables are independently distributed.

Then go one level up and multiply the messages from the leaf nodes $\mu_{t \to j}(X_j)$ to obtain the message $\mu_{j \to s}(X_j)$ that will be passed one level up to the parent of X_j, which is the factor node f_s.

Once f_s receives all messages from its children, it multiplies all messages and the potential function $f_s(X_s)$. The product is marginalized over all random variables except the parent of f_s, which is X_i. The result is the message $\mu_{s \to i}(X_i)$ passing to X_i. Similar to the case of X_j, when X_i receives all messages from its children factor nodes, it multiplies them and pass the product to the parent of X_i. This process is repeated until the root is reached, where the root must be a variable node and all incoming messages to the root is multiplied to obtain an un-normalized marginal distribution of the root. The normalization can be done locally at the root.

To find the marginal distribution of each X_i in an MRF, tt is not necessary to choose each variable node as the root and conduct the above message passing algorithm. Rather, any variable can be used a root and once all incoming messages to the root is obtained, all nodes, except the root, have received messages from all of its neighboring factor nodes except the one in the direction from the selected root. To find this missing message, the root can pass messages to its factor children, who will continue to pass messages to their children, until all variable nodes receive the message from the direction of the root. At this point, all variable nodes have all messages from their neighboring factors, and their marginal distributions can be computed as the root.

5 Hidden Markov Model (HMM)

Sequential data are abundant in many applications. For example, nucleic acid sequences are commonly studied in bioinformatics; the daily stock prices of a company is a sequence of real numbers; human languages are expressed by sentences that are sequences of words.

Graphical models are powerful tools for processing sequential data without the typical I.I.D. assumption. The joint distribution of a sequence of observed data is

$$\Pr(X_1, \ldots, X_n) = \Pr(X_1) \prod_{i=2}^{n} \Pr(X_i | X_1, \ldots, X_{i-1}) \tag{45}$$

The corresponding graphical model links all past data to the current data, whose distribution can

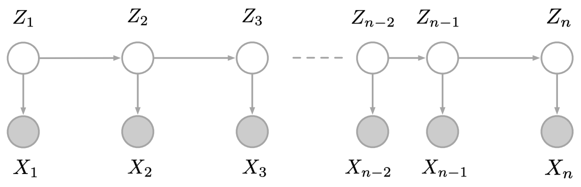


Figure 12: The graphical model for an HMM for observed sequential data $X_1, \ldots, X_n$.

be modeled as $\Pr(X_n|X_1, \ldots, X_n)$. However, this model has the number of dependencies grow quadratically in the number of past observations, and even worse, the number of parameters to fully specify the distribution $\Pr(X_n|X_1, \ldots, X_n)$ is exponential in n. A trade-off is to have X_n depend on the recent past data only $\Pr(X_n|X_1, \ldots, X_{n-1}) = \Pr(X_n|X_{n-t}, \ldots, X_{n-1})$, and in the extreme case, the immediate past $\Pr(X_n|X_1, \ldots, X_{n-1}) = \Pr(X_n|X_{n-1})$ (the Markov model). These graphical models are shown in Figure 11. As some of the earlier data will not be factored in the model for X_n due to d-separation, there can be a loss of information if X_i indeed depends on the earlier data in the farther past. It is in general hard to know how far a model should look into the past.

5.1 HMM

HMM is invented to addressed the above dilemma. That is, X_n is distributed depending on all history, using a number of parameter independent of the length of the history. HMM does this by introducing a hidden variable Z_n for each observed variable X_n. The graphical model of HMM is shown in Figure 12 and the corresponding factorization of the joint distribution of the observed data and hidden variables is

$$\Pr(X_1, \ldots, X_n, Z_1, \ldots, Z_n) = \Pr(Z_1)\Pr(X_1|Z_1) \prod_{i=2}^{n} \Pr(Z_i|Z_{i-1})\Pr(X_i|Z_i). \tag{46}$$

That is, the observed data are generated by first sampling the hidden variable Z_1, then generating the observation X_1 according to $\Pr(X_1|Z_1)$. At step $i = 2, \ldots, n$, the hidden variable is sampled from the distribution $\Pr(Z_i|Z_{i-1})$ and X_i is sampled from $\Pr(X_i|Z_i)$. For simplicity, we assume that Z_i are discrete random variables taking values $\{1, \ldots, K\}$ (called "states" in the more general state-space models). The observed random variables X_i can be continuous or discrete. Let's explain each of these factors.

- Starting probability $\Pr(Z_1 = k|\boldsymbol{\pi}) = \pi_k$: Z_1 takes value $k \in \{1, \ldots, K\}$ with probability π_k. Therefore, we must have $\pi_k \geq 0$ and $\sum_k \pi_k = 1$. There are $K - 1$ parameters for the distribution of Z_1.

- Transition probability $\Pr(Z_i|Z_{i-1}; A)$ for $i = 2, \ldots, n$: this specifies how Z_i depends on Z_{i-1} and $\Pr(Z_i = k|Z_{i-1} = j) = A_{kj}$, $k, j = 1, \ldots, K$. Then we have $\sum_j A_{kj} = 1$ and $A_{kj} \geq 0$. The $K \times K$ matrix A is called "transition matrix".

  We can see that the hidden variables $Z_1, \ldots, Z_n$ are modeled as a first-order Markov model. The transition according to A can be represented as a transition diagram or trellis diagrams. See Figure 13.6 and 13.7 of PRML for an example.

- Emission probability $\Pr(X_i|Z_i; \boldsymbol{\phi})$ for $i = 1, \ldots, n$ specifies how the observed X_i is distributed according to Z_i with parameters $\boldsymbol{\phi}$. If X_i is also discrete and takes value from $\{1, \ldots, D\}$, then we have $\Pr(X_i = d|Z_i = k) = \mu_{kd} \geq 0$ and $\sum_d \mu_{kd} = 1$ for any k. There are $K \times (D-1)$ μ_{kd} parameters to specify $\Pr(X_i = d|Z_i = k)$. If X_i is continuous, such as a Gaussian-distributed vector, then $\Pr(X_i|\boldsymbol{\mu}_k, \Sigma_k)$ is a Gaussian density with mean $\boldsymbol{\mu}_k$ and covariance matrix Σ_k. The parameters are then $\boldsymbol{\mu}_k$ and Σ_k, $k = 1, \ldots, K$. We let $\boldsymbol{\phi}$ be the collection of parameters for the emission probability distributions.

Let the collection of parameters be denoted by $\boldsymbol{\theta} = (\boldsymbol{\pi}, \boldsymbol{\phi}, A)$. Eq. (46) can be written as

$$\Pr(X_1, \ldots, X_n, Z_1, \ldots, Z_n|\boldsymbol{\theta}) = \Pr(Z_1|\boldsymbol{\pi})\Pr(X_1|Z_1; \boldsymbol{\phi}) \prod_{i=2}^{n} \Pr(Z_i|Z_{i-1}; A)\Pr(X_i|Z_i; \boldsymbol{\phi}). \tag{47}$$

Example 5.1 (POS tagging in NLP)**.** HMM with discrete hidden and observed variables can be used to model the distribution of a sentence, assuming a Markov model in the transition of part-of-speech (POS) tags of the language. For example, in English, the most common POS tags are `Noun, Verb, Adjective, Adverb, Determiner`. Human language grammar specifies how likely one POS tag $Z_i = k$ can follow another tag $Z_{t-1} = j$, and this is modeled by the transition probability A_{jk}. For instance, the probability of seeing a `Verb` after a `Noun` is higher than seeing an `Adjective`. Given a POS tag Z_i, a word X_i can be generated according to the emission probability $\Pr(X_i|Z_i)$, with $X_i = d$ meaning that the i-th word of the sentence is the d-th word in the vocabulary. Lastly, a sentence starts with a POS tag with various probability according to $\pi(Z_1)$, such as a determiner is more likely to start a sentence than an adjective.

Example 5.2 (Continuous observations)**.** One such example is still from NLP. Rather than observing words in a sentence, one observes spectrogram of spoken sentences that represent the frequency patterns of human speech. The hidden variables Z_i are words and the probability of observing a spectrogram pattern given a word is specified by a Gaussian distribution. This HMM was used in speech recognition as early as 1970's.

HMM can be regarded as a mixture model, so that an observation X_i is generated from a mixture of K possible underlying distributions $\boldsymbol{\phi}_k$, $k = 1, \ldots, K$. For example, in the POS tagging example, the word X_i="*check*" can be generated from Z_i =`Verb` or `Noun`. The selection of a mixture component at step i depends on the mixture component at the last step according to $\Pr(Z_i|Z_{i-1})$. For example, seeing Z_{i-1} =`Noun` will increase the likelihood of seeing Z_{i-1} =`Verb`.

5.2 Learning HMM

There are two main tasks when modeling sequential data using an HMM. The first task is called "learning", where parameters $\boldsymbol{\theta} = (\boldsymbol{\pi}, \boldsymbol{\phi}, A)$ needs to be learned from training data. The second task is called "inference", where the necessary probabilities need to be computed given a fixed set of parameters $\boldsymbol{\theta}$. We discuss the learning task here and leave the inference task to the next section.

Maximum likelihood estimation (MLE) can be used to estimate the parameters of the HMM. The incomplete-data likelihood of the observed data $X_1, \ldots, X_n$ given $\boldsymbol{\theta}$ is

$$\Pr(X_1, \ldots, X_n|\boldsymbol{\theta}) = \sum_{Z_1,\ldots,Z_n} \Pr(X_1, \ldots, X_n, Z_1, \ldots, Z_n|\boldsymbol{\theta}) \tag{48}$$

The marginalization of the hidden variables $Z_1, \ldots, Z_n$ makes the maximization difficult for the same reason as in learning mixture models: the log operator acts on the summation of probabilities, so that taking the partial derivative of the incomplete-data log-likelihood with respect to $\boldsymbol{\theta}$ will not allow a closed-form solution of $\boldsymbol{\theta}$.

Like mixture models, one can use the EM algorithm to learn $\boldsymbol{\theta}$. In particular, the following lower bound of the incomplete-data likelihood $\Pr(X_1, \ldots, X_n|\boldsymbol{\theta})$ can be maximized to improve $\boldsymbol{\theta}$ from the previous estimation $\boldsymbol{\theta}^{\text{old}}$:

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}; X) = \sum_{Z_1,\ldots,Z_n} \Pr(Z_1, \ldots, Z_n|X_1, \ldots, X_n; \boldsymbol{\theta}^{\text{old}}) \ln \Pr(X_1, \ldots, X_n, Z_1, \ldots, Z_n|\boldsymbol{\theta}). \tag{49}$$

Now the log operator acts on $\Pr(X_1, \ldots, X_n, Z_1, \ldots, Z_n|\boldsymbol{\theta})$ and a closed-form estimation of $\boldsymbol{\theta}$ can be obtained. In particular,

$$\ln \Pr(X_1, \ldots, X_n, Z_1, \ldots, Z_n|\boldsymbol{\theta}) = \ln \Pr(Z_1|\boldsymbol{\theta}) + \sum_{i=2}^{n} \ln \Pr(Z_i|Z_{i-1}; \boldsymbol{\theta}) + \sum_{i=1}^{n} \ln \Pr(X_i|Z_i; \boldsymbol{\theta}), \tag{50}$$

according to the graphical model of HMM (or equivalently the factorization Eq. (46)).

The marginalization in Eq. (49) can then be computed over each of the terms in the summation easily, because the joint distribution of $Z_1, \ldots, Z_n$ over which the expectation is taken is parametried by a fixed parameter $\boldsymbol{\theta}^{\text{old}}$, and each term involves only a small number of hidden variables (at most two). In particular,

$$\sum_{Z_1,\ldots,Z_n} \Pr(Z_1, \ldots, Z_n|X_1, \ldots, X_n; \boldsymbol{\theta}^{\text{old}}) \ln \Pr(Z_1|\boldsymbol{\pi}) = \sum_{k=1}^{K} \gamma_1(k) \ln \pi_k, \tag{51}$$

where $\gamma_i(k) = \Pr(Z_i = k|\boldsymbol{\theta}^{\text{old}})$, which depends on the observed data $X_1, \ldots, X_n$ (not shown).

$$\sum_{Z_1,\ldots,Z_n} \Pr(Z_1, \ldots, Z_n|X_1, \ldots, X_n; \boldsymbol{\theta}^{\text{old}}) \sum_{i=2}^{n} \ln \Pr(Z_i|Z_{i-1}; \boldsymbol{\theta}) = \sum_{i=1}^{n} \sum_{k=1}^{K} \sum_{j=1}^{K} \xi_i(j,k) \ln A_{jk}, \tag{52}$$

where $\xi_i(j,k) = \Pr(Z_{i-1} = j, Z_i = k|\boldsymbol{\theta}^{\text{old}})$, which also depends on $X_1, \ldots, X_n$ (not shown).

$$\sum_{Z_1,\ldots,Z_n} \Pr(Z_1, \ldots, Z_n|X_1, \ldots, X_n; \boldsymbol{\theta}^{\text{old}}) \sum_{i=1}^{n} \ln \Pr(X_i|Z_i; \boldsymbol{\theta}) = \sum_{i=1}^{n} \sum_{k=1}^{K} \gamma_i(k) \ln \Pr(X_i|Z_i = k; \boldsymbol{\phi}). \tag{53}$$

Assuming that γ_i and ξ_i are known, then the new estimation of the parameters are

- Starting probability:

$$\pi_k = \frac{\gamma_1(k)}{\sum_{k'=1}^{K} \gamma_1(k')} \tag{54}$$

- Transition probability:

$$A_{jk} = \frac{\sum_{i=1}^{n} \xi_i(j,k)}{\sum_{k=1}^{K} \sum_{i=1}^{n} \xi_i(j,k)} \tag{55}$$

- Emission probability: depending on the emission model. If $\Pr(X_i = d|Z_i = k) = \mu_{kd}$ is the distribution a discrete X_i taking values from $\{1, \ldots, D\}$ (e.g., the POS-tagging problem), then

$$\mu_{kd} = \frac{\sum_{i=1}^{n} \gamma_i(k) \mathbb{1}[X_i = d]}{\sum_{i=1}^{n} \gamma_i(k)}. \tag{56}$$

  If X_i is continuous random vector and $\Pr(X_i|Z_i = k)$ is a Gaussian density with mean μ_k and covariance Σ_k (e.g., the speech recognition problem), then

$$\boldsymbol{\mu}_k = \frac{\sum_{i=1}^{n} \gamma_i(k) X_i}{\sum_{i=1}^{n} \gamma_i(k)}, \tag{57}$$

$$\Sigma_k = \frac{\sum_{i=1}^{n} \gamma_i(k)(X_i - \boldsymbol{\mu}_k)^\top (X_i - \boldsymbol{\mu}_k)}{\sum_{i=1}^{n} \gamma_i(k)}. \tag{58}$$

5.3 Inference in HMM

Inference of HMM includes computing various probabilities using the HMM factorization and predicting the most likely values of the sequence of hidden variables.

First, the probability $\Pr(Z_i = k|X_1, \ldots, X_n; \boldsymbol{\theta}^{\text{old}})$ must be inferred for parameter estimation. Since $\boldsymbol{\theta}^{\text{old}}$ is fixed, for simplicity we ignore $\boldsymbol{\theta}^{\text{old}}$ and find $\Pr(Z_i = k|X_1, \ldots, X_n)$:

$$\Pr(Z_i = k|X_1, \ldots, X_n) = \frac{\Pr(X_1, \ldots, X_n, Z_i = k)}{\Pr(X_1, \ldots, X_n)}. \tag{59}$$

Using d-separation of the graphical model of HMM (Figure 12), we can see that $X_1, \ldots, X_{i-1}$ are independent of $X_{i+1}, \ldots, X_n$ conditioned on $Z_i = k$ and

$$\begin{aligned} \Pr(Z_i = k|X_1, \ldots, X_n) &= \frac{\Pr(X_1, \ldots, X_i, Z_i = k)\Pr(X_{i+1}, \ldots, X_n|Z_i = k)}{\Pr(X_1, \ldots, X_n)} && (60) \\ &= \frac{\alpha_i(k)\beta_i(k)}{\Pr(X_1, \ldots, X_n)}, && (61) \end{aligned}$$

where $\boldsymbol{\alpha}_i = [\alpha_i(1), \ldots, \alpha_i(K)]$ and $\boldsymbol{\beta}_i = [\beta_i(1), \ldots, \beta_i(K)]$ are the forward and backward messages, respectively.

- Computing $\boldsymbol{\alpha}_i$ recursively: for each $\alpha_i(k)$, $k = 1, \ldots, K$,

$$\begin{aligned}
\alpha_i(k) &= \Pr(X_1, \ldots, X_i, Z_i = k) && (62)\\
&= \Pr(Z_i = k)\Pr(X_1, \ldots, X_i | Z_i = k) && (63)\\
&= \Pr(Z_i = k)\Pr(X_1, \ldots, X_{i-1} | Z_i = k)\Pr(X_i | Z_i = k) && (64)\\
&= \Pr(Z_i = k)\sum_{j=1}^{K}\Pr(X_1, \ldots, X_{i-1}, Z_{i-1} = j | Z_i = k)\Pr(X_i | Z_i = k) && (65)\\
&= \sum_{j=1}^{K}\Pr(Z_i = k)\Pr(X_1, \ldots, X_{i-1}, Z_{i-1} = j | Z_i = k)\Pr(X_i | Z_i = k) && (66)\\
&= \sum_{j=1}^{K}\Pr(X_1, \ldots, X_{i-1}, Z_{i-1} = j, Z_i = k)\Pr(X_i | Z_i = k) && (67)\\
&= \sum_{j=1}^{K}\Pr(X_1, \ldots, X_{i-1}, Z_{i-1} = j)\Pr(Z_i = k | Z_{i-1} = j)\Pr(X_i | Z_i = k) && (68)\\
&= \sum_{j=1}^{K}\alpha_{i-1}(j)A_{jk}\Pr(X_i | Z_i = k) && (69)
\end{aligned}$$

  The base condition is

$$\alpha_1(k) = \Pr(X_1, Z_1 = k) = \pi_k \Pr(X_1 | Z_1 = k). \qquad (70)$$

  Note that computing $\alpha_i(k)$ requires the availability of the parameters $\boldsymbol{\theta}^{\text{old}}$.
  Since $\alpha_n(k) = \Pr(Z_n = k, X_1, \ldots, X_n)$, $\Pr(X_1, \ldots, X_n) = \sum_{k=1}^{K}\alpha_n(k)$.
  Computing all forward messages requires $O(nK^2)$ time complexity.

- Computing $\boldsymbol{\beta}_i$ recursively: for each $\beta_i(k)$, $k = 1, \ldots, K$,

$$\begin{aligned}
\beta_i(k) &= \Pr(X_{i+1}, \ldots, X_n | Z_i = k) && (71)\\
&= \sum_{j=1}^{K}\Pr(X_{i+1}, \ldots, X_n, Z_{i+1} = j | Z_i = k) && (72)\\
&= \sum_{j=1}^{K}\Pr(Z_{i+1} = j | Z_i = k)\Pr(X_{i+1}, \ldots, X_n | Z_{i+1} = j) && (73)\\
&= \sum_{j=1}^{K}\Pr(Z_{i+1} = j | Z_i = k)\Pr(X_{i+2}, \ldots, X_n | Z_{i+1} = j)\Pr(X_{i+1} | Z_{i+1} = j) && (74)\\
&= \sum_{j=1}^{K}A_{kj}\beta_{i+1}(j)\Pr(X_{i+1} | Z_{i+1} = j) && (75)
\end{aligned}$$

  The base condition is

$$\beta_n(k) = \frac{\Pr(Z_n = k | X_1, \ldots, X_n)\Pr(X_1, \ldots, X_n)}{\alpha_n(k)} = 1, \qquad (76)$$

  since by definition, $\alpha_n(k) = \Pr(Z_n = k, X_1, \ldots, X_n)$.
  Similar to forward message computing, computing the backward messages takes $O(nK^2)$ time complexity.

To complete the parameter learning algorithm, the probability $\xi_i(j,k) = \Pr(Z_{i-1} = j, Z_i = k | X_1, \ldots, X_n)$ also needs to be computed. But this is easy when the forward and backward messages are computed:

$$\begin{aligned}
\xi_i(j,k) &= \Pr(Z_{i-1}=j, Z_i=k|X_1,\ldots,X_n) && (77)\\
&= \frac{\Pr(Z_{i-1}=j, Z_i=k, X_1,\ldots,X_n)}{\Pr(X_1,\ldots,X_n)} && (78)\\
&= \frac{\Pr(X_1,\ldots,X_{i-1},Z_{i-1}=j)\Pr(Z_i=k|Z_{i-1}=j)\Pr(X_i|Z_i=k)\Pr(X_{i+1},\ldots,X_n|Z_i=k)}{\Pr(X_1,\ldots,X_n)}\\
&= \frac{\alpha_{i-1}(j)\Pr(Z_i=k|Z_{i-1}=j)\Pr(X_i|Z_i=k)\beta_i(k)}{\Pr(X_1,\ldots,X_n)} && (79)
\end{aligned}$$

Now we have the sufficient information to learn the parameters $\boldsymbol{\theta}$ of the HMM given the observed sequential data $(X_1,\ldots,X_n)$ After initialize $\boldsymbol{\theta}$, forward and backward messages are computed using Eq. (69) and (75) with the fixed $\boldsymbol{\theta}$. This is the E-step of the EM algorithm. Then the inferred probabilities γ and ξ are used in Eq. (54), (55), and (56) (for discrete X_i) (or (57) and (58) for continuous X_i) to estimate a new $\boldsymbol{\theta}$. This is the M-step of the EM algorithm. The two steps alternate until the model parameters do not change significantly.

Once an HMM is learned, that is, $\boldsymbol{\theta}$ stabilizes, then one wants to use the model to make prediction. That is, to find a sequence of values for $(Z_1,\ldots,Z_n)$ so that $\Pr(Z_1,\ldots,Z_n|X_1,\ldots,X_n)$ is maximized. It is tempting to estimate $\Pr(Z_i=k|X_1,\ldots,X_n)$ using forward and backward messages and take the k^* that maximizes $\Pr(Z_i=k|X_1,\ldots,X_n)$ for each $i=1,\ldots,n$. However, there can be situations that maximizing $\Pr(Z_i=k|X_1,\ldots,X_n)$ at individual locations $i=1,\ldots,n$ will not lead to the maximum probability $\max_{Z_1,\ldots,Z_n}\Pr(Z_1,\ldots,Z_n|X_1,\ldots,X_n)$.

The Viterbi algorithm is a specicial case of the max-product algorithm. The algorithm replaces the sum with the max operator in the forward message calculation:

$$v_i(k) = \max_j[v_{i-1}(j)A_{jk}\Pr(X_i|Z_i=k)]. \tag{80}$$

As a result, $v_i(k)$ records which of the previous state j that leads to the maximum probability of $\Pr(X_1,\ldots,X_i,Z_i=k)$, when the state of Z_i is k. The base condition is $v_1(k)=\pi_k\Pr(X_1|Z_1=k)$. When $v_n(k)$ is reached, one has the information to decide which state of Z_n leads to the maximum $\Pr(X_1,\ldots,X_n)$:

$$Z_n^* = \arg\max_k v_n(k). \tag{81}$$

The decision at each step $i=2,\ldots,n$ can be cached in an additional array prev so that

$$\text{prev}_i(k) = \arg\max_j[v_{i-1}(j)A_{jk}\Pr(X_i|Z_i=k)], \tag{82}$$

so that $\text{prev}_i(k)$ stores the best state of Z_{i-1} that leads to the maximum $\Pr(X_1,\ldots,X_i,Z_i=k)$. The array can be used to recover the optimal sequence Z_i by

$$Z_{i-1}^* = \text{prev}_i(Z_i^*), i=2,\ldots,n. \tag{83}$$

6 Conditional Random Field (CRF)

HMM is a generative model using the distribution $\Pr(X_1,\ldots,X_n,Z_1,\ldots,Z_n)$, with the conditional independence $X_i \not\perp\!\!\!\perp X_j|Z_i$ and $X_i \not\perp\!\!\!\perp Z_j|Z_i$for any $i\neq j$. This makes more complex dependencies difficult to model. For example, the word X_i at position i depends not just on Z_i, the POS-tag at position i, but also the contextual words around position i, and the POS-tag Z_i can also depends on future words. While more edges can be added to the HMM graphical model to allow such dependencies, the model can eventually become intractable.

By giving up the capability of generating observed sequences, such as English sentences, conditional random field, as a discriminative model, can be used for predicting the hidden variables $Z_1,\ldots,Z_n$ given observed sequential data. The relationship between HMM and CRF is similar to the relationship between naive Bayes and logistic regression (see Figure 7. Naive Bayes is generative since it estimates $\Pr(X_i|Y)$, with the assumption $X_i \not\perp\!\!\!\perp X_j|Y$ to simplify the generation model, while logistic regression is discriminative as it directly estimates $\Pr(Y|X_1,\ldots,X_n)$, leaving the dependencies between X_i and X_j to the model parameters $(X_i \not\perp\!\!\!\perp X_j)$ if Y is not observed, using d-separation).

CRF models the distribution

$$\Pr(Z_1,\dots,Z_n|X_1,\dots,X_n) = \frac{1}{Z}\prod_{i=1}^{n}\psi_i(Z_i,Z_{i-1},X_i) = \frac{1}{Z}\prod_{i=1}^{n}\exp\left\{\sum_{j=1}^{D}\theta_j f_j(Z_i,Z_{i-1},X_i)\right\}. \tag{84}$$

We let Z_{-1} be a special constant starting symbol so that we don't need to worry about the special case $i=1$. The normalization factor

$$Z = \sum_{Z_1,\dots,Z_n}\prod_{i=1}^{n}\psi_i(Z_i,Z_{i-1},X_i). \tag{85}$$

The factor $\psi_i(Z_i,Z_{i-1},X_i)$ is parametrized by the vector $\boldsymbol{\theta}=(\theta_1,\dots,\theta_D)$, and each θ_j corresponds to one feature function $f_d(Z_j,Z_{j-1},X_j)$ that models the dependencies among Z_j,Z_{j-1}, and X_j.

We can see that HMM is related to CRF by the following conversion of HMM joint distribution.

$$\begin{aligned}
& \Pr(X_1,\dots,X_n,Z_1,\dots,Z_n)\\
= & \prod_{i=1}^{n}\Pr(Z_i|Z_{i-1})\Pr(X_i|Z_i)\\
= & \frac{1}{Z}\prod_{i=1}^{n}\exp\left\{\sum_{k,j}\log(Z_i|Z_{i-1})\mathbb{1}[Z_i=k,Z_{i-1}=j]+\sum_{k,d}\log\Pr(X_i|Z_i)\mathbb{1}[X_i=d,Z_i=k]\right\}\\
= & \frac{1}{Z}\prod_{i=1}^{n}\exp\left\{\sum_{k,j}\theta_{jk}\mathbb{1}[Z_i=k,Z_{i-1}=j]+\sum_{k,d}\theta_{kd}\mathbb{1}[X_i=d,Z_i=k]\right\}\\
= & \frac{1}{Z}\prod_{i=1}^{n}\exp\left\{\sum_{k,j}\theta_{jk}f_{jk}(Z_i,Z_{i-1})+\sum_{k,d}\theta_{kd}f_{kd}(Z_i,X_i)\right\}.
\end{aligned}$$

$Z=1$ since each local factor of HMM is a distribution. The feature functions are indicator functions detecting patterns in the values assigned to (Z_i,X_i) and (Z_{i-1},Z_i) for all location i. The parameters are $\boldsymbol{\theta}=\{\theta_{jk},\theta_{kd}:j,k=1,\dots,K,d=1,\dots,D\}$.

CRF can use more complex feature functions beyond $f_{jk}(Z_i,Z_{i-1})$ and $f_{kd}(Z_i,X_i)$. Designing the feature function is called "feature engineering" and is domain-specific.

Example 6.1 (POS tagging feature functions)**.** CRF has been used to prediction the POS-tags of a sentence. A feature function can be

- $\mathbb{1}[Z_i=\text{Verb}]\mathbb{1}[Z_{i-1}=\text{Noun}]\mathbb{1}[X_i=\text{“go”}]$.
- $\mathbb{1}[Z_{i+1}=\text{Verb}]\mathbb{1}[Z_i=\text{Verb}]\mathbb{1}[Z_{i-1}=\text{Noun}]$.
- $\mathbb{1}[Z_{i+1}=\text{Verb}]\mathbb{1}[X_i=\text{“Him”}]$.

You can see that there is much more flexibility in the dependencies among multiple observed and hidden variables. In fact, the feature function can depend on the entire observed sequence.

6.1 Inference and parameter learning for CRF

The inference of CRF is to calculate the probabilities $\Pr(Z_i|X_1,\dots,X_n)$ and $\Pr(Z_{i-1},Z_i|X_1,\dots,X_n)$ for parameter learning. The forward-backward algorithm for HMM can be applied to CRF without much change, due to the similarity between Eq. (46) and Eq. (84). In particular,

$$\alpha_i(k) \propto \sum_{j=1}^{K}\alpha_{i-1}(j)\psi_i(Z_i=k,Z_{i-1}=j,X_i), \tag{86}$$

$$\beta_i(k) \propto \sum_{j=1}^{K}\beta_{i+1}(j)\psi(Z_{i+1}=j,Z_i=k,X_{i+1}). \tag{87}$$

The $\propto$ signs above are necessary since the summations on the right-hand-sides are not normalized probability distributions about Z_i. The interpretations of α_i and β_i are not $\Pr(X_1, \ldots, X_i, Z_i)$ and $\Pr(X_{i+1}, \ldots, X_n | Z_i)$, respectively since CRF is not a generative model.

To learn the parameter $\boldsymbol{\theta}$ of CRF, an EM algorithm can be employed as well. In the E-step, the above forward-backward inference finds the following marginal probability

$$\Pr(Z_i = k, Z_{i-1} = j | X_1, \ldots, X_n) \propto \alpha_i(k)\beta_{i+1}(j)\psi(Z_{i+1} = j, Z_i = k, X_{i+1}). \tag{88}$$

In the M-step, we perform an MLE of $\boldsymbol{\theta}$ by maximizing the log-likelihood

$$\begin{aligned} \ell(X_1, \ldots, X_n, Z_1, \ldots, Z_n; \boldsymbol{\theta}) &= \log \Pr(Z_1, \ldots, Z_n | X_1, \ldots, X_n; \boldsymbol{\theta}) & (89) \\ &= \sum_{i=1}^{n} \sum_{j=1}^{D} \theta_j f_j(Z_i, Z_{i-1}, X_i) - \log Z. & (90) \end{aligned}$$

Based on Eq. (84), we have the partial derivative of ℓ with respect to a single parameter θ_j:

$$\begin{aligned} \frac{\partial \ell}{\partial \theta_j} &= \sum_{i=1}^{n} f_j(Z_i, Z_{i-1}, X_i) - \frac{1}{Z} \sum_{Z_1,\ldots,Z_n} \frac{\partial}{\partial \theta_j} \prod_{i=1}^{n} \psi_i(Z_i, Z_{i-1}, X_i) & (91) \\ &= \sum_{i=1}^{n} f_j(Z_i, Z_{i-1}, X_i) - \frac{1}{Z} \sum_{Z_1,\ldots,Z_n} \sum_{i=1}^{n} \prod_{i=1}^{n} \psi_i(Z_i, Z_{i-1}, X_i) f_j(Z_i, Z_{i-1}, X_i) & (92) \\ &= \sum_{i=1}^{n} f_j(Z_i, Z_{i-1}, X_i) - \sum_{Z_1,\ldots,Z_n} \sum_{i=1}^{n} \frac{\prod_{i=1}^{n} \psi_i(Z_i, Z_{i-1}, X_i)}{Z} f_j(Z_i, Z_{i-1}, X_i) & (93) \\ &= \sum_{i=1}^{n} f_j(Z_i, Z_{i-1}, X_i) - \sum_{i=1}^{n} \sum_{Z_1,\ldots,Z_n} \Pr(Z_1, \ldots, Z_n | X_1, \ldots, X_n) f_j(Z_i, Z_{i-1}, X_i) & (94) \\ &= \sum_{i=1}^{n} f_j(Z_i, Z_{i-1}, X_i) - \sum_{i=1}^{n} \Pr(Z_i = k, Z_{i-1} = j | X_1, \ldots, X_n) f_j(Z_i, Z_{i-1}, X_i). & (95) \end{aligned}$$

The last equation shows that the partial derivative is the difference between the empirical mean of the feature function f_j, evaluated on the given training sequence (both X_i and Z_i are known for all $i = 1, \ldots, n$), and the mean of f_j under the model distribution $\Pr(Z_i = k, Z_{i-1} = j | X_1, \ldots, X_n)$.

A gradient ascent step can be taken to update θ_j:

$$\theta_j \leftarrow \theta_j + \eta \frac{\partial \ell}{\partial \theta_j}, \quad i = 1, \ldots, D. \tag{96}$$

where η is the learning rate. Regularization, such as $\ell - 2$ or $\ell - 1$ regularization can be added to the log-likelihood Eq. (89).