3

Representation

Our journey in probabilistic graphical modeling begins with the topic of *representation*. How do we choose a probability distribution to model some interesting aspect of the world? Coming up with a good model is not always easy: we have seen in the introduction that a naive model for spam classification would require us to specify a number of parameters that is exponential in the number of words in the English language. In this chapter, we will show that we can instead leverage the *conditional independencies* present in our distribution to represent the joint distribution with fewer parameters. Thus, we can obtain compact, tractable models for many complex systems. Further, we can express the structure of our joint as a *graphical model*: a mathematical object that visually encodes our conditional independence assumptions and enables the use of efficient graph-based algorithms for statistical inference and learning. Graphical models provide an intuitive visual language for reasoning over our system of interest, where complex computations used in learning and inference can be expressed as operations on the graph.

In this chapter, we will focus on effective and general techniques for parameterizing probability distributions with relatively few parameters. We will explore how the resulting models can be elegantly described as

directed and *undirected* graphs (Figure 2.4). We will explore connections between graphical structures and the assumptions made by the distributions that they describe. This will both make our modeling assumptions more explicit and help us design more efficient inference algorithms. We will discuss how the properties of different graphical representations can present advantages and shortcomings in particular settings, with a focus on Bayesian networks (Section 3.1.1), Markov random fields (Section 3.2.1), factor graphs (Section 3.2.2), and conditional random fields (Section 3.2.3). As a general accompaniment to this chapter, we recommend reading Chapters 3 and 4 in Koller and Friedman (2009) and Chapter 8 in Bishop (2006).

3.1 Representing Distributions with Directed Graphs

3.1.1 Bayesian Networks

We begin with directed graphical models: a powerful class of graphs defined foremost by the fact that their edges have directionality. While directed graphs can contain cycles, we will restrict our attention to those that do not. Directed graphs that do not contain cycles are known as *directed acyclic graphs* (DAGs; e.g., Figure 2.4b,c). In probabilistic graphical modeling, these are used in models known as *Bayesian networks*, *Bayes nets*, or *belief networks*. In causal modeling, these are used in models often referred to as *causal DAGs*.

For now, we will primarily focus on the probabilistic interpretation of DAGs using the term *Bayesian networks*. The power of Bayesian networks lies in their ability to enable reasoning in the presence of uncertainty. The following is just a short list of their useful properties.

1. Bayesian networks are *economical*: they allow us to represent probability distributions using compact parameterizations.

2. Unlike undirected graphs, Bayesian networks can admit both probabilistic and causal interpretations under sufficient conditions. Thus, Bayesian networks can potentially enable associational, interventional, and counterfactual insights (Pearl, 1995b).

3. Though Bayesian networks are not *innately* Bayesian, they can be combined with Bayesian statistical methods to elegantly incorporate prior knowledge and techniques for preventing overfitting (Heckerman, 1998).

Bayesian Networks as Economical Models: Intuition

How do Bayesian networks model probability distributions? What makes these representations *concise* or *economical*? We will begin by providing some intuition.

The general idea behind the compact parameterization of Bayesian networks is surprisingly simple. Recall that by the chain rule (Definition 2.28), we can write any probability p as

$$p(x_1, x_2, \ldots, x_n) = p(x_1)p(x_2 \mid x_1) \cdots p(x_n \mid x_{n-1}, \ldots, x_2, x_1).$$

A compact Bayesian network is a distribution in which each factor on the right hand side depends only on a set of parent variables, denoted $\mathbf{pa}(x_i)$:

$$p(x_i \mid x_{i-1}, \ldots, x_1) = p(x_i \mid \mathbf{pa}(x_i)).$$

Often, the cardinality of $\mathbf{pa}(x_i)$ will be significantly less than n. For example, consider a model with five variables where node x_5 has parents $\mathbf{pa}(x_5) = \{x_4, x_3\}$. We can choose to write the factor $p(x_5 \mid x_4, x_3, x_2, x_1)$ as $p(x_5 \mid x_4, x_3)$ instead.

When the variables are discrete (as is often the case in the problems we will discuss), we can think of the factors $p(x_i \mid \mathbf{pa}(x_i))$ as *probability tables* in which rows correspond to assignments to $\mathbf{pa}(x_i)$, columns correspond to values of x_i, and entries contain the probabilities $p(x_i \mid \mathbf{pa}(x_i))$ (Figure 3.1). If each variable takes d values and has at most k parents, then the entire table will contain at most $O(d^{k+1})$ entries. Since we have one table per variable, the entire probability distribution can be compactly described with only $O(nd^{k+1})$ parameters. Recall that we required $O(d^n)$ parameters under the naive approach.

An Illustrative Example Consider a model of student performance (Koller and Friedman, 2009). In reality, student performance is determined by complex social dynamics. For ease of exposition, we will pretend that

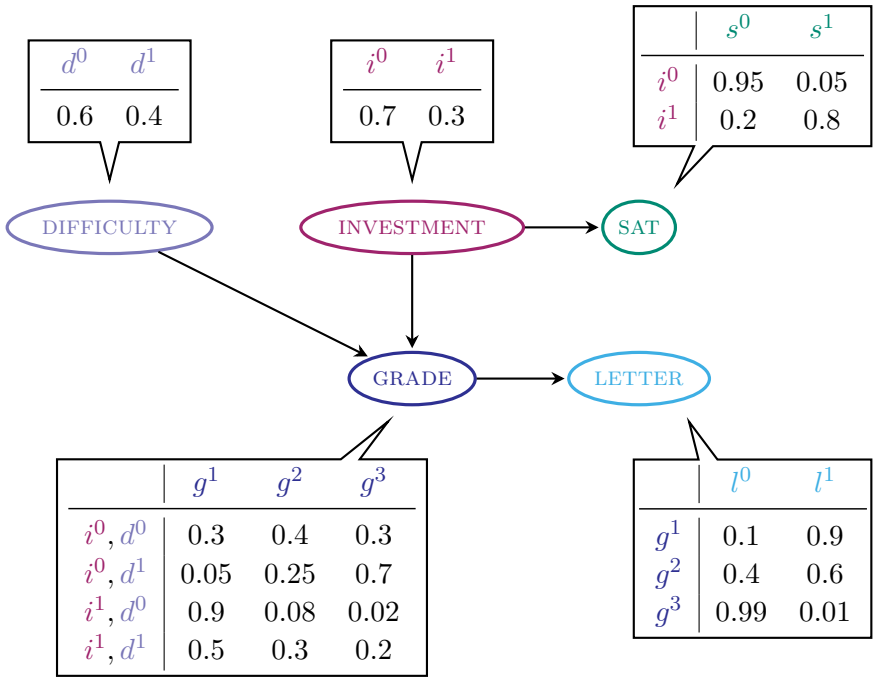


Figure 3.1: Bayesian network describing student academic performance. The joint distribution can be represented as a product of conditional probability distributions specified by the tables associated with each variable. Adapted from Koller and Friedman (2009).

performance relies on a small number of straightforward determinants that we can observe. In our simplified model, grade g depends on the exam's difficulty d and the student's investment in studying i. In turn, this grade influences the quality l of the reference letter from the teacher who taught the course. In addition, investment i influences SAT score s. Variable g takes 3 possible values. All other variables are binary.

The graphical representation of this distribution is a DAG that visually specifies how the random variables depend on each other (shown in Figure 3.1). Each variable is associated with a conditional probability table that is representative of the factor $p(x_i \mid \mathbf{pa}(x_i))$. We can read off these conditional distributions from the graphical structure to obtain a factorization for the joint distribution (following from Equation 3.1). In this case, the joint probability distribution over the 5 variables naturally

factorizes as

$$p(l, g, i, d, s) = p(l \mid g)\, p(g \mid i, d)\, p(s \mid i)\, p(i)\, p(d).$$

If we were provided the factorization first, we could easily reverse-engineer the DAG using the conditional independence assumptions that the factorization encodes. Thus, the graph and its corresponding factorization both clearly indicate the parent-child relationships in this distribution.

Another way to interpret a directed graph is as a story for how the data were generated — that is, a narrative to explain the *data generating process*. In the above example, to determine the quality of the reference letter, we can first sample an investment level and an exam difficulty; then, a student's grade is sampled given these parameters; finally, the recommendation letter is generated based on that grade.

The Factorization of Bayesian Networks

We will now define Bayesian networks and their factorization more formally. Suppose we have a Bayesian network with an associated graph $\mathcal{G} = (\mathbf{V}, \mathbf{E})$. Each node $V_i \in \mathbf{V}$ corresponds to a random variable X_i. Each node is associated with one conditional probability distribution $p(x_i \mid \mathbf{pa}(x_i))$, specifying the probability of x_i conditioned on the values taken by its parent set, $\mathbf{pa}(x_i)$. Thus, a Bayesian network defines a probability distribution p. Conversely, we say that a probability distribution *factorizes* over a DAG $\mathcal{G}$ if it can be decomposed into a product of factors, as specified by $\mathcal{G}$. We can formalize these observations, beginning with the *Markov condition*.

Definition 3.1 (Markov condition for Bayesian networks)**.** A node is conditionally independent of its non-descendants given its parents.

As a consequence of the Markov condition, we obtain the form for factorizing Bayesian networks.

Definition 3.2 (Factorization of Bayesian networks)**.** Let $p(\mathbf{x})$ be a joint probability distribution over random variables $\mathbf{X} = \{X_i\}_{i=1}^n$. Given the

Markov condition, we can factorize the joint as

$$p(\mathbf{x}) = \prod_{i=1}^{n} p(x_i \mid \mathbf{pa}(x_i)). \tag{3.1}$$

The conditional distribution describing a variable given its parent set is a *factor* or *module* that is invariant to all other factors. This fact explains the relationship between the factorization and graph given by Figure 1.1 in our introduction. Given this modularity, for all $i \neq j$, knowing $p(x_i \mid \mathbf{pa}(x_i))$ does not inform us about $p(x_j \mid \mathbf{pa}(x_j))$. Further, altering $p(x_i \mid \mathbf{pa}(x_i))$ does not alter $p(x_j \mid \mathbf{pa}(x_j))$ (Parascandolo *et al.*, 2018). These independence properties allow us to obtain compact factorizations for complex joint distributions, enabling efficient inference and learning.

It is not hard to see that a probability represented by a Bayesian network will be valid: clearly, it will be nonnegative and one can show using an induction argument (and using the fact that the conditional probability distributions are valid probabilities) that the sum over all variable assignments will be 1. Conversely, we can also show by counter-example that when $\mathcal{G}$ contains cycles, its associated probability may not sum to 1.

Independencies in Bayesian Networks

To summarize, Bayesian networks represent probability distributions that can be formed via products of simpler, local conditional probability distributions (i.e., one *factor* for each variable). By expressing a probability in this form, we introduce assumptions that certain variables are conditionally independent of each other.

This raises the question: exactly which independence assumptions are we making by using a Bayesian network model with a given structure? This question is important for two reasons. Firstly, we must clearly state the assumptions that we place on our model and be able to justify the extent to which we believe these assumptions are correct. Secondly, this information can help us design more efficient inference algorithms.

Let $I(p)$ denote the set of all independencies that hold for a joint distribution p. For example, if $p(x, y) = p(x)p(y)$, then we say that

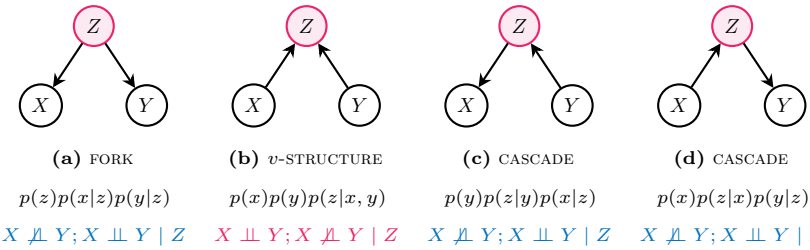


Figure 3.2: Triple DAGs where Z plays the role of common parent in a fork structure **(a)**, common child in a v-structure **(b)**, and mediator in a cascade or chain **(c, d)**. Note that **(a)**, **(c)**, and **(d)** all share the same statistical independencies, though their joint factorizations are unique.

$x \perp\!\!\!\perp y \in I(p)$. It turns out that a Bayesian network $\mathcal{G}$ very elegantly describes many independencies in $I(p)$. In actuality, Bayesian networks are imperfect representations that can fail to capture all independencies in some settings. We will return to this issue in Section 3.1.1. For now, we will build our understanding of how these independencies can be recovered from the graph by introducing three primitive independence structures.

The Triple DAG: Independence in Primitive Structures

We now introduce three primitive graphical structures that are fundamental building blocks for representation, learning, and inference on DAGs. Understanding these primitives will help us characterize how conditional independencies are encoded in a graph, which is an important step toward building models that support efficient inference procedures. For simplicity, consider a Bayesian network $\mathcal{G}$ with three nodes: X, Y, and Z. We will focus on node Z for illustration. In Figure 3.2 and Table 3.1, we enumerate every role that Z can play with respect to $\{X, Y\}$. Remark that Z can take on one of three roles: *common parent* (i.e., a special kind of *confounder*), *common child* (i.e., a special kind of *collider*; see Example 3.1 for discussion), or *mediator* (i.e., an intermediate variable on the directed path from X to Y). These correspond to the three primitive structures in Bayesian networks: the *fork*, the *v-structure*, and the *cascade* or *chain* (of which there are two,

Graphical Structure			Statistical Dependencies		
Primitive	$\mathcal{G}$	*Role of* Z	$\{X, Y\}$	$\{X, Y, Z\}$	*Intuition*
FORK	$X \leftarrow Z \rightarrow Y$	CONFOUNDER	$X \not\perp\!\!\!\perp Y$	$X \perp\!\!\!\perp Y \mid Z$	X is not a parent of Y, and has no impact on the outcome of Y. The statistical association shared by X and Y is due to shared parent Z. Thus, knowing X provides no additional information about Y once we know Z.
CASCADE	$X \rightarrow Z \rightarrow Y$	MEDIATOR	$X \not\perp\!\!\!\perp Y$	$X \perp\!\!\!\perp Y \mid Z$	The outcome of Y relies directly on Z but only indirectly on X. As Z mediates all the signal flowing from X to Y, knowing X provides no additional information about Y once we know Z.
v-STRUCTURE	$X \rightarrow Z \leftarrow Y$	COLLIDER	$X \perp\!\!\!\perp Y$	$X \not\perp\!\!\!\perp Y \mid Z$	X and Y are marginally independent, but knowing their common child Z induces statistical association between them. Thus, knowing X provides additional information about Y only once we know Z.

Table 3.1: Conditional independencies encoded by primitive DAG structures.

one with root X and one with root Y). Each of the four resulting graphs corresponds to a different factorization of the joint probability distribution. However, only the v-structure differs in the conditional independencies that it encodes. This crucial observation propels much of graph structure learning (see Section 6.2.2).

Example 3.1 (Intuition for the v-structure)**.** The v-structure is perhaps the most unintuitive primitive. Given its centrality in graph structure learning, we will elaborate on it briefly. Consider the Bayesian network in Figure 3.3, where all variables are Bernoulli (taking values 0 = false or 1 = true). Assume a simplified world where there are only two reasons why a security alarm sounds (ALARM = 1): a burglary took place or an earthquake occurred. This network encodes the belief that burglaries and earthquakes are probabilistically independent (i.e., BURGLARY $\perp\!\!\!\perp$ EARTHQUAKE). Hence, there is no edge between BURGLARY and EARTHQUAKE. Additionally, the network encodes the belief that BURGLARY and EARTHQUAKE are both causal for a security alarm sounding, and

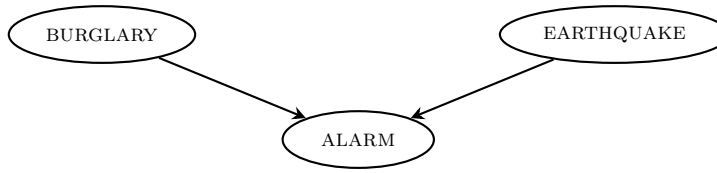


Figure 3.3: A v-structure in the classic EARTHQUAKE network (Pearl, 1988; Koller and Friedman, 2009), available in the `bnlearn` Bayesian network repository (Scutari, 2010). This network encodes the belief that burglaries and earthquakes are probabilistically independent, yet both are causal for a security alarm sounding.

are therefore probabilistically dependent on ALARM (i.e., BURGLARY $\not\perp\!\!\!\perp$ ALARM and EARTHQUAKE $\not\perp\!\!\!\perp$ ALARM). And yet, when conditioned on whether the alarm sounded, BURGLARY and EARTHQUAKE are now probabilistically *dependent* (i.e., BURGLARY $\not\perp\!\!\!\perp$ EARTHQUAKE $\mid$ ALARM). This means that knowledge of the values that ALARM and one of its parents take can provide knowledge of the value that its other parent takes. For example, if ALARM $= 1$ and BURGLARY $= 0$, then we know that EARTHQUAKE must equal 1.

Markov Equivalence The simplified setting of the triple DAG also spotlights the idea of *equivalence* among DAGs. As discussed, the fork and cascade share the same statistical dependencies, though their joint factorizations and graphical structures differ. In fact, these two primitives are *Markov equivalent*. Formally, we can define an equivalence class over structures as follows.

Definition 3.3 (Markov equivalence class (MEC), Andersson *et al.* 1997)**.** Two DAGs are Markov equivalent if and only if they share the same undirected skeleton and the same v-structures.

We can also easily see from this definition that the v-structure primitive is *not* Markov equivalent to the fork and cascade primitives: these share the same undirected skeleton, but not the same v-structures.

Independence in General Graphs

We can extend our three primitives to general networks by applying them recursively over larger graphs. To explore this in greater formality,

we will introduce the concepts of d-separation and d-connection (Pearl, 1988). To lay the groundwork, we will first introduce notions of *active paths* and *inactive paths* (Figure 3.4). These concepts require us to generalize the notions of colliders, confounders, and mediators from DAGs with three nodes to the setting of general graphs of arbitrary complexity.

Note that colliders, confounders, and mediators are all defined with respect to a specified pair of variables. In the causal setting, this might be with respect to a cause-effect pair (also known as an exposure-outcome or treatment-outcome pair). In the simplest case, colliders are *common descendants* or *common effects* of the pair of interest (e.g., node A with respect to pair X and Y in Figure 3.4). This generalizes the *common child* relationship shown in Figure 3.2b to DAGs of arbitrary size. However, a collider can also take the form of node B in Figure 3.4. In this case, the pair of interest shares confounders I and J with B, rather than being ancestral to B. Given its characteristic M-shape, this important kind of collider structure is often referred to as an *M-structure* (Ding and Miratrix, 2015; Pearl, 2015).

The formal definition of an active path relies on this general notion of a collider. We have seen basic examples of active paths already: the cascade and fork when no intermediate variable is conditioned on (Figure 3.2c,d; Table 3.1). Now we provide a general notion, which we define this with respect to the undirected skeleton of a directed path and a node set $\mathbf{S}$.

Definition 3.4 (Active path, Spirtes *et al.* 2001)**.** An undirected path is *active* relative to a node set $\mathbf{S}$ if every node on this path is active relative to $\mathbf{S}$. A node $V \in \mathbf{V}$ is active on a path relative to $\mathbf{S}$ if

1. $V \notin \mathbf{S}$ is not a collider on the corresponding directed path,
2. $V \in \mathbf{S}$ is a collider, or
3. $V \notin \mathbf{S}$ is a collider and at least one of its descendants is in $\mathbf{S}$.

We then define an *inactive* or *blocked* path as one that is not active (e.g., due to existence of a collider $\notin \mathbf{S}$ on that path). As the definitions of active and inactive are with respect to $\mathbf{S}$, we assume $\mathbf{S} = \emptyset$ unless otherwise stated. For example, when $\mathbf{S} = \emptyset$, the red paths between X and Y in Figure 3.4 are active. The black paths between X and Y are inactive

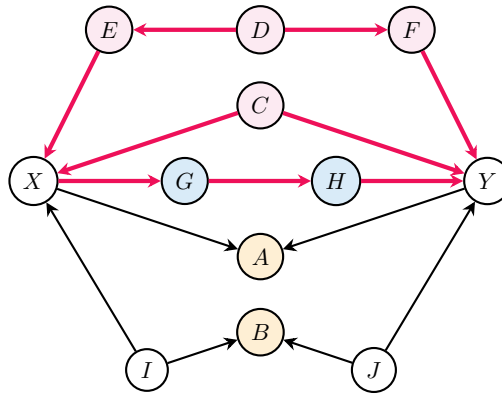


Figure 3.4: Colliders $\{A, B\}$, confounders $\{C, D, E, F\}$, and mediators $\{G, H\}$ with respect to nodes X and Y. Red paths are *active* for $\{X, Y\}$ with respect to the empty set, while black paths are *inactive* with respect to the empty set.

when $\mathbf{S} = \emptyset$, as the flow of association is blocked by colliders A and B. When $\mathbf{S} = \{A, B\}$, the black paths $X \to A \leftarrow Y$ and $X \leftrightarrow B \leftrightarrow Y$ are rendered active. To block the red path $X \leftarrow E \leftarrow D \to F \to Y$, we must condition on at least one variable in $\{E, D, F\}$. For the red path $X \to G \to H \to Y$, conditioning on at least one of the mediators G or H would render this path inactive.

With this context in mind, we finally come to define the fundamental concept of d-separation.

Definition 3.5 (d-separation)**.** Let $\mathcal{G} = (\mathbf{V}, \mathbf{E})$ be a directed graph. Let $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}$ be disjoint subsets of $\mathbf{V}$. If there is no active path between any $X \in \mathbf{X}$ and $Y \in \mathbf{Y}$ given $\mathbf{Z}$, then $\mathbf{X}$ and $\mathbf{Y}$ are d-separated given $\mathbf{Z}$.

If two variables are d-separated relative to a set of variables $\mathbf{Z}$ in a directed graph, then they are independent conditional on $\mathbf{Z}$ in all probability distributions that factorize over the graph. Roughly, variables X and Y are independent conditional on $\mathbf{Z}$ if knowledge of X provides no extra information about Y once you have knowledge of $\mathbf{Z}$.

We can then define d-connection as the negation of d-separation. Variables X and Y are d-connected by set $\mathbf{Z}$ if and only if there exists an undirected path between X and Y such that for every collider on the path, the collider or a descendent of it is in $\mathbf{Z}$, and no non-collider

on the path is in $\mathbf{Z}$.

The notion of d-separation is useful, because it lets us describe a large fraction of the dependencies that hold in our model. Let $I(\mathcal{G}) = \{(X \perp\!\!\!\perp Y \mid Z) : X, Y \text{ are } d\text{-separated given } Z\}$ be a set of variables that are d-separated in $\mathcal{G}$.

Definition 3.6 (I-map)**.** If p factorizes over $\mathcal{G}$, then $I(\mathcal{G}) \subseteq I(p)$. In this case, we say that $\mathcal{G}$ is an I-map (independence map) for p.

In other words, all the independencies encoded in $\mathcal{G}$ are sound: variables that are d-separated in $\mathcal{G}$ are truly independent in p. However, the converse is not true: a distribution may factorize over $\mathcal{G}$, yet have independencies that are not captured in $\mathcal{G}$.

In a way this is almost a trivial statement. If $p(x, y) = p(x)p(y)$, then this distribution still factorizes over the graph $y \to x$, since we can always write it as $p(x, y) = p(x \mid y)p(y)$ with a conditional probability distribution $p(x \mid y)$ in which the probability of x does not actually vary with y. However, we can construct a graph that matches the structure of p by simply removing that unnecessary edge.

Markov Blankets in Bayesian Networks

Per the Markov condition, we know that a variable X is conditionally independent of its non-descendants given its parents (Definition 3.1). Another important conditional independence fact in Bayesian networks follows from the definition of a *Markov blanket* (Pearl 1988).

Definition 3.7 (Markov blankets in Bayesian networks)**.** Let $\mathcal{G} = (\mathbf{V}, \mathbf{E})$ and $X \in \mathbf{V}$ be a graph and node of interest, respectively. When $\mathcal{G}$ is a Bayesian network, the Markov blanket for X is the union of the parents, children, and spouses of X.

Following from this definition, a node X is conditionally independent of all other nodes in $\mathcal{G}$, given its Markov blanket. As node X is d-separated from the remainder of the graph given its Markov blanket, we can use the Markov blanket as a parsimonious set of features to explain X (Aliferis *et al.*, 2010). As we will see in Chapter 5, Markov blankets are useful objects for approximate inference. Note that Markov

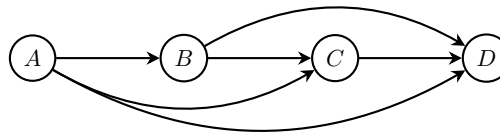


Figure 3.5: A fully connected Bayesian network over four variables. There are no independencies in this model, and it is an I-map for any distribution.

blankets are defined differently in undirected graphs (see Definition 3.8 and Figure 3.8).

Powerful Yet Imperfect Representations

In summary, we have shown that a Bayesian network $\mathcal{G}$ representing joint probability distribution p specifies

1. a factorization of p that is a product of conditional distributions, each describing a variable given its parent set; and

2. a set of conditional independencies that must be satisfied by any p factorizing according to $\mathcal{G}$.

This raises our last and perhaps most important question: can DAGs express *all* the independencies of any distribution p? More formally, given a distribution p, can we construct a graph $\mathcal{G}$ such that $I(\mathcal{G}) = I(p)$?

First, note that it is easy to construct a DAG $\mathcal{G}$ such that $I(\mathcal{G}) \subseteq I(p)$. A fully connected DAG $\mathcal{G}$ is an I-map for any distribution since $I(\mathcal{G}) = \emptyset$ (Figure 3.5). A more interesting question is whether we can find a *minimal* I-map $\mathcal{G}$ for p. That is, we might wish to find an I-map $\mathcal{G}$ such that the removal of even a single edge from $\mathcal{G}$ will result in it no longer being an I-map. To do this, we could start with a fully connected $\mathcal{G}$ and iteratively remove edges until $\mathcal{G}$ is no longer an I-map.

However, what we are truly interested in determining is whether *any* probability distribution p always admits a *perfect* map $\mathcal{G}$ for which $I(p) = I(\mathcal{G})$. Unfortunately, the answer is no. For example, consider the following distribution p over three variables X, Y, Z. We can sample Bernoulli random variables $X, Y \sim \text{Ber}(0.5)$ and set

$$Z = (X \text{ xor } Y) \text{ xor } \epsilon,$$

where ϵ is a noise term representing a biased coin flip (also Bernoulli) and "xor" denotes the *exclusive or* logical operator (which is true if and only if its two arguments differ). This data generating process is referred to as the *noisy-xor example*. One can check using some algebra $\{X \perp\!\!\!\perp Y, Z \perp\!\!\!\perp Y, X \perp\!\!\!\perp Z\} \in I(p)$ but $Z \perp\!\!\!\perp \{Y, X\} \notin I(p)$. Thus, $X \rightarrow Z \leftarrow Y$ is an I-map for p, but none of the 3-node graph structures that we discussed perfectly describes $I(p)$, and hence this distribution doesn't have a perfect map. The noisy-xor example is a classic case of a *faithfulness violation*, where faithfulness dictates that conditional independence in p implies d-separation in $\mathcal{G}$ (Marx *et al.*, 2021). We will revisit faithfulness briefly in Section 6.2, as it is a common assumption in graph structure learning.

A related question is whether perfect maps are unique when they exist. Again, this is not the case. As a counterexample, $X \rightarrow Y$ and $X \leftarrow Y$ encode the same independencies, yet form different graphs. This invokes again the concept of Markov equivalence (Definition 3.3). Indeed, this phenomenon presents a fundamental challenge for learning graphical structures using tests of conditional independence alone, and we will see in Section 6.2.4 that such methods cannot provide graphical information beyond the MEC (Spirtes *et al.*, 2001).

Further Reading

- D. Heckerman. (1998). "A Tutorial on Learning with Bayesian Networks". *Learning in Graphical Models*: 301–354.
- J. Pearl. (1995a). "Causal Diagrams for Empirical Research". *Biometrika*. 82(4): 669–688.
- J. Pearl. (1995b). "From Bayesian Networks to Causal Networks". In: *Mathematical Models for Handling Partial Knowledge in Artificial Intelligence*. Springer. 157–182.
- G. F. Cooper. (1990). "The Computational Complexity of Probabilistic Inference Using Bayesian Belief Networks". *Artificial Intelligence*. 42(2-3): 393–405.

3.2 Representing Distributions with Undirected Graphs

While Bayesian networks can compactly represent many interesting probability distributions, we have seen that some distributions cannot be perfectly represented by models in this class. To prevent false independencies among the variables of our model, it may seem that such cases require a less compact representation. Under a naive approach, this could potentially result in a graph with superfluous edges — and, consequently, unnecessary parameters in the model. This could make it more difficult to learn the parameters of our model and to make predictions.

However, Bayesian networks are not the only technique for compactly representing and visualizing probability distributions. In this section, we review three representational forms for undirected graphical models: *Markov random fields*, *conditional random fields*, and *factor graphs*.

3.2.1 Markov Random Fields

We begin our discussion of undirected graphical models with *Markov random fields* (MRFs), also known as *Markov networks*. This class of models can compactly represent independence assumptions that directed models cannot represent. MRFs have featured prominently in image analysis and computer vision (Geman and Graffigne, 1986; Li, 2009; Kato, Zerubia, *et al.*, 2012), spatiotemporal statistics (Clifford, 1990), models of term dependency in text analysis (Metzler and Croft, 2005), and more. In this section, we will explore the advantages and drawbacks of MRFs for various problem settings.

Formally, an MRF is a probability distribution p over variables $\mathbf{X} = \{X_i\}_{i=1}^n$ defined by an undirected graph $\mathcal{G}$, in which nodes correspond to variables X_i. The probability distribution p takes the form

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{c \in \mathbf{C}} \phi_c(\mathbf{x}_c), \tag{3.2}$$

where $\mathbf{C}$ denotes the set of cliques in $\mathcal{G}$ (Definition 2.35; Figure 2.6), *partition function* Z is a normalizing constant, and each *factor* ϕ_c is a nonnegative function $\phi_c : \text{Val}(\mathbf{X}_c) \mapsto \mathbb{R}$ over the variables in a clique. As $p(\mathbf{x})$ is a product of nonnegative functions, $p(\mathbf{x})$ is also guaranteed

to be nonnegative. The clique constitutes the *scope* (i.e., the domain) of factor ϕ_c (e.g., $Scope[\phi_c] = \mathbf{X}_c$).

Factors ϕ_c are also referred to as *potential functions* or *clique potentials*, and we often prefer to define these over *maximal* cliques (Definition 2.36). However, it is equally valid for $p(\mathbf{x})$ to contain factors whose scope is *any* clique in $\mathcal{G}$ (e.g., pairwise potentials). Counter to what we have seen for the factorization of Bayesian networks (Definition 3.2), MRF potentials are not necessarily restricted to marginal or conditional distributions. This lack of restriction can result in a product of potentials that does not sum to 1. This explains the importance of partition function

$$Z \coloneqq \sum_{x_1,\ldots,x_n} \prod_{c \in \mathbf{C}} \phi_c(\mathbf{x}_c),$$

which ensures that $p(\mathbf{x})$ is a properly normalized probability distribution summing to 1. Note that we can generalize the formula for Z to accommodate continuous random variables by replacing summation with integration, where appropriate.

In summary, a distribution p can be expressed as a normalized product of factors, each with a scope over any clique in the corresponding undirected graph $\mathcal{G}$. Thus, the scope of a potential can be a single node, an edge, a triangle, etc. Furthermore, we do not need to specify a factor for each clique. The cliques over which we define our factors is a modeling choice, as we will illustrate in the following example.

An Illustrative Example

As a motivating example, suppose that we are modeling voting preferences among individuals. Let each person receive a label $\in \{A, B, C, D\}$. Let's say that $\{A, B\}$, $\{B, C\}$, $\{C, D)$, and $\{D, A\}$ are friends. From domain knowledge, we assume that friends tend to have similar voting preferences. These influences can be naturally represented by an undirected graph, like that in Figure 3.6.

One way to define a probability over the joint voting decision of $\{A, B, C, D\}$ is to assign scores to each assignment to these variables, and then define a probability as a normalized score. A score can be any

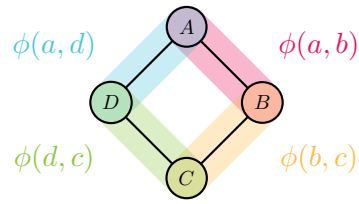


Figure 3.6: MRF representing a joint probability distribution of voting preferences over four individuals. Highlighting indicates the potential functions (ϕ) associated with each clique.

function, but here we will define it to be

$$\tilde{p}(a,b,c,d) = \phi(a,b)\phi(b,c)\phi(c,d)\phi(d,a),$$

where $\phi(x,y)$ is a factor that assigns more weight to consistent votes among friends X, Y. For example,

$$\phi(x,y) = \begin{cases} 10 & \text{if } X = Y = 1 \\ 5 & \text{if } X = Y = 0 \\ 1 & \text{otherwise.} \end{cases}$$

The final probability is then defined as

$$p(a,b,c,d) = \frac{1}{Z}\tilde{p}(a,b,c,d),$$

where $Z = \sum_{a,b,c,d} \tilde{p}(a,b,c,d)$ is a normalizing constant that ensures that the distribution sums to 1.

When normalized, we can view $\phi(a,b)$ as an interaction that pushes B's vote closer to that of A. The term $\phi(b,c)$ pushes B's vote closer to C, and the most likely vote will require reconciling these conflicting influences.

In this example, a sensible modeling choice was to define a factor over each edge (i.e., a clique of two nodes). Technically, we could have chosen to additionally specify *unary factors* (i.e., cliques over single nodes). However, this would not have been a useful choice for this setting, as our assumptions concern only relations between friends, not individual voting tendencies.

Unlike in the directed case, the undirected model we describe here is not saying anything about how one variable is generated from another

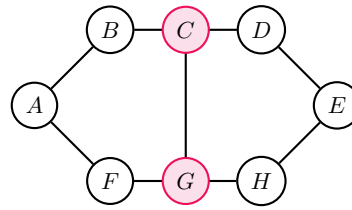


Figure 3.7: As $\{C, G\}$ forms a cutset of nodes for this graph, $\{A, B, F\}$ and $\{D, E, H\}$ are conditionally independent given $\{C, G\}$.

set of variables (as a conditional probability distribution would do). We have simply indicated a level of coupling between dependent variables in the graph. In a sense, this requires less prior knowledge, as we no longer have to specify a full generative story of how the vote of B is constructed from the vote of A (which we would need to do if we had a factor of the form $p(b \mid a)$). Instead, we simply identify dependent variables and define the strength of their interactions. This in turn defines an *energy landscape* over the space of possible assignments, and we convert this energy to a probability via the normalizing constant.

Independencies in Markov Random Fields

Recall that in the case of Bayesian networks, we defined a set of independencies $I(\mathcal{G})$ that were described by a directed graph $\mathcal{G}$. We then showed how these describe true independencies that must hold in a distribution p that factorizes over the directed graph (i.e., $I(\mathcal{G}) \subseteq I(p)$).

What independencies can be described by an MRF? The answer here is very simple and intuitive. Take MRF $\mathcal{G} = (\mathbf{V}, \mathbf{E})$. Variables $X, Y \in \mathbf{V}$ are dependent if they are connected by a path of unobserved variables. Now assume that every neighbor of X, denoted as set $\mathbf{ne}(X)$, is observed. Then, X is conditionally independent of $\mathbf{V} \setminus \mathbf{ne}(X)$ given $\mathbf{ne}(X)$. This is because the neighbors of X *directly* influence X, but all remaining variables *indirectly* influence X via its neighbors. Thus, we can *block* their influence by conditioning on $\mathbf{ne}(X)$.

We can also consider independence in MRFs in terms of *cutsets*, where a cutset is a set of nodes (or a set of edges) whose removal from $\mathcal{G}$ disconnects $\mathcal{G}$ into at least two connected components. Given the nodes

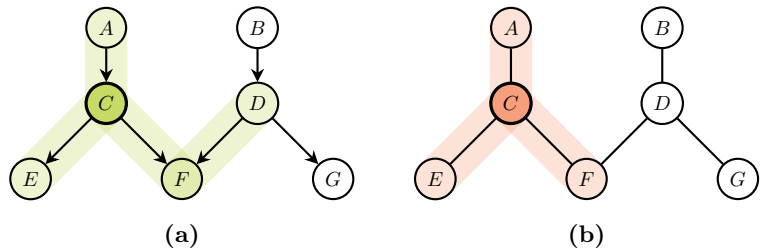


Figure 3.8: Highlighted nodes belong to the Markov blanket for variable C in a directed graph **(a)** and an undirected graph **(b)**.

in a cutset, the nodes in one connected component will be conditionally independent of those in another component. In Figure 3.7, $\{C, G\}$ is a cutset of nodes that partitions the graph into two disjoint subsets.

Finally, we can revisit *Markov blankets* (Definition 3.7). As highlighted in Figure 3.8, the Markov blanket for a node in a directed graph can differ from its Markov blanket in the corresponding undirected skeleton.

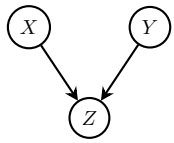
Definition 3.8 (Markov blankets in undirected graphs)**.** Let $\mathcal{G} = (\mathbf{V}, \mathbf{E})$ and $X \in \mathbf{V}$ be a graph and node of interest, respectively. When $\mathcal{G}$ is an undirected graph, the Markov blanket for a node X is the set of all neighbors for X, i.e., $\mathbf{ne}(X)$.

Thus, X is independent from the remaining nodes in $\mathcal{G}$ when its Markov blanket is observed. The utility of this fact will become more clear when we explore approximate inference in Chapter 4.

Comparison to Bayesian Networks

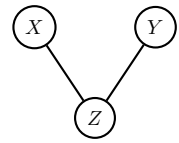
As with Bayesian networks, MRFs are powerful but imperfect modes of representation. In the directed case, we found that $I(\mathcal{G}) \subseteq I(p)$ but yet there were distributions p whose independencies could not be exactly described by $\mathcal{G}$. In the undirected case, the same holds.

So, what independencies *cannot* be described by an MRF? Consider a probability distribution described by a DAG $\mathcal{G}$ that takes the form of a v-structure (Figure 3.9). Neither the undirected skeleton of $\mathcal{G}$ nor



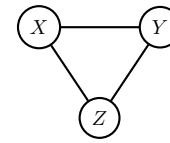
(a) DAG

$X \perp Y, X \not\perp Y \mid Z$



(b) MRF

$X \not\perp Y, X \perp Y \mid Z$



(c) MORAL GRAPH

$X \not\perp Y, X \not\perp Y \mid Z$

Figure 3.9: The v-structure (Figure 3.2) is an example of a probability distribution that has a perfect directed graphical representation but no undirected representation.

its corresponding moral graph can describe the same independence assumptions encoded by the original DAG.

In our voting example, we had a distribution over $\{A, B, C, D\}$ that satisfied $A \perp\!\!\!\perp C \mid \{B, D\}$ and $B \perp\!\!\!\perp D \mid \{A, C\}$. Intuitively, this is reasonable: we assumed that only friends can directly influence each others' vote, and so a person's voting preference is independent of their non-friends' preferences given their friends' preferences. The MRF turns out to be a perfect map for this distribution (Figure 3.6). However, we can demonstrate by counterexample that these independencies cannot be perfectly represented by a Bayesian network (Figure 3.10). Given the acyclicity constraint in Bayesian networks, there must be at least one sink node (in this case, colliders with no descendants). Thus, $A \perp\!\!\!\perp C \mid \{B, D\}$ and $B \perp\!\!\!\perp D \mid \{A, C\}$ cannot both be satisfied as at least one conditioning set will open up an active path (Definition 3.4).

More generally, MRFs have several advantages over directed models in certain settings. These include:

1. MRFs can provide compact representations for a wider range of problems in which there is no natural directionality associated with variable dependencies.

2. MRFs can succinctly express certain dependencies that DAGs cannot easily describe (although the converse is also true).

However, MRFs also possess several important drawbacks relative to Bayesian networks. These include:

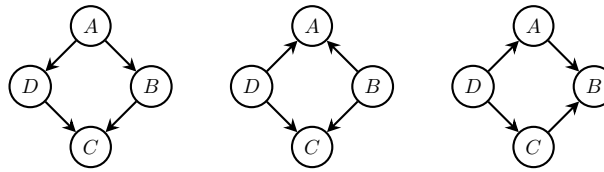


Figure 3.10: Examples of directed models for our four-variable voting example. None can accurately express our prior knowledge about the dependency structure among the variables, such that all variables are marginally dependent but $A \perp\!\!\!\perp C \mid \{B, D\}$ and $B \perp\!\!\!\perp D \mid \{A, C\}$.

1. Computing the normalizing constant Z requires summing over a potentially exponential number of assignments. In the general case, this will be NP-hard. Thus, many MRFs will be intractable and will require approximation techniques.

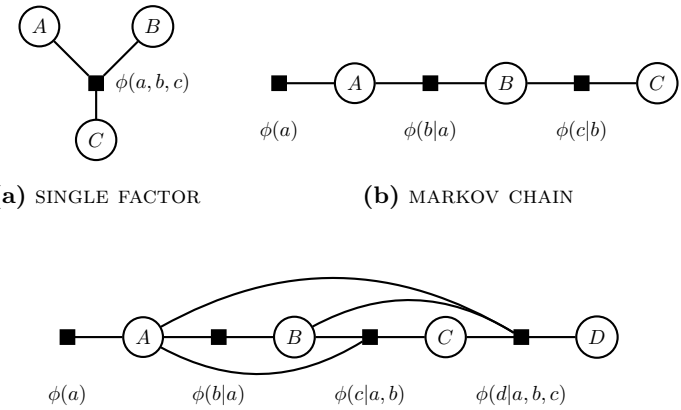
2. Undirected models can be difficult to interpret.

3. It is much easier to generate data from a Bayesian network, which is important in some applications.

4. MRFs do not admit a causal interpretation.

It is not hard to see that Bayesian networks are a special case of MRFs with (1) a very specific type of clique factor, which corresponds to a conditional probability distribution and implies a directed acyclic structure in the graph; and (2) a normalizing constant of 1. Recall our prior discussion on *moralization* (Figure 2.7): by "marrying" non-adjacent nodes that share children (i.e., removing *immoralities*), a Bayesian network can always be converted into an undirected network with normalizing constant 1 (e.g., Figure 3.9). The converse is also possible. However, this may yield a very large (e.g., fully connected) directed graph, making inference computationally intractable

Thus, in some senses, MRFs have more power than Bayesian networks. Nevertheless, they are more difficult to deal with computationally and they limit certain forms of interpretation. A general rule of thumb is to use Bayesian networks whenever possible, opting for MRFs only



(a) SINGLE FACTOR **(b)** MARKOV CHAIN

(c) FACTORIZATION BY CHAIN RULE

Figure 3.11: Example factor graphs with factors ϕ. **(a)** A probability distribution represented by a single factor (i.e., the trivial case). **(b)** A Markov chain. **(c)** The factor graph resulting from applying the chain rule of probability to a distribution (Definition 2.28). Adapted from Kschischang *et al.* (2001).

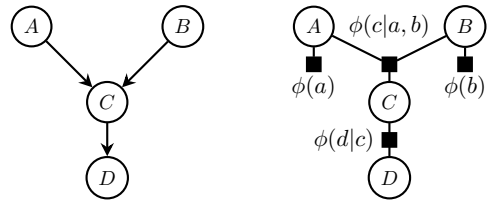
if there is no natural way to model the problem with a directed graph (e.g., in our voting example).

Further Reading

- Z. Kato, J. Zerubia, *et al.* (2012). "Markov Random Fields in Image Segmentation". *Foundations and Trends® in Signal Processing*. 5(1–2): 1–155.

3.2.2 Factor Graphs

It is often useful to view MRFs in a way where factors and variables are explicit and separate in the representation. A *factor graph* is one such way to do this. A factor graph is a *bipartite graph* (Definition 2.44). Nodes representing variables in the distribution constitute one partition, while nodes representing the factors defined on these variables constitute the second partition. In other words, all edges go between factors and the variables that those factors depend on (Figure 3.11).



(a) DAG $\mathcal{G}$ **(a)** FACTOR GRAPH $\mathcal{F}$

Figure 3.12: Factor graph for a simple DAG representing a probability distribution that factorizes as $p(a, b, c, d) = p(a)p(b)p(c|a, b)p(d|c)$.

This graphical representation allows us to more readily see the factor dependencies between variables. We can obtain factor graphs for undirected graphs or for directed graphs (Figure 3.12). As MRFs can admit multiple valid factorizations, multiple factor graphs can correspond to the same distribution. We will see in Chapter 4 that factor graphs are a useful tool in inference, particularly in the family of message-passing algorithms.

Further Reading

- F. R. Kschischang *et al.* (2001). "Factor graphs and the sum-product algorithm". *IEEE Transactions on information theory.* 47(2): 498–519.

3.2.3 Conditional Random Fields

An important special case of MRFs arises when MRFs are applied to model a conditional probability distribution $p(\mathbf{y} \mid \mathbf{x})$, where $\mathbf{x} \in \mathcal{X}$ and $\mathbf{y} \in \mathcal{Y}$ are vector-valued variables. *Conditional random fields* (CRFs) were introduced for segmenting and labeling sequence data, and are essentially MRFs with clique potentials that are conditioned on a set of features (Lafferty *et al.*, 2001). CRFs were designed to address several drawbacks of hidden Markov models, stochastic grammars, and other generative models commonly used in domains that model sequence data

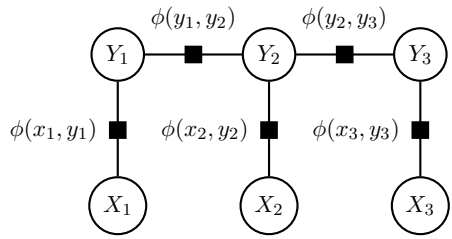


Figure 3.13: A factor graph representation of a linear-chain CRF.

(e.g., computational linguistics and computational biology).

Conditional distributions $p(\mathbf{y} \mid \mathbf{x})$ are common in supervised learning settings in which we are given $\mathbf{x}$ and want to predict $\mathbf{y}$. This setting is also known as *structured prediction*, and CRFs can be viewed as a structured output extension of logistic regression (Murphy, 2012). CRFs are more statistically efficient than MRFs when all we care to model is the distribution of labels given the data (just as discriminative classifiers can be efficient relative to generative classifiers).

The conditional distribution specified by a CRF can be expressed as a normalized product of potentials

$$p(\mathbf{y} \mid \mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{c \in \mathbf{C}} \phi_c(\mathbf{x}_c, \mathbf{y}_c)$$

with partition function

$$Z(\mathbf{x}) = \sum_{\mathbf{y} \in \mathcal{Y}} \prod_{c \in \mathbf{C}} \phi_c(\mathbf{x}_c, \mathbf{y}_c).$$

Note that in this case, the normalizing constant now depends on $\mathbf{x}$ (therefore, we say that it is a function). This is not surprising: $p(\mathbf{y} \mid \mathbf{x})$ is a probability over $\mathbf{y}$ that is parameterized by $\mathbf{x}$. That is, it encodes a different probability function for each $\mathbf{x}$. In that sense, a CRF results in an instantiation of a new MRF for each input $\mathbf{x}$.

CRFs are often represented as factor graphs (e.g., Figure 3.13). Every conditional distribution $p(\mathbf{y} \mid \mathbf{x})$ is a CRF for some factor graph, though that graph may be trivial (Sutton, 2012). Formally, we can say that a CRF meets the following condition.

Definition 3.9 (Conditional random field (CRF), Sutton 2012)**.** Let $\mathcal{G}$ be

a factor graph over random vector $\mathbf{Y}$. If the distribution $p(\mathbf{y} \mid \mathbf{x})$ for any fixed $\mathbf{x}$ factorizes according to $\mathcal{G}$, then $p(\mathbf{y} \mid \mathbf{x})$ is a CRF.

An Illustrative Example

As a motivating example, consider the problem of recognizing a word from a sequence of black-and-white character images $\mathbf{x}_i \in [0,1]^{d \times d}$ given to us in the form of pixel matrices. The output of our predictor is a sequence of alphabet letters $\mathbf{y}_i \in \{\mathcal{A}, \mathcal{B}, \ldots, \mathcal{Z}\}$.

We could in principle train a classifier to separately predict each $\mathbf{y}_i$ from $\mathbf{x}_i$. However, since the letters together form a word, the predictions across different i ought to inform each other. Consider the example CRF shown in Figure 3.14, which illustrates a prediction of the word *"QUEST"*. In this example, say it was ambiguous whether the second letter was a $\mathcal{U}$ or a $\mathcal{V}$. Since we can tell with high confidence that its neighbors are $\mathcal{Q}$ and $\mathcal{E}$, we can infer that $\mathcal{U}$ is the most likely true label. CRFs enable us to perform this prediction jointly.

More formally, suppose $p(\mathbf{y} \mid \mathbf{x})$ is a chain CRF with two types of factors. First, we have image factors $\phi(\mathbf{x}_i, \mathbf{y}_i)$ for $i = 1, \ldots, n$. These assign higher values to $\mathbf{y}_i$ that are consistent with an input $\mathbf{x}_i$. We can also think of the $\phi(\mathbf{x}_i, \mathbf{y}_i)$ as probabilities $p(\mathbf{y}_i \mid \mathbf{x}_i)$ given by, say, standard (unstructured) softmax regression. Second, we have pairwise factors $\phi(\mathbf{y}_i, \mathbf{y}_{i+1})$ for $i = 1, \ldots, n-1$. These pairwise factors can be seen as empirical frequencies of letter co-occurrences obtained from a large corpus of English text (e.g., Wikipedia).

Given a model of this form, we can jointly infer the structured label $\mathbf{y}$ using MAP inference, which we will discuss further in Chapter 5:

$$\hat{\mathbf{y}} = \operatorname*{argmax}_{\mathbf{y}} \phi(\mathbf{y}_1, \mathbf{y}_1) \prod_{i=2}^{n} \phi(\mathbf{y}_{i-1}, \mathbf{y}_i) \phi(\mathbf{x}_i, \mathbf{y}_i).$$

CRF Features

In most practical applications of CRFs, we further assume that the factors $\phi_c(\mathbf{x}_c, \mathbf{y}_c)$ are of the form

$$\phi_c(\mathbf{x}_c, \mathbf{y}_c) = \exp(\mathbf{w}_c^T \mathbf{f}_c(\mathbf{x}_c, \mathbf{y}_c)),$$

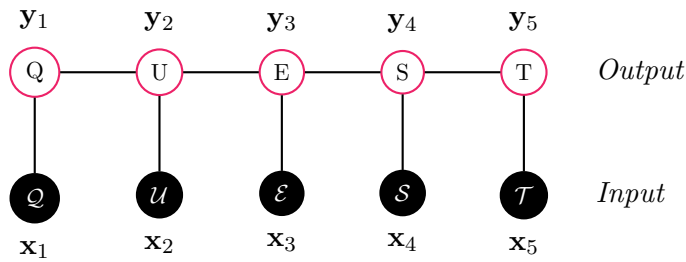


Figure 3.14: A chain-structured CRF for optical character recognition.

where $\mathbf{w}$ denotes a weight vector and $\mathbf{f}_c(\mathbf{x}_c, \mathbf{y}_c)$ is a set of *features* describing the compatibility between $\mathbf{x}_c$ and $\mathbf{y}_c$.

In our optical character recognition example, we can introduce features $\mathbf{f}(\mathbf{x}_i, \mathbf{y}_i)$ that encode the compatibility of the letter $\mathbf{y}_i$ with the pixels $\mathbf{x}_i$. For example, $\mathbf{f}(\mathbf{x}_i, \mathbf{y}_i)$ may be the probability of letter $\mathbf{y}_i$ produced by logistic regression (or a deep neural network) evaluated on pixels $\mathbf{x}_i$. In addition, we can introduce features $\mathbf{f}(\mathbf{y}_i, \mathbf{y}_{i+1})$ between adjacent letters. These may be indicators of the form

$$\mathbf{f}(\mathbf{y}_i, \mathbf{y}_{i+1}) = \mathbb{I}(\mathbf{y}_i = \ell_1, \mathbf{y}_{i+1} = \ell_2),$$

where ℓ_1, ℓ_2 are two letters of the alphabet. The CRF would then learn weights $\mathbf{w}$ that assign more weight to more common letter sequences (ℓ_1, ℓ_2), while at the same time making sure that the predicted $\mathbf{y}_i$ are consistent with the input $\mathbf{x}_i$. This process would allow us to determine $\mathbf{y}_i$ in cases where $\mathbf{x}_i$ is ambiguous, like in our above example.

The most important thing to know about CRF features is that they can be arbitrarily complex. In fact, we can define an optical character recognition model with factors $\phi(\mathbf{x}, \mathbf{y}_i) = \exp(\mathbf{w}_i^T \mathbf{f}(\mathbf{x}, \mathbf{y}_i))$ that depend on the entire input $\mathbf{x}$. This does not affect computational performance at all: at inference time, $\mathbf{x}$ is always observed and our decoding problem involves maximizing

$$\phi(\mathbf{x}, \mathbf{y}_1) \prod_{i=2}^{n} \phi(\mathbf{y}_{i-1}, \mathbf{y}_i) \phi(\mathbf{x}, \mathbf{y}_i) = \phi'(\mathbf{y}_1) \prod_{i=2}^{n} \phi(\mathbf{y}_{i-1}, \mathbf{y}_i) \phi'(\mathbf{y}_i),$$

where $\phi'(\mathbf{y}_i) = \phi(\mathbf{x}, \mathbf{y}_i)$. Using global features only changes the values of the factors but not their scope, which possesses the same type of

chain structure. We will see in the next chapter that this structure is all that is needed to ensure a tractable solution to this optimization problem.

This observation may be interpreted in a slightly more general form. If we were to model $p(\mathbf{x}, \mathbf{y})$ using an MRF (viewed as a single model over $\mathbf{x}, \mathbf{y}$ with normalizing constant $Z = \sum_{\mathbf{x},\mathbf{y}} \tilde{p}(\mathbf{x}, \mathbf{y})$), then we would need to fit two distributions to the data: $p(\mathbf{y} \mid \mathbf{x})$ and $p(\mathbf{x})$. However, if all we are interested in is predicting $\mathbf{y}$ given $\mathbf{x}$, then modeling $p(\mathbf{x})$ is unnecessary. In fact, it may be statistically disadvantageous to do so. For example, we might not have enough data to fit both $p(\mathbf{y} \mid \mathbf{x})$ and $p(\mathbf{x})$; since the models have shared parameters, fitting one may not result in the best parameters for the other. Further, it might not be a good idea computationally: we would need to make simplifying assumptions so that $p(\mathbf{x})$ can be handled tractably. CRFs forgo this assumption, and thus often perform better on prediction tasks.

Further Reading

- C. Sutton. (2012). "An Introduction to Conditional Random Fields". *Foundations and Trends in Machine Learning*. 4(4): 267–373. ISSN: 1935-8237, 1935-8245. DOI: 10.1561/2200000013.