Markov Random Fields and Conditional Random Fields

Introduction

Markov chains provided us with a way to model 1D objects such as contours probabilistically, in a way that led to nice, tractable computations. We now consider 2D Markov models. These are more powerful, but not as easy to compute with. In addition we will consider two additional issues. First, we will consider adding *observations* to our models. These observations are conditioned on the value of the set of random variables we are modeling. (If we had considered observations with Markov chains, we would have arrived at Hidden Markov Models (HMMS), which are widely used in vision and other fields). Second, we will consider the case in which these observations potentially break our Markov assumptions, because the same observation may depend on multiple random variable, making the variables themselves dependent on each other. In such a case, we may retrieve the desired Markov properties when we condition on the observations.

Markov Random Fields

In MRFs, we also consider a set of random variables which have some conditional independence properties. We introduce some terminology which is slightly different from the 1D case. We say that the MRF contains a set of *sites*, which we may index with two values, i and j, when we wish to emphasize the 2D structure of the sites. So we might talk about site $S_{i,j}$ and it's horizontal neighbor, $S_{i+1,j}$. It may also be convenient to use a single index, referring to sites as S_i. In this case, the order of the sites is arbitrary.

We also suppose that we have a set of labels, L_d. Labels might take on continuous values, but we'll assume that we have a discrete set of labels. Every site will have a label. So the sites may be thought of as random variables that can take a discrete set of values. A *labeling* assigns a label to every site. We denote this as $f = \{f_1, ..., f_m\}$, so that f_i is the label of site S_i.

Next, we assume that our sites have a neighborhood structure. N_i denotes all the sites that are neighbors to S_i. That is, $S_j \in N_i$ means S_j and S_i are neighbors. We can define any neighborhood structure that we want, with the constraint that being neighbors is symmetric, that is, that $S_j \in N_i \Leftrightarrow S_i \in N_j$. Also, a site is not its own neighbor. We denote the set of all neighborhoods as N.

This neighborhood structure now allows us to define the Markov structure of our distribution. We say that F is an MRF on S with respect to N if and only if:

$$P(f) > 0, \forall f$$

and

$$P(f_i | f_{S-\{i\}}) = P(f_i | f_{N_i})$$

The first condition is needed for some technical reasons. It means that in modeling we can't make any labeling have 0 probability, but since the probability can be very small this isn't much of a restriction. The second condition provides the Markov property.

As an example of this, let's consider the problem of segmenting an image into foreground and background. We can assign a site to every pixel in the image. Our label set is binary, indicating foreground or background. Suppose we wish to encode the constraint that foreground regions

tend to be compact, by stating that if a pixel is foreground, its neighboring pixels are also likely to be foreground. We can define a simple neighborhood structure based on 4-connected neighbors. That is, $N_{i,j} = \{S_{i-1,j}, S_{i+1,j}, S_{i,j-1}, S_{i,j+1}\}$ (notice how we switch from using one to two subscripts). Then with the conditional probabilities available to us, we can encode constraints such as that if all of a pixels neighbors are foreground, it is probably foreground, if all its neighbors are background, it is probably background, and in other cases, it is fairly likely to be either. (You may also notice that this MRF has no connection yet to the intensities in the image. This will be handled below.)

MRFs and Gibbs Distributions

Given any set of random variables, there are a number of natural problems to answer. First, given an MRF it is straightforward to determine the probability of any particular labeling. However, figuring out what set of conditional probabilities to use in an MRF is not so simple. For one thing, an arbitrary set of conditional probabilities for different sites and neighborhoods may not be mutually consistent, and it is not obvious how to determine this. Finally, a key problem will be to find the most likely labeling of an MRF.

These problems are made easier by the use of Gibbs distributions, which turn out to be equivalent to MRFs, but in some ways are much easier to work with. In a Gibbs distribution, the cliques capture dependencies between neighborhoods. A set of sites, $\{i_1, i_2, ..., i_n\}$ form a clique if for all k, j, $i_k \in N_j$. Given a probability distribution defined for a set of sites and labels, we say that it is a Gibbs distribution if the distribution is of the following form:

$$P(f) = \frac{1}{Z} e^{-\frac{U(f)}{T}} \tag{1}$$

in which the *energy function*, $U(f)$, is of the form:

$$U(f) = \Sigma_{c \in C} V_c(f)$$

where C is the set of all cliques, and $V_c(f)$ is the *clique potential*, defined for every clique. That is, $P(f)$ is an exponential function over the sum of potentials that can be defined independently for each clique. This is analogous to the independence structure given by neighborhoods in MRFs. In the above terms, T is a scalar called the temperature. In the above equation, Z is a normalizing value needed to make the probabilities sum to 1:

$$Z = \Sigma_{f \in F} e^{-\frac{U(f)}{T}}$$

T is a scalar that determines how sharply peaked the distribution is; note that as T becomes very small, the distribution is dominated by its most likely element.

The main reason Gibbs distributions are important to us is that they turn out to be equivalent to MRFs. That means that for any MRF, we can write it's probability distribution in the form of Equation 1. That means that in learning or designing an MRF, we can focus on finding the clique potentials. Note that to fully specify this distribution is still hard, since we must determine the value of Z. A straightforward way of computing this involves computing a sum with an exponential number of terms. There is a lot of work on approximating this value. However, in many cases we

don't need it, because to find the MAP distribution for an MRF we just have the problem of finding the labeling that minimizes the energy function $U(f)$, since Z is a constant factor that applies to all labeling. Finding the labeling that minimizes $U(f)$ is still an NP-hard combinatorial optimization problem in most cases, but there are many algorithms that attack this problem.

MRFs with Observations

So far, we have only considered distributions over labels. This amounts to a means for specifying a prior distribution over labeled images. But we also want to connect this with the information in a specific image. To do this, we'll use examples in which every site is a pixel, so that there is one piece of image information at each site. We'll call the image information X, with the information at each pixel given by X_i or $X_{i,j}$. We are then interested in solving problems like:

$$argmax_f P(f|X)$$

To make this concrete, let's consider an example of image denoising. We consider an MRF in which each pixel is a site, and two pixels are neighbors if they are 4-connected. Suppose every pixel has an intrinsic intensity from 0 to 255, given by its label. X_i is this intensity, with noise added, so that

$$X_i = f_i + e_i$$

where the e_i are iid and drawn from a zero mean, Gaussian distribution with variance σ^2. Using Bayes law we have:

$$P(f|X) = \frac{P(X|f)P(f)}{P(X)}$$

To compute this we have:

$$P(X|f) = \Pi P(X_i|f_i)$$

Note that each X_i is conditionally independent of all labels, given f_i, ie., that

$$P(X_i|f) = P(X_i|f_i)$$

and also that the X_i are independent of each other given the labels, ie., that:

$$P(X|f) = \Pi P(X_i|f)$$

Our noise model states that the $P(X_i|f_i)$ will be a Gaussian distribution with mean f_i, so that:

$$P(X_i|f_i) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{(f_i - X_i)^2}{2\sigma^2}}$$

Therefore, we can define the clique potential:

$$V_c(f_i) = \frac{(f_i - X_i)^2}{2\sigma^2}$$

(Note we can ignore constant factors, since we normalize anyway with Z). These are the unary cliques, which capture the data (pixels). If we want to, we can add to this a prior on different labels.

We can also define a pairwise clique potential to encode the prior that neighboring pixels should have similar intensities. The simplest way to do this is to define:

$$\begin{aligned} For \quad & c = \{i, j\} \\ V_c = 0 \quad & f_i = f_j \\ V_c = k \quad & f_i \neq f_j \end{aligned}$$

This biases us to have piecewise constant regions in the restored image. On the other hand, if we give:

$$V_c = \|f_i - f_j\|$$

we penalize according to the total variation in the image.

Conditional Random Fields

In the way we incorporated the image into an MRF, it was critical that the image information at each pixel, X_i, was conditionally independent of all other pixels, given f_i. To see why this is the case, let's consider a simple example of a 1D image corrupted by *colored* noise. By a 1D image, I mean that S_i is neighbors with S_{i-1} and S_{i+1} only. By colored noise, I mean that the noise at one pixel is correlated with the noise at an adjacent pixel. With white noise, and the MRF formulation we had, knowing f_i would make f_{i+1} conditionally independent of f_{i-1} and X_{i-1}. But this is not true with colored noise. If we know that $f_{i-1} - X_{i-1} = 20$, for example, this tells us that the noise at S_{i+1} is also likely to be high, and might tell us that the expected value of f_{i+1} is larger than would be indicated by f_i and X_{i+1}. That is, even if the labels alone have conditional independence as specified by the neighborhood structure of the MRF, introducing observations that do not have the same conditional independence structure introduces new dependencies in the labels.

This is actually a very common problem. Suppose, for example, that we want to pixels semantically. If a region is labeled *grass*, for example, this tells us it should have a grassy texture. Such a texture cannot be described by pixels that are iid. There will be large scale dependencies between the appearance of pixels. (For example, we often describe textures using distributions on multi-scale filter outputs that can incorporate information from pretty large sets of pixels).

The same issue can arise in stereo matching. We can formulate stereo as an MRF problem. Each pixel is labeled with a disparity. This disparity implies a match between a pixel in the left image and one in the right image. The unary clique potential encodes the similarity of these two pixels (say their squared difference). The pairwise clique potentials can then encode a smoothness prior on the matching, for example, by penalizing neighboring pixels if they are labeled with different disparities. However, in some situations one achieves better performance by comparing windows around pixels, rather than individual pixels. This allows us, for example, to normalize the intensities in a window to diminish lighting effects. In this case, though, the unary potential depends on a window of pixels, so that the unary potential for two non-neighboring pixels is based on many of the same intensities. This again, can remove the conditional independence structure of the MRF.

One solution to this is to use CRFs. In this setting, this can seem pretty simple. We simply take the image information as fixed, and condition on it. That is, rather than

$$P(f_i | f_{S-\{i\}}) = P(f_i | f_{N_i})$$

we assume

$$P(f_i|f_{S-\{i\}}, X) = P(f_i|f_{N_i}, X)$$

Our variables do not form an MRF by themselves, but they do when conditioned on the image data. This is fine for many problems we are interested in, such as finding a MAP estimate of the labels given X.

There is an important difference, though, in that we have moved from a generative model to a conditional model. For example, in the denoising example given above, we have a generative model of f and X. That is, we know the joint distribution of (f, X), so we can sample from this distribution. With a CRF, we no longer try to model the full joint distribution. We have a model for $P(f|X)$, but we do not try to model $P(X)$. This can be good or bad. On the one hand, generative models can be useful. On the other hand, it may be very difficult to model $P(X)$, and if we can't model it accurately we may wind up with a poor generative model that leads to poor inference. One mantra of people who prefer conditional models is: don't try to solve an intermediate problem (building a generative model) that is harder than the problem you actually want to solve (conditional inference).

Computing MAP estimates of MRFs and CRFs

There are several kinds of computations that we might want to perform with MRFs and CRFs. These include learning the clique potentials and other parameters, finding a MAP estimate of the labels, given an MRF/CRF and image information, sampling from the conditional distribution (instead of just getting a MAP estimate). We will focus mainly on the MAP estimate problem, since this is very useful.

Initialization: All these algorithms are iterative, and the results can depend on how the solution (labeling) is initialized. The simplest initialization method is to compute the MAP estimate using only the unary clique potentials. So, for denoising, for example, we would initialize the labels to be the corresponding pixel intensity. Of course, there are many other standard heuristics, such as using domain knowledge (ie., a prior on the labels), or trying many random initializations and picking the one that leads to the best solution.

Iterated Conditional Modes: This is the simplest, greedy algorithm. Visit the sites in some order, and for a given site, S_i, choose the label f_i that maximizes $P(f)$ given that all other labels are fixed. Notice that this only requires computing the clique potentials that include S_i for all possible labels, so this requires computation that is linear in the number of labels. ICM is efficient, but can quickly converge to a poor local minimum. It is likely to work well when the unary clique potentials are very strong (note that in the limit, as the unary clique potentials dominate, ICM produces the global optimum.)

Simulated Annealing: This method uses stochastic optimization to avoid some of the local minima that occur with ICM. The idea is to randomly change a label, and then accept this change with a probability that depends on the extent to which this change increases or decreases the overall probability of the labeling. For example, given a set of nodes with labels, f, we randomly select a site, S_i and consider changing the label to f_i'. We let f' denote this new label set containing f_i'. Let $p = \min(1, P(f')/P(f))$. Then we replace f with f' with probability p. This results in our always accepting a change that improves the labeling, but also in our accepting changes that

reduce the probability of the labeling. This can allow us to escape from labelings that are locally, but not globally optimal.

Note that the distribution $P(f)$ becomes very flat for large values of T, and more peaked as T decreases. To take advantage of this, we use an annealing schedule in which T begins with a high value, and gradually decreases. This means that at first we make changes to the labeling almost at random, often moving to less probable labelings, and then gradually move more deterministically only to more probable labelings. It can be proven that if the annealing schedule is chosen properly this will converge to the globally optimal solution, but of course since the problem is NP-hard, this must require an annealing schedule that uses an exponential amount of time.

Belief Propagation: Belief propagation allows for exact inference in graphical models that do not contain loops, such as Markov chain models or models with a tree structure. It has been shown that this can also lead to effective inference in models with loops, such as MRFs, but we won't discuss this algorithm.

Graph Cuts: