10-708: Probabilistic Graphical Models 10-708, Spring 2012

Spectral Algorithms for Graphical Models

Scribes: Kenton Murray, Shashank Srivastava

1 Motivation

Many problems in machine learning involve large number of variables or observation sequences. These can often be conveniently represented as Latent variable models (LVMs), that feature the observed variables, as well as sets of hidden/latent variables. As we have seen before, inference of model parameters for LVMs using exact maximum likelihood is often intractable; and typically hinges on approximate inference or local search heuristic methods such as Expectation Maximization (EM) . Unfortunately, EM has several drawbacks, the most significant of which is its slow linear convergence. Further, EM can get stuck in bad local minima, and may need several re-initializations to explore the complete parameter space.

Spectral learning approaches provide an alternative representation for several latent variable problems. By setting up the problems in this framework, we can avoid the hard optimizations, while dealing with a different set of tradeoffs. The focus here is on using the model for prediction on observed variables, rather than recovering states of hidden variables. While spectral methods don't model hidden variables and aspects of latent variable models explicitly, they can provide an equivalent representations that lead to algorithms that are efficient, statistically consistent and free of local minima.

We begin with looking at some simple graphical models, which highlight the crux of the spectral learning approach: the close connection between Latent Variable Models and low-rank matrix factorizations.

2 Low Rank factorizations

Spectral methods focus on an alternative *observable representation* of LVMs that does not model the hidden variables. As an example, let us consider the trivial case, where we integrate out all the hidden variables from a HMM. We see that integrating out hidden variables results in a large clique containing all the observable variables (which is the minimal I-map). While this representation of a HMM does not contain hidden variables, it is not useful because of the very large number of parameters. The idea here is that meaningful latent variables model relations between variables with a relatively few number of states. Whether a latent variables model is intrinsically different from a clique of the observed variables depends only on the number of states of the latent variables.

This is evident in the following example illustrated in Figure 1, showing a tree model with three observed variables, each of which can take m values. If the hidden variable H has just one state, the marginal probabilities $P(X_i)$ are the same as the conditionals $P(X_i|H)$, and the variables are independent. At the other extreme, if H takes m^3 states, the situation is the same as having no hidden variable, but a fully connected clique (since the joint CPT for the observed variables has m^3 rows).

The interesting cases are the intermediate ones, where the observed variables are coupled through a smaller number of hidden states. Traditional methods are blind to this perspective, and require EM to learn irre-

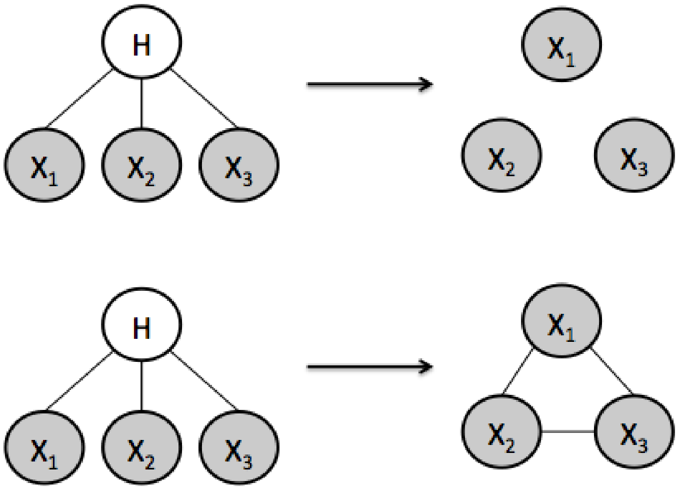
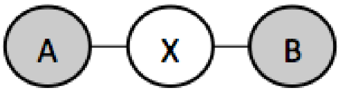


Figure 1:

spective of the number of hidden states. Spectral algorithms deal with this issue, when observed variables are not independent, but are coupled through low rank matrix factors.

2.1 Independence and Rank

We now develop a linear algebra perspective towards joint distributions of variables. Let us consider two variables A and B, each of which can take m values. The joint probability can be represented as an $m \times m$ matrix $\mathcal{P}[A, B]$, where $\mathcal{P}_{ij} = P(A = i, B = j)$. We note that if A and B are independent, the rank of the joint probability matrix is 1 (since all rows/columns would be multiples of each other). On the other hand, the maximum rank of $\mathcal{P}$ can be m.



Now let us consider the case (in Figure 2) when A and B are conditionally independent given an intermediate variable X. Let us say that X takes k states. Then, using the linear algebra formalism, we can express $\mathcal{P}(A, B)$ as:

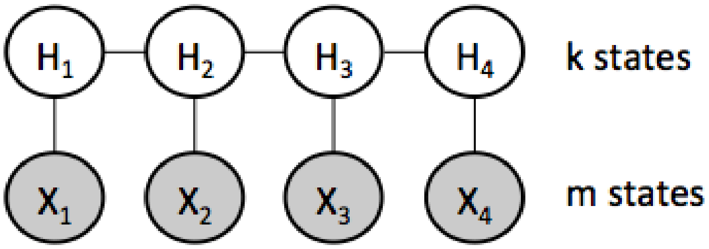
$$\mathcal{P}(A, B) = \mathcal{P}(A|X) \times \mathcal{P}(\oslash X) \times \mathcal{P}(B|X)^T$$

where $\mathcal{P}(\oslash X)$ represents the diagonalized probability matrix for variable X. In the matrix notation, X gets marginalized due to the summations in the two matrix-multiplication operations. Here, $\mathcal{P}(A|X)$ is $m \times k$, $\mathcal{P}(\oslash X)$ is $k \times k$, and $\mathcal{P}(B|X)^T$ is $k \times m$. The principal observation is that in this case, $rank(\mathcal{P}(A, B)) \leq k$, as $\mathcal{P}(A, B)$ is a product of low rank factors.

The crux of the spectral view is that meaningful latent variable models encode *low rank dependecies* among the observed variables, and methods from linear algebra can analyze and exploit this structure.

2.2 HMM example

Let us consider a HMM like chain-LVM shown below:



Let us suppose all hidden variables can take k states, while observable variables take m values. Similar to the factorization seen above, the observable joint probability can be factorized as:

$$\mathcal{P}[X_{\{1,2\}}, X_{\{3,4\}}] = \mathcal{P}[X_{\{1,2\}}|H_2] \times \mathcal{P}[\oslash H_2] \times \mathcal{P}[X_{\{3,4\}}|H_2]^T$$

where the first and third factors involve three variables each, and the second factor involves a single variable (H_2).

The key insight here is that this factorization is not unique. In specific, any invertible matrix and its inverse can be multiplied to two given factors to yield another valid factorization. The basis for spectral learning is that there exists an alternative factorization that depends only on the observed variables. Further, all the factors for this factorization involve at most three observed variables (such as $\mathcal{P}[X_{\{1,2\}}, X_3]$ or $\mathcal{P}[X_2, X_{\{3,4\}}]$).

2.2.1 Alternate factorization

Similar to the factorization in Eqn 1, we can get :

$$\mathcal{P}[X_{\{1,2\}}, X_3] = \mathcal{P}[X_{\{1,2\}}|H_2] \times \mathcal{P}[\oslash H_2] \times \mathcal{P}[X_3|H_2]^T$$

$$\mathcal{P}[X_2, X_{\{3,4\}}] = \mathcal{P}[X_2|H_2] \times \mathcal{P}[\oslash H_2] \times \mathcal{P}[X_{\{3,4\}}|H_2]^T$$

We can see that:

$$\mathcal{P}[X_{\{1,2\}}|H_2] \times \mathcal{P}[\oslash H_2] \times \mathcal{P}[X_{\{3,4\}}|H_2]^T = \mathcal{P}[X_{\{1,2\}}, X_{\{3,4\}}]$$

Also,

$$\mathcal{P}[X_2|H_2] \times \mathcal{P}[\oslash H_2] \times \mathcal{P}[X_3|H_2]^T = \mathcal{P}(X_2, X_3)$$

From the above, we have :

$$\mathcal{P}[X_{\{1,2\}}, X_{\{3,4\}}] = \mathcal{P}[X_{\{1,2\}}, X_3] \times \mathcal{P}[X_2, X_3]^{-1} \times \mathcal{P}[X_2, X_{\{3,4\}}]$$

Thus we have an alternate factorization that depends only on the observed variables (X_i's). This means that these factors can be directly computed from the observed data (without EM!). However, the factors are not restricted to be probability tables (not constrained to be non-negative). This factorization is called the *observable factorization* for the HMM.

Further , each of these factors involves *at most* three observed variables. In fact, it can be proved that for any latent tree model:

- There exists a factorization where all factors are only functions of observable variables
- All factors are of size 3

While the observable representation is advantageous since it can be computed directly from data; the inverse operation presents computational challenges, since small errors in estimating the original matrix can be amplified during the inversion operation.

2.2.2 Existence of the Inverse

We now look at the conditions for the inverse $\mathcal{P}[X_2, X_3]^{-1}$ in Section 2.2.1 to be well defined. We note that:

$$\mathcal{P}[X_2, X_3] = \mathcal{P}[X_2|H_2] \times \mathcal{P}[\oslash H_2] \times \mathcal{P}[X_3|H_2]^T$$

For the inverse to exist, all three factors on the RHS must be full rank. If $m \geq k$, the inverse does not exist. However, this situation can be remedied by projecting through a $k \times m$ projection matrix . If $k > m$, the inverse exists. However, in this case, we have a deeper problem as the following relation used in the derivation of the alternate factorization now does not hold:

$$\mathcal{P}[X_2, X_3]^{-1} = (\mathcal{P}[X_3|H_2]^T)^{-1} \times \mathcal{P}[\oslash H_2]^{-1} \times \mathcal{P}[X_2|H_2]^{-1}$$

This intuitively corresponds to the intractable case when the number of latent states is much larger than the number of observable states.

2.3 Tree Example

2.3.1 Tensor algebra preliminaries

Tensor Factorization is the generalization of the process we have been discussing into multiple dimensions. Matrices represent two dimensions, but when we have higher order graphical models, we cannot represent spectral methods using matrices. Instead, we represent the latent states using tensors that are products of lower order tensors.

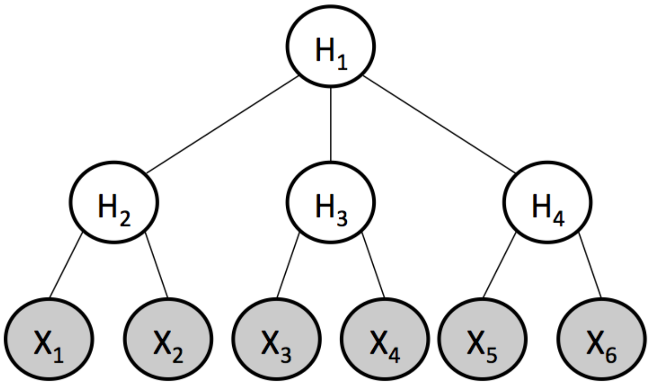
Tensors are the generalization of matrices into multiple dimensions. In particular, a first order tensor is merely a vector and a second order tensor is a standard matrix. An N order tensor has N modes, with each mode associated with a dimension. The dimension of a mode is identical to the dimension of a vector (the first order tensor). Descriptive symbolisms of rectangles, cubes, and circles were used to represent different tensor orders in the lecture. Circles represent fourth and higher order tensors that are not easily depictable.

As with matrices, it is possible to define diagonals for tensors. A diagonal third order tensor $T(i, j, k)$ is only non-zero when $i = j = k$. This generalizes to higer orders. Due to the higher order nature of tensors compared to matrices, it is also possible to define partially diagonal tensors. These are diagonal tensors in lower orders, but not necessarily of the full rank of the tensor. Formally, a third order tensor $T(i,j,k)$ is non-zero when $i = j$ regardless of k, though it can be just for specific values of k as well. The main criteria being that $i \neq k$ && $j \neq k$ are also allowed but not in a fully diagonal tensor.

Tensor vector multiplication is a relatively straight forward extension of matrix mutliplication. For instance multiplying a third order tensor by a first order tensor (vector) yields a second order tensor (matrix). As the dimension is preserved in matrix multiplication, the modes are preserved as well. Similarily, multiplying this resultant second order tensor by another vector (first order tensor) will yield a vector (first order). Tensor multiplication is useful in graphical models as we are able to decompose a fourth order tensor into a product of two third order tensors. It is worth noting that tensor multiplication is not closed under multiplication, which is how two third order tensors can factorize a fourth order. The product of two tensors of order n is a tensor of order $2n-2$. Thus two third order tensors multiplied $= 2*(3)-2 = 6-2 = 4$. Matrix multiplication is closed under multiplication as $2n-2 = 2*(2)-2 = 4-2 = 2$. To be closed under multiplication, the result of the product is the same as the inputs.

2.3.2 Message Passing on Trees

Tensors can be used to represent trees in message passing. In this figure, we can compute the entire marginal tensor using message passing



The tree can be represented as a sixth order tensor, which can be expressed as the product of four third order tensor factors: a third order diagonalization of the root node, H_1, multiplied by the message passed from the leaves. Each of the leaves is a joint probability given the root node that can be represented using a third order tensor.

$$P[X_1, X_2, X_3, X_4, X_5, X_6] = P(\oslash_3 H_1) \times P[X_1, X_2|H_1] \times P[X_3, X_4|H_1] \times P[X_5, X_6|H_1]$$

This comes from the fact that $P[X_1, X_2|H_1] = P(\oslash H_2|H_1) \times P(X_1|H_2) \times P(X_2|H_2)$ This is a third order diagonal tensor multiplied with two second order tensors on the H_2 modes, preserving the third order tensor.

We can also decompose this similar to earlier parts of the lecture. If decompose (cut) the link between H_1 and H_4, we get a factorization of $P[X_{\{1,2,3,4\}}, X_{\{5,6\}}]$. From the earlier part, viewed in a “matricized way”, this is $= P[X_{\{1,2,3,4\}}, X_5]P[X_4, X_5]^{-1}P[X_4, X_{\{5,6\}}]$. In the “tensor way”, this can be thought of as:

$$P[X_1, X_2, X_3, X_4, X_5, X_6] = P[X_1, X_2, X_3, X_4, X_5] \times_{x_5} P[X_4, X_5]^{-1} \times_{x_4} P[X_4, X_5, X_6]$$

The first term can be decomposed recursively:

$$P[X_1, X_2, X_3, X_4, X_5] = P[X_1, X_2, X_3, X_5] \times_{x_3} P[X_5, X_3]^{-1} \times_{x_5} P[X_3, X_4, X_5]$$

And again, the first term from the above equation decomposes as:

$$P[X_1, X_2, X_3, X_5] = P[X_1, X_2, X_3] \times_{x_3} P[X_1, X_3]^{-1} \times_{x_1} P[X_1, X_3, X_5]$$

These are only decomposed into third order tensors as multiplying second order tensors does not increase the tensor order (remember that matrix multiplication is closed under multiplication).

So the final decomposition for the tree (into factors with at most three variables) can given by:

$P[X_1, X_2, X_3, X_4, X_5, X_6] = P[X_1, X_2, X_3] \times_{x_3} P[X_1, X_3]^{-1} \times_{x_1} P[X_1, X_3, X_5] \times_{x_3} P[X_5, X_3]^{-1} \times_{x_5} P[X_3, X_4, X_5] \times_{x_5} P[X_4, X_5]^{-1} \times_{x_4} P[X_4, X_5, X_6]$