An HMM example in GMTK

John Halloran

April 4, 2012

For those interested in how to use GMTK (Graphical Models ToolKit) in order to perform various forms of inference/learning using graphical models, I put together a (really really) short GMTK example. Eventually, I plan on augmenting the problem framework to necessitate other tools that are available in GMTK (ie Continuous child variables (Gaussians and Beta distributions), switching parents, EM for Gaussian distributions, there are certainly many more interesting features that GMTK contains), but for now we will consider a particular problem within which we assume an HMM structure, we have some training data and some multinomial distribution parameters which we'd like to learn, and some testing data with which we would to perform inference over.

1 Problem set up: Weather Identification

Consider the scenario wherein the area you work/reside only encounters three types of weather: sunny, rainy, and foggy. Assume that you go to work in an office everyday, and during your working hours you do not get to see nor experience a given days weather (you're office also has no windows looking outside; this was my office in Hawaii ironically enough). Curiosity mounts as time goes on and you become more and more curious about what the daily weather is. Your hope lies in that you have an office mate who comes in hours later than you, and this office mate brings in an umbrella some days, and no umbrella all other days.

In the framework of the HMM, the hidden layer is the state of the weather (indeed, it is hidden from you). The observed layer consists of the daily observations of weather your office mate has brought an umbrella to the office or not. Formally, let X be a random variable such that $X \in \{0, 1, 2\}$, where state 0 denotes sunny, state 1 denotes rainy, and state 2 denotes foggy. We assume that the Xs are Markov, ie denoting a specific day by t and the corresponding random variable describing the weather of that day as X_t, X_{t+1} is indepedent of X_{t-1} given X_t. Let O be random variable s.t. $O \in \{0, 1\}$, where $O_t = 0$ if your coworker did not bring an umbrella into the office that day and $O_t = 1$ otherwise. The graphical representation of the HMM is available in figure 1.

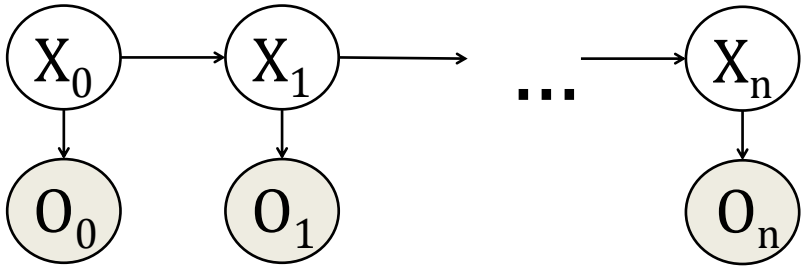


Figure 1: Hidden Markov Model (HMM) of length n. Shaded nodes, O_t are observed, unshaded, X_t are hidden. The Markov chain $X_{t-1} \rightarrow X_t \rightarrow X_{t+1}$ is called the hidden layer, while the lower level of nodes is called the observed layer.

In the graph, O_t is a child of X_t, as is X_{t-1} for $t > 0$ (assume that all observations began at day 0). Thus, after n days, the joint distribution over these variables will be the following:

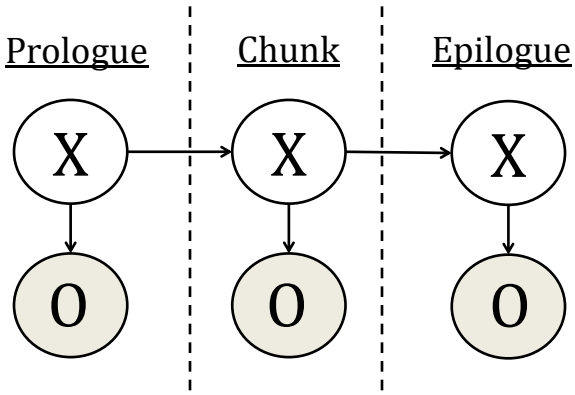
$$\begin{aligned} p(X_0, X_1, X_2, \ldots, X_n, O_0), \ldots, O_n) &= p(X_0)p(X_1|X_0)p(X_2|X_0, X_1)\ldots \\ &\quad p(O_0|X_0, \ldots, X_n)p(O_1|O_0, X_0, \ldots, X_n)\ldots \\ &\quad p(O_n|O_0, \ldots, O_{n-1}, X_0, \ldots, X_n), \quad \text{by Bayes' rule} \\ &= p(X_0)p(X_1|X_0)p(X_2|X_0)\ldots \\ &\quad p(O_0|X_0)p(O_1|X_1)p(O_n|X_n) \\ &= p(X_0)\prod_{t=1}^{n} p(X_t|X_t - 1))p(O_t|X_t) \\ &\quad \text{by the joint density factorization implied by the graph} \end{aligned}$$

2 DBNs and GMTK: training/testing this HMM

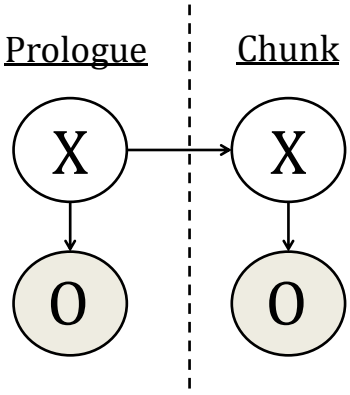
In the context of a DBN, the graph is defined arbitrarily over a sequence of length n wherein each time (day in the weather example) instance is called. When n is instantiated, ie we oberve a particular sequence, the DBN is arbitrarily unrolled to fill all instances of the n sequence samples. In our case, the DBN turns out to be an HMM. The prerequisite definitions for this graph being defined for an arbitrary sequence are a graph which defines the first sample of any given sequence ($t = 0$), and a arbitrary graph for $t > 0$.

In GMTK speak, the first frame is called the prologue, and defines what happens over the first observed sequence sample. The following frame is called the chunk, and this is the workhorse of the DBN. Essentially, the chunk is specified to unroll for the following $0 < t < n$ sequence samples. Finally, the last frame is called the epilogue, and this details what happens in the final sample. One can also imagine that the prologue, chunk, and epilogue can be defined for multiple frames, and this would just amount to describing the dependencies over runs of your observed sequence. The

semantics will remain the same, in that the chunk is specified to unroll and essentially fill the middle portion of a sequence. The prologue, chunk, and epilogue of the HMM are shown below. Note that the epilogue is just a replica of the chunk. There is no restriction which says that the epilogue (nor chunk) need to be defined, and if you can recursively define the graph you are interested in without these you are free to do so in GMTK.



(a) HMM template in GMTK



(b) Equivalent HMM templates in GMTK

Figure 2: HMM templates in GMTK. The two are equivalent since in figure 2a, the chunk is the same as the epilogue, so the chunk need only be unrolled over the last frame as is the case for the template in figure 2b.

2.1 Graph structure definition

To define the structure of the graph in GMTK, we will create a structure file, *hmm.str*. The objects described in this file will be the random variables of the graph. To describe the variables in the prologue of the HMM in figure 2, the following is the gmtk syntax:

```
frame: 0 {
variable : X {
type : discrete hidden cardinality 3;
conditionalparents : nil using DenseCPT("pX");
  }

variable : O {
type : discrete observed 0:0 cardinality 2;
conditionalparents : X(0) using DenseCPT("pO_given_X");
  }

}
```

The frame for which we are describing variables is first defined, frame 0. Within this frame we have two variables, X and O. There are 3 types of random variables in GMTK: discrete and hidden (ie, random in which case we iterate over all values of the random variable during inference), discrete and observed (these variables do not vary during inference, and we could further define them to be deterministic or define a pdf over them), and continuous and observed (not covered here). All hidden variables have a cardinality C associated with them, and it is assumed that they take on values in the range $[0, C]$. In frame 0, $X(0)$ is discrete and hidden. Since this is the first frame, $X(0)$ also has no parents, which is designated by conditonalparents : nil. The marginal distribution over X is a DenseCPT called pX, the symantics of which will be described in the next section.

$O(0)$ is discrete and observed, as evidenced by the type declaration. The value that $O(0)$ takes on is fed into GMTK via what is called the *global observation matrix*. Basically, a file containing the sequence of data is specified on the command line and loaded into memory. The syntax $0 : 0$ designates a range in the global observation matrix. Corresponding to our binary sequence (umbrella observations), the global observation matrix will consist of a single column, and each row will correspond to each frame in the model. So, if we had a sequence 010, then the global observation matrix would be $\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$, the range specification $0 : 0$ would tie each observation to the first element of each row, ie $O(0) = 0$, $O(1) = 1$, and $O(2) = 0$. Finally, $O(0)$ has a single parent $X(0)$, the conditional probability table (DenseCPT) of which is called pO_given_x. One important thing to note is that the index of a random variable in a frame is relative only to its local frame value; the designation $X(0)$ says the value of random variable X in the current frame, whereas $X(-1)$ would say the value of the random variable X is the previous frame. This will become more obvious when describing the next frame (the chunk) of the HMM.

To complete the HMM, we describe the chunk:

```
frame: 1 {
variable : X {
type : discrete hidden cardinality 3;
conditionalparents : X(-1) using DenseCPT("pX_given_X");
  }

variable : O {
type : discrete observed 0:0 cardinality 2;
conditionalparents : X(0) using DenseCPT("pO_given_X");
  }

}

chunk 1:1;
```

Many of the semantics are of the same as we have seen for the prologue; we define random variables tied to a particular frame and describe their conditional dependence relationships. The line chunk $1:1$ says the range of frames which are used for the chunk. Since we only have frame 1 which unrolls for the $n-1$ last data sequences, $1:1$ says to use only frame 1 to arbitrarily unroll upon all but the first observed data sequence. As previously mentioned, the designation of $X(-1)$ as a conditionalparent of X in this frame means that after the chunk has been unrolled, X_0 will be a parent of X_1 corresponding to the prologue and the first unrolled chunk frame, X_1 will be a parent of X_2 corresponding to the first and second unrolled chunk frames, and so on and so forth.

2.2 Trained parameter files: defining CPTs

There are several ways of representing probability distributions in GMTK. We'll be Ignoring continuous distributions (although might add an example using these later). The most natural way of specifying a discrete distibution would be via a table/matrix of probability values. This is called a *DenseCPT* in GMTK. The syntax of these objects is as follows

```
number of DenseCPTs to follow

%notation for a CPT:
cpt_index
cpt_name
number of parents
cardinality of parents
cardinality of self
dense_cpt values
```

where % denotes a comment. First, we define the number of DenseCPTs to follow, say N. Moving on to define a particular CPT, we begin by referring to that CPTs index (its in-file rank starting from 0 and ending at $N-1$). Next we define the name of the CPT, followed by the number of parents of the random variable the CPT describes, followed by the cardinality of the random variable's parents, followed by the cardinality of the random variable itself, and finally the matrix of probability values. For instance, the following would be the definition of all the HMM CPTs set to uniform probability distributions.

```
% Dense CPTs
3
0
pX
0 % number parents
3 % cardinalities
0.33333333333 0.33333333333 0.33333333333

1
pX_given_X
1 % number parents
3 3 % cardinalities
0.33333333333 0.33333333333 0.33333333333
0.33333333333 0.33333333333 0.33333333333
0.33333333333 0.33333333333 0.33333333333

2
pO_given_X
1 % number parents
3 2 % cardinalities
0.5 0.5
0.5 0.5
0.5 0.5
```

3 GMTK training and testing

Two bash scripts, train_cmd.sh and test_cmd.sh, or included in the tarball and should be runnable upon untarring. To properly run these, make sure the gmkt binary (or at least gmtkEMtrain and gmtkViterbi) are in your path.

In order to infer the maximum probability assignment of the hidden layer to explain the observed layer (called Viterbi decoding, wherein we calculate the maximum configuration of hidden states with respect to all possible hidden configurations), we would first like to train our data. GMTK features EM training both for multinomial distributions (using Dirichlet distributions) and Gaussian

distributions (just using EM applied to Gaussians). To proceed with training our multinomial distributions, one can specify a file of initial parameters for the distributions. In the accompanying tarball, this file is called init_hmm.params and follows the semantics of the section describings CPTs. Next, the binary gmtkEMtrain is called with various switches and parameters. To take a look at this call, take a look at train_cmd.sh. The switches and their definitions are:

strFile	the the structure file
inputMasterFile	the master file (empty for this example)
inputTrainableParameters	file of initial CPT values
outputTrainableParameters	file to write output CPT values
of1	observed file 1, file containing global list of sequences (one file per sequence in ascii)
fmt1	format for observed file 1, ascii in our case
nf1	number of floats in observed file 1 (0)
ni1	number of ints in observed file 1 (1, umbrella/no umbrella)
dirichletPriors	boolean, use or don't use dirichlet priors (T is true)
maxE	maximum number of EM iterations
lldp	threshold at which to stop based on consecutive values between iterations
objsNotToTrain	file listing CPTs not to train
random	boolen, randomly initialize all values (T is true)

Once training completes, we're ready to calculate the Viterbi assignment! To do so, we'll be using the binary gmtkViterbi. The bash script which calls this binary is test_cmd.sh. Many of the switches are the same as those for gmtkEMtrain (and gmtk binaries in general). The switches which are distinct and worth noting are:

verbosity	integer value specification ranging from 0 to 99; output to terminal, ranges from showing the log-likelihood probabilities (verb 10), to showing the instantiation of variables in message-passing (verb 66), to the probability of variables in message-passing (verb 86), and so on.
vitValsFile	file to wrie the viterbi path to

All training and testing data are contained in the tarball subdirectory data, which also contains the matlab script used to generate the data. The true labels are also included in the data directory, both for the testing and training data. Running test_cmd.sh will perform a verification of the Viterbi assignment versus the ground truth labels.

I was also planning on expanding on this example, firstly by adding structure (Markovian) to the generated data, such that the data generation of the true hidden weather was a first over Markov Model. This would allow for much better verification (I was able to achieve 70% accuracy

with the current data generation, which is simply to generate iid weather). Also some examples with switching parents and deterministic mappings are in the works. More to come!