Graphical Models

Name: Kunal Lad
UT Eid: KL28697
Email: kunal.lad@utexas.edu

Objective

The goal of this assignment is to study inference learning in graphical models. Main objective of the assignment is to reconstruct probability distribution for hidden variable form observed values using Expectation Maximization algorithm.

Introduction

A Bayesian network is a probabilistic graphical model that represents a set of random variables and their conditional dependencies via a directed acyclic graph (DAG). For the bayesian network shown in fig1 we can compute the joint probability distribution as follows:

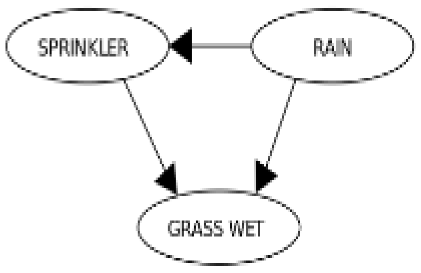


Fig 1 : Example of a Bayesian Network

$$\mathrm{P}(G, S, R) = \mathrm{P}(G \mid S, R)\mathrm{P}(S \mid R)\mathrm{P}(R)$$

Because of the dependencies among the variables, the joint probability distribution can be computed from conditional probability tables for individual expressions. This is significantly reduces the number of parameters required to be learnt for task of estimating the joint distribution.

If we have a set of data points with values for all the variables, then we can learn the parameters which maximize the likelihood of observed data. This is called Maximum Likelihood Estimate (MLE).

For estimating maximum likelihood parameters from partially observed data, we use Expectation Maximization algorithm. It consists of 2 main steps:

- **Expectation Step:** In this step we infer the distribution over the hidden variables given the observed variables using belief propagation. Each sample in training set is used as observed evidence to execute the sum-product algorithm on the model, using an estimate for the model's parameters. Marginal probability distribution extracted from above is then used to fill in missing information for hidden nodes in the sample.

- **Maximization Step:** In this step we use the inferred distribution and the observed variables to update the parameters of the model.

These 2 steps are repeated until convergence or a fixed number of iterations has been completed.

To evaluate the similarity of the estimated distribution to original distribution we compute the Kullback–Leibler (KL) divergence. KL divergence of *Q* from *P* is defined as:

$$D_{\mathrm{KL}}(P\|Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)}.$$

Methodology

Bayesian Network

We use the simple bayesian network shown in fig 2 for this assignment. For performing belief propagation this directed graph is converted into an undirected graph by moralization.

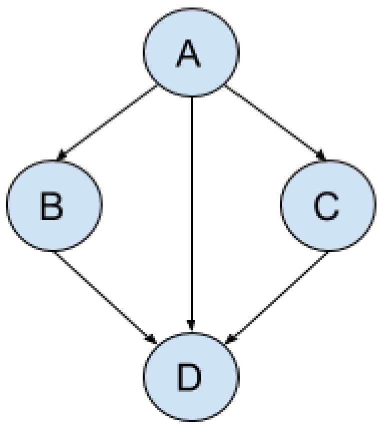


Fig 2: Bayesian Network used for this assignment

Data Details

1. We generate the data samples using sample_bnet method in Bayes Net Toolbox for matlab. This method generates samples for a given bayesian network.
2. We hide the values of one of the variables in a predefined proportion of samples.

Algorithm

1. Initialize each potential $\Phi_i(X_s)$ to 1 and the Expected Sufficient Statistics $ESS_i(X_s)$ for each potential to zero.
2. For every sample S_j in the sample set S execute the following steps:
   If the sample is fully observed
   - Then increase the value of the entry it indexes in $ESS(X_s)$ by 1.

   Else
   - Use the sample S_j as evidence and execute the sum-product algorithm on the Graphical model using the current value of the potentials as its parameters.
   - Obtain marginal distribution over hidden variables in the sample.
   - Reshape marginal distribution to match the dimensions of $ESS_i(X_s)$ by filling in new entries with 0.
   - Add reshaped marginal to $ESS(X_s)$
3. Normalize $ESS(X_s)$ and assign it to $\Phi(X_s)$ then reset $ESS(X_s)$ to 0
4. Check for convergence of log likelihood and return to step if not converged.

Implementation Details

1. We used the open source Bayes Network Toolbox by Kevin Murphy. It provides methods for belief propagation, Maximum Likelihood Estimation, EM etc.

2. First we build the DAG shown in fig 2, where each node is a discrete binary variable.

3. Next we build a Bayesian net from this DAG and CPT values.

4. Next we create a dataset by using sample_bnet method on bnet created in step 3.

5. Predict MLE estimate for the dataset using learn_params method from Bayesian Toolbox.

6. Randomly hide values of of variable D and run EM algorithm on partial dataset through learn_params_em method.

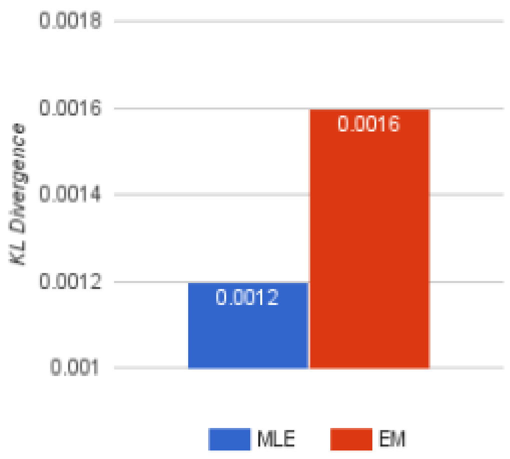
7. Compute KL divergence for both MLE and EM estimates of probability distributions with the original distribution.

Experiments and Results

This section gives details of the various experiments performed in this assignment to understand inference in graphical models. For all the experiments:

- We used the bayesian network in fig 2
- Made D as unobserved variable in some examples.
- Computed MLE and EM probability distribution estimates.
- Computed KL divergence of predicted and actual probability distributions to evaluate the quality of results.

E1: Performance of MLE vs EM



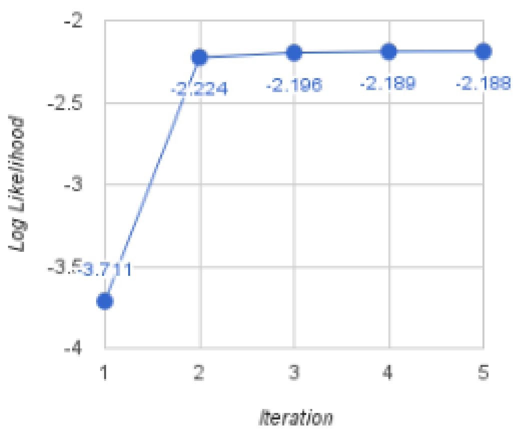


Fig 3a: KL Divergence of MLE and EM estimates. **Fig 3b: EM Log Likelihood vs Iteration**

In this experiment we compared the performance of MLE against EM. We used 10000 samples for MLE and hide the values of node D in half of the samples for EM. Figure 3 shows the results. As expected the divergence of MLE estimate is less than EM estimate because MLE is performed on full data and hence it is more accurate. From figure 3b we see that Log Likelihood based estimate converges after 2 iterations since our graph is a directed acyclic graph.

E2: Performance variation with amount of unobserved data

In this experiment we studied how the EM model estimate varies with number of unobserved samples. We took a dataset of 5000 samples and varied the percent of unobserved the samples from 10 to 90. Plot in Fig 4 shows the results. We can see that as the proportion of unobserved samples increases, KL divergence of estimated probability distribution increases since it is harder to reconstruct original distribution from lesser data.

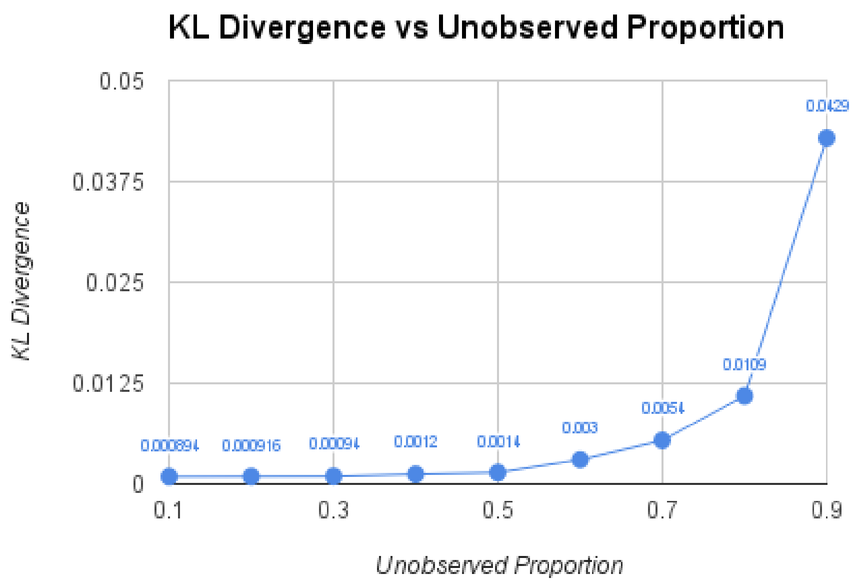


Fig 4: KL Divergence vs Proportion of unobserved samples.

E3: Performance variation with amount of training data

In this experiment we keep the proportion of unobserved samples fixed at 0.5 and study how the EM model estimate varies with increase in number of samples. Plot in Fig 5 shows the

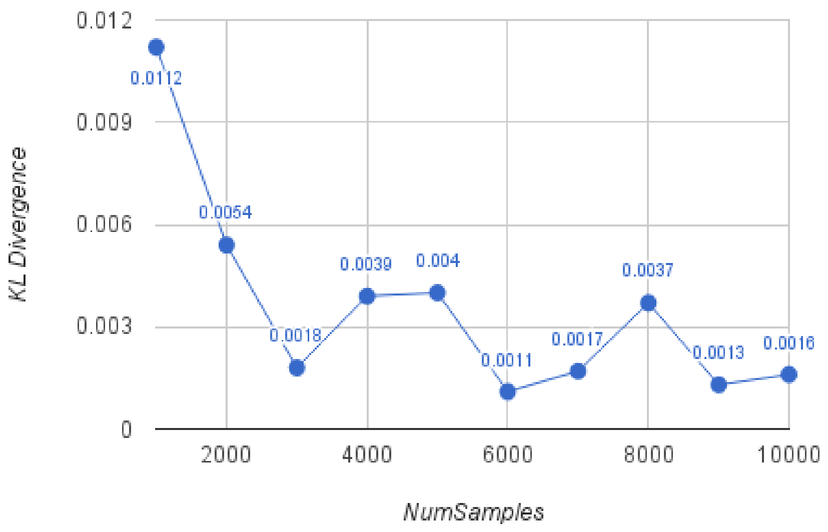


Fig 5 Performance of different kernels with pca, image and hog features

results. We can see that as the proportion of unobserved samples increases, KL divergence of estimated probability distribution increases since it is harder to reconstruct original distribution from lesser data.

E4: How does probability distribution affect model performance ?

In this experiment we compared performance of EM on two different types of probability distributions. In uniform distribution all the CPT values are equal, whereas in skewed distribution values are heavily skewed. Results in Fig 6 suggest that it is easier to learn parameters of skewed distribution due to lower variability in data as compared to uniform random distribution.

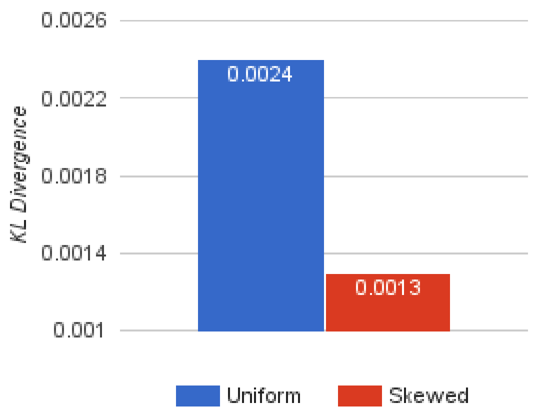


Fig 6 Performance of EM on Uniform vs Skewed probability distribution

NOTE : Detailed statistics of all the experiments can be found here

Discussion and Conclusions

In this assignment we looked at inference and learning on a small graphical model with single unobserved variable. On the basis of experimental findings we can conclude that:

- **What factors affect stability of EM ?** Stability of EM is characterized by number of training samples, proportion of unobserved data and the probability distribution we are trying to estimate.
- **Under what circumstances does EM converge to original data ?** EM converges to original data if we have sufficient number of training samples representative of original distribution and the EM doesn't get stuck in local optimas.
- MLE estimate from full data is better than EM estimate from partial estimate.

References

[1] Pattern Recognition And Machine Learning by Christopher Bishop
[2] A Python Implementation of Graphical Models
[3] Wikipedia
[4] Bayes Net Toolbox Matlab