52935 - Probabilistic Methods in Artificial Intelligence **(Due: 27/06/24)**

Project 2 - Inference

Instructor: Prof. Gal Elidan *TA:* Ela Fallik *Name:* Hadar Tal

Part I: Theoretical Questions

1. Extreme Cases of the Mutilated Network

1.1 Assuming we see evidence $e_r = \{M = m_0, F = f_1\}$

(a) What is the graph of the mutilated network B_{e_r}? Which CPDs have changed?

(b) Show that the proposal distribution is equal to the posterior $p_B(X \mid e_r)$ in this case.

(c) What are the IS weights?

(d) Is $q = p_{B_e}$ a good choice for the proposal distribution in this case?

1.2 Assuming we see evidence $e_l = \{n = n_1, L = l_1\}$

(a) What is the graph of the mutilated network B_{e_l}? Which CPDs have changed?

(b) Show that the proposal distribution is equal to the prior $p_B(X)$ in this case.

(c) What are the IS weights?

(d) Is $q = p_{B_e}$ a good choice for the proposal distribution in this case?

1.3 Conclusion

Use these two extreme cases to conclude, for a general BN, when $q = p_{B_e}$ will be a good proposal distribution.

2. Data Association

1. Compute the acceptance probability $A(c, c')$ for each MH step.

   The acceptance probability $A(c, c')$ for transitioning from state c to c' is given by:

   $$A(c \to c') = \min\left(1, \frac{\pi(c')T(c' \to c)}{\pi(c)T(c \to c')}\right)$$

   where:

   - $\pi(c)$ is the target distribution at state c.
   - $T(c \to c')$ is the proposal distribution for moving from state c to state c'.

   In our case, the target distribution $\pi(c)$ is the posterior $p(c \mid v_1, \ldots, v_K)$, which can be expressed using Bayes' theorem as:

   $$p(c \mid v_1, \ldots, v_K) = \frac{p(v_1, \ldots, v_K \mid c)p(c)}{p(v_1, \ldots, v_K)}$$

   Since $p(v_1, \ldots, v_K)$ is a normalizing constant and does not affect the ratio, it can be ignored for the acceptance probability calculation. Thus, we have:

   $$\frac{p(c' \mid v_1, \ldots, v_K)}{p(c \mid v_1, \ldots, v_K)} = \frac{p(v_1, \ldots, v_K \mid c')p(c')}{p(v_1, \ldots, v_K \mid c)p(c)}$$

   Assuming a uniform prior $p(c)$ over all permutations:

   $$p(c) = p(c') \implies \frac{p(c')}{p(c)} = 1$$

   Therefore, the acceptance probability simplifies to:

   $$A(c \to c') = \min\left(1, \frac{p(v_1, \ldots, v_K \mid c')}{p(v_1, \ldots, v_K \mid c)} \cdot \frac{T(c' \to c)}{T(c \to c')}\right) \overset{\text{uniform transition}}{=} \min\left(1, \frac{p(v_1, \ldots, v_K \mid c')}{p(v_1, \ldots, v_K \mid c)}\right)$$

   Given the independence of observations given the correspondences:

   $$p(v_1, \ldots, v_K \mid c) = \prod_{i=1}^{K} p(V_i \mid C_i = c_i)$$

   Thus, the acceptance probability can be expressed as:

   $$A(c \to c') = \min\left(1, \frac{\prod_{i=1}^{K} p(V_i \mid C_i = c'_i)}{\prod_{i=1}^{K} p(V_i \mid C_i = c_i)}\right)$$

   Since c and c' differ only by the swap of two correspondence variables C_i and C_j, for most objects, the terms in the numerator and denominator cancel out, except for V_i and V_j. Therefore, the formula simplifies to:

   $$A(c \to c') = \min\left(1, \frac{p(V_i \mid C_i = c'_i) \cdot p(V_j \mid C_j = c'_j)}{p(V_i \mid C_i = c_i) \cdot p(V_j \mid C_j = c_j)}\right)$$

   This simplification arises because c and c' differ only in the assignments of C_i and C_j. The proposal distribution is symmetric, meaning $T(c \to c') = T(c' \to c)$. Therefore, the acceptance probability further simplifies to:

   $$A(c \to c') = \min\left(1, \frac{p(V_i \mid C_i = c'_i) \cdot p(V_j \mid C_j = c'_j)}{p(V_i \mid C_i = c_i) \cdot p(V_j \mid C_j = c_j)}\right)$$

   This formula gives us the acceptance probability for each MH step, ensuring the correctness and convergence of the algorithm.

2. Suppose we have run the MH sampler for a long time and collected M samples $\{(C_1^{[m]}, \ldots, C_K^{[m]})\}_{m=T+1}^{T+M}$ after the chain has mixed. Give an explicit expression for estimating the posterior $p(C_i \mid v_1, \ldots, v_K)$.

   The posterior probability $p(C_i = k \mid v_1, \ldots, v_K)$ for a specific correspondence variable C_i taking the value k can be estimated as the fraction of samples where $C_i = k$. Mathematically, this is given by:

$$p(C_i = k \mid v_1, \ldots, v_K) \approx \frac{1}{M} \sum_{m=T+1}^{T+M} \delta(C_i^{[m]}, k)$$

   where $\delta(C_i^{[m]}, k)$ is the Kronecker delta function, defined as:

$$\delta(C_i^{[m]}, k) = \begin{cases} 1 & \text{if } C_i^{[m]} = k \\ 0 & \text{if } C_i^{[m]} \neq k \end{cases}$$

   Thus, the posterior distribution $p(C_i \mid v_1, \ldots, v_K)$ can be estimated as:

$$p(C_i \mid v_1, \ldots, v_K) \approx \left\{ \frac{1}{M} \sum_{m=T+1}^{T+M} \delta(C_i^{[m]}, k) \right\}_{k=1}^{K}$$

   This expression gives an explicit method for estimating the posterior distribution of the correspondence variables after collecting M samples from the MH sampler, leveraging the convergence property of MC (which converges to the true posterior as $M \to \infty$).

3. Your fellow student hears about your MH algorithm and suggests that you can also consider using Gibbs sampling to compute your marginals. Will this work? Explain.

   consider $K = 2$.

   The possible states are:

$$c_1 = (1, 2), \quad c_2 = (2, 1)$$

   In Gibbs sampling, each variable is updated in turn by sampling from its conditional distribution given the other variables. Suppose we start in state $c_1 = (1, 2)$. When we update C_1, we sample from $p(C_1 \mid C_2 = 2)$. Since $C_2 = 2$, C_1 must be 1, so no change occurs. Similarly, updating C_2 by sampling from $p(C_2 \mid C_1 = 1)$ keeps C_2 at 2. Thus, the state remains $c_1 = (1, 2)$.

   If we start in state $c_2 = (2, 1)$, updating C_1 by sampling from $p(C_1 \mid C_2 = 1)$ keeps C_1 at 2, and updating C_2 by sampling from $p(C_2 \mid C_1 = 2)$ keeps C_2 at 1. Thus, the state remains $c_2 = (2, 1)$.

   This shows that Gibbs sampling gets stuck in the initial state and cannot transition between $(1, 2)$ and $(2, 1)$. Therefore, the process is not irreducible.

   For larger K, the situation remains similar. If K is even, Gibbs sampling can only reach even permutations (those that can be achieved by an even number of swaps), and if K is odd, it can only reach odd permutations. This restricts the state space significantly.

   In contrast, the Metropolis-Hastings algorithm with the swap proposal can move between any states, ensuring irreducibility and the ability to explore the entire state space.

   Thus, Gibbs sampling will not work for this data association problem, whereas the Metropolis-Hastings algorithm is appropriate.

3. Block Gibbs

1. What will happen if we try to use Gibbs sampling on B to estimate $p(x_1 \mid z_1)$?
2. Now suppose we make Z a noisy XOR of its parents. Specifically,

$$p(z_1 \mid X, Y) = \begin{cases} \epsilon & \text{if } x = 0, y = 0 \\ 1 - \epsilon & \text{if } x = 0, y = 1 \\ 1 - \epsilon & \text{if } x = 1, y = 0 \\ \epsilon & \text{if } x = 1, y = 1 \end{cases}$$

   What is the expected number of iterations until a state transition (i.e., from one state of the chain - some instantiation x, y, z - to a different state) as a function of ϵ? What can you conclude about problems that Gibbs sampling might encounter in this scenario?

3. Alternatively, we can think of a variant of Gibbs sampling where larger steps are taken. Specifically, larger sets of variables are sampled simultaneously while the rest are fixed. Show that sampling two variables given the third overcomes the problem of Gibbs sampling in the deterministic XOR network.
4. To use this last sampler, we need to calculate the probability of a pair of variables given the rest. Write down a (simplified as possible) formula for $p_\Phi(X_i, X_j \mid X \setminus \{X_i, X_j\})$ for a general Gibbs distribution p_Φ.

4. Collapsed Particles

1. Show that
$$E_{p(X|e)}[f(X)] = \sum_{x_p} p(x_p \mid e) E_{p(X_d|x_p,e)}[f(x_p, X_d, e)]$$
Use this to explain how to estimate $E_{p(X|e)}[f(X)]$ using the collapsed particles.
2. Describe what this approach is equivalent to in the two extreme cases: When $X_p = \emptyset$ and when $X_p = X$.
3. Describe what will be a good "rule-of-thumb" in choosing X_p and X_d for this method.
4. Give an example of a Bayesian Network over X, a set of observed variables E and a query $p(X \mid e)$ that is hard to calculate directly, but can be estimated using the described method of collapsed particles using efficient calculations only. X_d and X_p should be $\neq \emptyset$ in your example.
5. In many cases, sampling from $p(X_p \mid e)$ is still hard. We therefore want to build a (Normalized) Importance Sampling version of this method. We do this by sampling x_p from a proposal distribution $q(X_p)$ and using the weights
$$w(x_p) = \frac{p(x_p, e)}{q(x_p)} = \frac{p(x_p, e_p) \cdot p(x_d \mid x_p, e_p)}{q(x_p)}$$
where $E_p = E \cap X_p$ and $E_d = E \cap X_d$. Prove the correctness of this method. Specifically, show that $E_q[w(x_p)] = p(e)$, and conclude that this procedure gives an estimation of $E_{p(X|e)}[f(X)]$.

Part II

Programming Assignment - Inference in HMM

1. Exact inference using dynamic-programming (DP)

1. You are supplied with code for the forward and backward algorithms. Use them to implement the `log_posterior_Xt` method in the HMM class.

$$P(X_t = k \mid O_{1:t}) = \frac{P(X_t = k, O_{1:t})}{P(O_{1:t})} = \frac{P(X_t = k, O_{1:t}) \cdot P(O_{t+1:T} \mid X_t = k)}{P(O_{1:T})}$$

$$\begin{aligned}\log P(X_t = k \mid O_{1:t}) &= \log\left(\frac{P(X_t = k, O_{1:t}) \cdot P(O_{t+1:T} \mid X_t = k)}{P(O_{1:T})}\right) \\ &= \log P(X_t = k, O_{1:t}) + \log P(O_{t+1:T} \mid X_t = k) - \log P(O_{1:T}) \\ &= \mathcal{F}[t,k] + \mathcal{B}[t,k] - \text{log\_likelihood}(O_{1:T})\end{aligned}$$

Implemented the `log_posterior_Xt` method in the HMM class to calculate the log posterior $P(X_t = k \mid O_{1:t})$ for each t and k.

2. Exact posteriors: We load the observations from `obs_data.csv`, calculate the marginal posteriors for the $N = 20$ observations in the dataset $p(X_t = 1|o_{1:T}[i])$, and plot the prior distributions $p(X_t = 1)$ versus the mean marginal posteriors $\mu_t = \frac{1}{N}\sum_{i=1}^{N} p(X_t = 1|o_{1:T}[i])$ for each t.
What are the differences and why do they exist? What does it suggest about the distribution from which the observations were sampled?

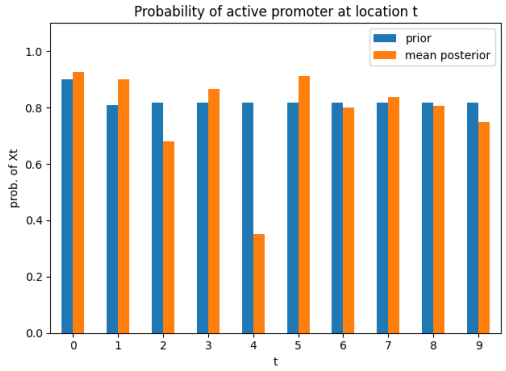

Figure 1: Prior distributions $p(X_t = 1)$ versus the mean marginal posteriors $\mu_t = \frac{1}{N}\sum_{i=1}^{N} p(X_t = 1|o_{1:T}[i])$ for each t.

The plot shows the prior distributions $p(X_t = 1)$ (blue bars) versus the mean marginal posteriors μ_t (orange bars) for each time step t. The differences between the prior and the mean marginal posteriors suggest that the observations have a significant impact on the estimated probabilities of active promoters. Specifically, the mean marginal posteriors differ from the prior distributions due to the evidence provided by the observations. This indicates that the dataset likely contains informative observations that influence the posterior estimates.
The deviations from the prior suggest that the distribution from which the observations were sampled has regions with varying levels of promoter activity, as captured by the posterior probabilities.

3. We want to use the marginal posteriors $p(X_t|o_{1:T}[i])$ to assign each location t its status and predict areas of active promoter:

$$\hat{X}_t[i] = \arg\max_x p(X_t = x|o_{1:T}[i])$$

Implement a `naive_predict_by_posterior` method in the HMM class.

Implemented the `naive_predict_by_posterior` method in the HMM class.

4. Prediction: We use this method to predict hidden sequences for the given observations. We compare them to the ground-truth sequences from `hidden_data.csv`. What is the accuracy of these predictions (use the same definition as Project 1)? What is the difference compared to the prediction we used in Project 1? Is the rule in Eq.4 a good prediction rule?

   Results:

   - Exact Posterior Prediction Accuracy: 0.900
   - Naive Prediction Accuracy: 0.895

   Analysis:

   - The exact posterior method is slightly more accurate than the naive method.
   - The exact method uses the forward-backward algorithm, considering the entire observation sequence, while the naive method likely uses a simpler, less contextual approach.
   - This accuracy difference shows the exact posterior is better at capturing dependencies in the data.
   - Eq. 4's rule is a good prediction method, leveraging the full observation sequence for more informed predictions.

2. Sampling-based inference

1. Gibbs sampling: In this algorithm, we'll start with a starting point $X_{1:T}$ sampled from the prior $p(X_{1:T})$, and sample M sequences of hidden states sequentially:

$$X_t^{(m)} \sim p(X_t | X_{-t}^{(m-1)}, o_{1:T}[i]), \quad t = 1, \ldots, T$$

   We then use the samples to estimate the posterior $p(X_t = x | o_{1:T}[i])$ for each t.

   (a) For observations $o_{1:T}[i]$, at iteration m, we want to use the samples from the previous iteration $\{X_t^{(m-1)}\}_{t=1}^T$ to sample $X_t^{(m)}$ from the distribution $p(X_t | X_{-t}^{(m-1)}, o_{1:T}[i])$. How can we sample from this distribution using the CPDs of the network?

   In a Hidden Markov Model (HMM), the Markov blanket of a hidden state X_t includes its immediate neighbors and the observation at time t. Specifically, the Markov blanket of X_t consists of X_{t-1}, O_t, and X_{t+1} (if they exist). This is because in an HMM, X_t is conditionally independent of all other variables given these three variables.
   Given this, we can write the conditional distribution for sampling X_t as follows:

$$\begin{aligned}
&P(X_t = k \mid X_{-t}^{(m-1)}, O_{1:T}[i]) \\
&= P(X_t = k \mid X_{t-1}^{(m-1)}, O_t[i], X_{t+1}^{(m-1)}) \\
&= \frac{P_\phi(X_t = k, X_{t-1}^{(m-1)}, O_t[i], X_{t+1}^{(m-1)})}{\sum_{x_t \in \mathrm{Val}(X_t)} P_\phi(X_t = x_t, X_{t-1}^{(m-1)}, O_t[i], X_{t+1}^{(m-1)})} \\
&= \frac{P_\phi(X_{t-1}^{(m-1)}) \cdot P(X_t = k \mid X_{t-1}^{(m-1)}) \cdot P(O_t[i] \mid X_t = k) \cdot P(X_{t+1}^{(m-1)} \mid X_t = k)}{\sum_{x_t \in \mathrm{Val}(X_t)} P_\phi(X_{t-1}^{(m-1)}) \cdot P(X_t = x_t \mid X_{t-1}^{(m-1)}) \cdot P(O_t[i] \mid X_t = x_t) \cdot P(X_{t+1}^{(m-1)} \mid X_t = x_t)} \\
&= \frac{P(X_t = k \mid X_{t-1}^{(m-1)}) \cdot P(O_t[i] \mid X_t = k) \cdot P(X_{t+1}^{(m-1)} \mid X_t = k)}{\sum_{x_t \in \mathrm{Val}(X_t)} P(X_t = x_t \mid X_{t-1}^{(m-1)}) \cdot P(O_t[i] \mid X_t = x_t) \cdot P(X_{t+1}^{(m-1)} \mid X_t = x_t)}
\end{aligned}$$

   (b) Given M samples $\{X_t^{(m)}\}_{m=1}^M$ from the process described above, how can we calculate the posterior $p(X_t = x | o_{1:T}[i])$, while discarding the first M_{start} samples to allow for a "burn-in" period?
   To calculate the posterior $p(X_t = x | o_{1:T}[i])$ for each t, we can use the samples $\{X_t^{(m)}\}_{m=M_{\text{start}}}^M$ obtained from the Gibbs sampling process. We can estimate the posterior by counting the number of times $X_t = x$ occurs in the samples and normalizing by the total number of samples. The posterior can be calculated as follows:

$$p(X_t = x|o_{1:T}[i]) = \frac{1}{M - M_{\text{start}}} \sum_{m=M_{\text{start}}}^{M} \mathbb{I}(X_t^{(m)} = x)$$

where $\mathbb{I}(\cdot)$ is the indicator function that returns 1 if the condition is true and 0 otherwise.

(c) Implement a method `gibbs_sampling_posterior` in the HMM class to calculate the posterior $p(X_t = x|o_{1:T}[i])$ for each t with $M - M_{\text{start}}$ samples.

Implemented the `gibbs_sampling_posterior` method in the HMM class.

(d) We run the Gibbs algorithm with $M \in [10, 50, 70, 100, 200, 300, 500]$ samples to calculate the marginal posteriors of each observation in the dataset. We run the algorithm 5 times for each value of M. We plot the posterior of X_5 for the first 10 observations $p(X_5|o_{1:T}[1:10])$ versus M with confidence intervals $(\mu \pm \sigma)$, and add to the plot the exact posterior calculated using DP. Does the Gibbs estimation converge? Does it converge to the correct posterior? Explain your results. What can you say about the convergence of this algorithm?

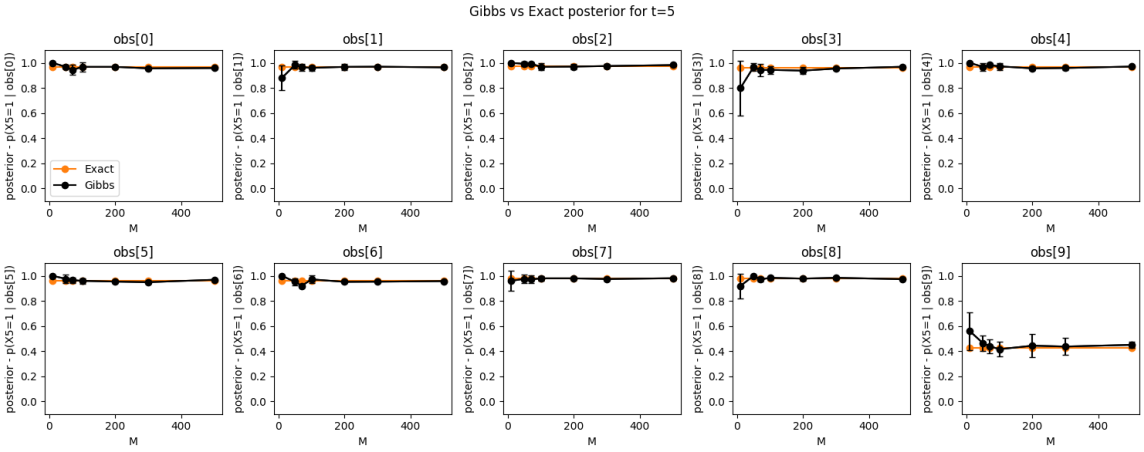

Figure 2: Posterior of X_5 for the first 10 observations $p(X_5|o_{1:T}[1:10])$ versus M with confidence intervals $(\mu \pm \sigma)$.

The plot shows the comparison between the Gibbs sampling estimation and the exact posterior calculated using dynamic programming (DP) for X_5 across the first 10 observations.

Analysis:

- **Convergence:** The Gibbs estimation converges fairly fast as M increases. Even with a relatively small number of samples (e.g., $M = 50$), the estimates are quite stable and close to the exact posterior.
- **Correctness:** The Gibbs estimates converge to values close to the exact posterior calculated using DP, indicating that the Gibbs sampling method is correctly estimating the posterior distributions.
- **Results:** The convergence of the Gibbs sampling algorithm demonstrates its effectiveness in approximating the true posterior distribution, especially with a higher number of samples M. The confidence intervals also decrease with increasing M, showing more precise estimates.
- **Implications:** The fast convergence of the Gibbs sampling method makes it a practical and efficient approach for posterior estimation in HMMs, providing reliable results with relatively few samples.

2. We use the estimated posteriors of each algorithm and the `naive_predict_by_posterior` method to assign each location its status (same as for the exact posterior). We plot the accuracy of the algorithms as a function of M, and plot on the same graph the accuracy of the exact posterior (as a constant function of M). What is the trend?

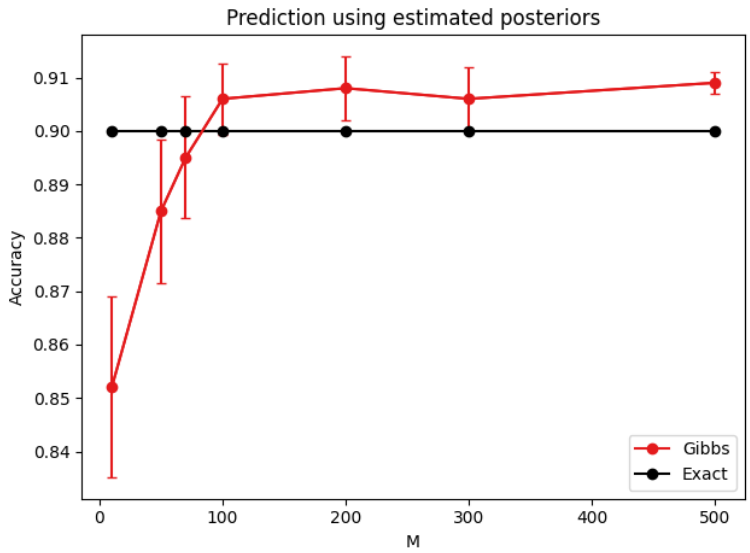

Figure 3: Accuracy of the algorithms as a function of M, with the accuracy of the exact posterior as a constant function of M.

The plot shows the accuracy of the Gibbs sampling algorithm compared to the exact posterior as a function of the number of samples M.

Analysis:

- **Trend:** The accuracy of the Gibbs sampling algorithm increases rapidly with the number of samples M and stabilizes around $M = 100$. The exact posterior accuracy remains constant.
- **Comparison:** The Gibbs sampling method eventually reaches and slightly surpasses the accuracy of the exact posterior, indicating that it is an effective method for estimating the posterior, especially with a larger number of samples.
- **Implications:** The Gibbs sampling method can achieve high accuracy with sufficient samples, making it a reliable alternative to exact inference methods. The slight fluctuation in accuracy with smaller M values suggests that more samples are needed for stabilization.