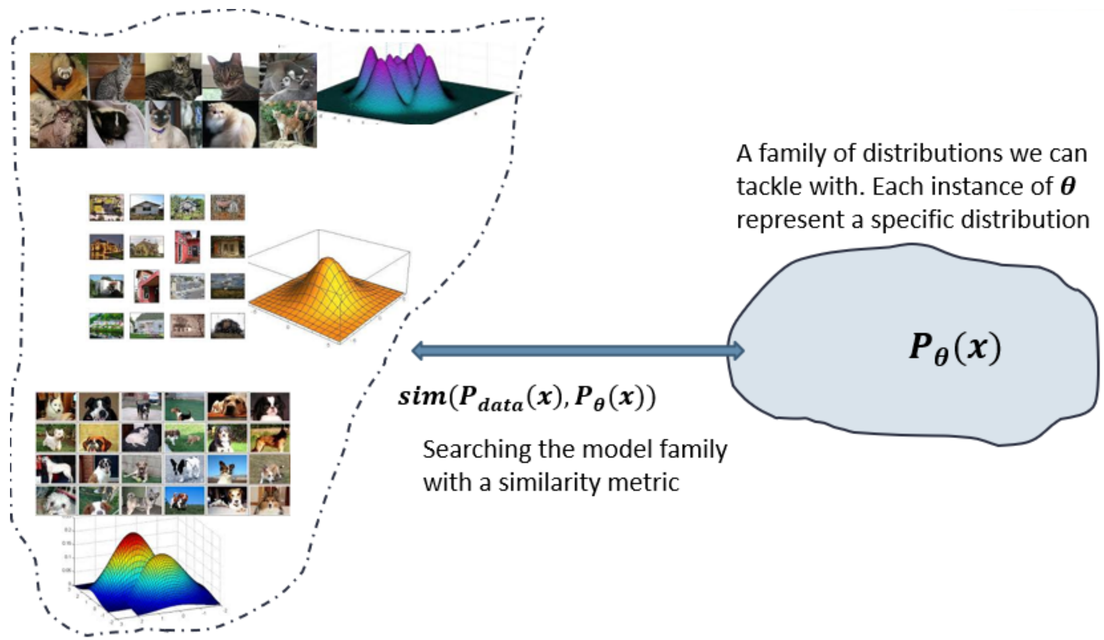
Probabilistic graphical models
Learning from data

22-808: Generative models
Sharif University of Technology
Fall 2025

Fatemeh Seyyedsalehi

Recap



- We need a framework to interact with distributions for statistical generative models.
  - Probabilistic generative models
    - Representation – Inference and Sampling – **Learning (today)**
  - Deep generative models

Learning in PGMs

- Let's assume that the real data is generated from a distribution p_{data}
  - A set of independent, identically distributed (i.i.d.) training samples, $\mathcal{D} = \{x^1, x^2, \dots, x^n\}$ is available.
  - Each sample is an assignment of values to (a subset of) the variables, e.g. pixel intensities.

- We are also given a family of models p_θ, and our task is to learn some "good" distribution in this set
  - For example, p_θ could be all Bayes nets with a given graph structure, for all possible choices of the CPDs

Learning in PGMs

- We want to learn the full distribution so that later we can answer any probabilistic inference query

- Learning in PGMs
  - Parameter learning ⟵
    - Learning parameters of potential functions and conditional probability distributions (CPDs)
  - Structure learning
    - For fixed nodes, learning edges!

Learning in PGMs
Parameter learning

- Given a set of i.i.d. training samples $\mathcal{D} = \{x^1, x^2, \dots, x^n\}$, the goal is learning parameters of factors, i.e. CPDs and potentials.
  - We assume that the structure of the graphical model is known.
  - Each sample $x^i = [x_1^i, x_2^i, \dots, x_m^i]$ is a vector of random variables in the graph.
    - **First, we assume data is completely observed**

- A parametric density estimation problem
  - p_θ is described in terms of a specific functional form which has a number of adjustable parameters

Learning in PGMs

- Density estimation techniques:
  - MLE: maximum likelihood estimation ⟵
  - Bayesian estimators: needs a prior distribution on parameters
    - Maximum a posteriori (MAP)
    - Full Bayesian estimator

Learning with MLE: maximum likelihood estimation

- The goal of learning is to return a model p_θ that precisely captures the distribution p_{data} from which our data was sampled .
- This is in general not achievable because of limited data only provides a rough approximation of the true underlying distribution.
- We want to select p_θ to construct the **best** approximation to the underlying distribution p_{data}.
- What is **best**?

Learning with MLE: maximum likelihood estimation

- Kullback-Leibler (KL) divergence to measure the distance between two distributions:

$$KL(p_{data} \parallel p_\theta) = \int p_{data} \log \frac{p_{data}}{p_\theta} dx$$

$$= \mathrm{E}_{p_{data}}[\log p_{data}] - \mathrm{E}_{p_{data}}[\log p_\theta]$$

- As the first term does not depend on p_θ, we have,

$$\underset{p_\theta}{\operatorname{argmin}} KL(p_{data} \parallel p_\theta) = \underset{p_\theta}{\operatorname{argmin}} -\mathrm{E}_{p_{data}}[\log p_\theta] = \underset{p_\theta}{\operatorname{argmax}} \mathrm{E}_{p_{data}}[\log p_\theta]$$

- p_θ should assign high probability to instances sampled from p_{data} to decrease the loss function.

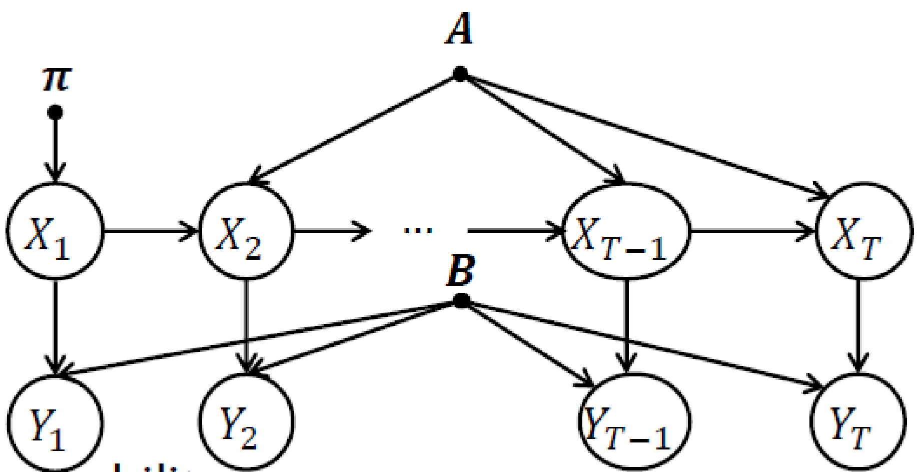
Learning with MLE: maximum likelihood estimation

- Monte Carlo Estimation
  - Approximate the expected log-likelihood

$$\mathrm{E}_{p_{data}}[\log p_\theta] = \int p_{data}(x) \log p_\theta(x)\ dx = \frac{1}{N}\sum_{i=1}^{N} \log p_\theta(x^i)$$

$$\underset{p_\theta}{\operatorname{argmax}}\, \mathrm{E}_{p_{data}}[\log p_\theta] = \underset{p_\theta}{\operatorname{argmax}} \frac{1}{N}\sum_{i=1}^{N} \log p_\theta(x^i)$$

Example
MLE for HMM – completely observed data



Initial state probability:

$$\pi_i = P(X_1 = i), \qquad 1 \leq i \leq K$$

State transition probability:

$$A_{ji} = P(X_{t+1} = i | X_t = j), \qquad 1 \leq i, j \leq K$$

State transition probability:

$$B_{ik} = P(Y_t = k | X_t = i), \qquad 1 \leq k \leq M$$

Example from Soleymani pgm-sharif

Example
MLE for HMM – completely observed data

$$P(\mathcal{D}|\boldsymbol{\theta}) = \prod_{n=1}^{N}\left[P\left(X_1^{(n)}\middle|\boldsymbol{\pi}\right)\prod_{t=2}^{T} P(X_t^{(n)}|X_{t-1}^{(n)},\boldsymbol{A})\prod_{t=1}^{T} P(Y_t^{(n)}|X_t^{(n)},\boldsymbol{B})\right]$$

$$\hat{A}_{ji} = \frac{\sum_{n=1}^{N}\sum_{t=2}^{T} I\left(X_{t-1}^{(n)} = j,\ X_t^{(n)} = i\right)}{\sum_{n=1}^{N}\sum_{t=2}^{T} I\left(X_{t-1}^{(n)} = j\right)}$$

$$\hat{\pi}_i = \frac{\sum_{n=1}^{N} I\left(X_1^{(n)} = i\right)}{N}$$

$$\hat{B}_{ik} = \frac{\sum_{n=1}^{N}\sum_{t=1}^{T} I\left(X_t^{(n)} = i,\ Y_t^{(n)} = k\right)}{\sum_{n=1}^{N}\sum_{t=1}^{T} I\left(X_t^{(n)} = i\right)}$$

Discrete observations

Example from Soleymani pgm-sharif

Learning from Incomplete data

- Now, we assume data is not completely observed
- Given a set of i.i.d. training samples $\mathcal{D} = \{x^1, x^2, \dots, x^n\}$, the goal is learning parameters of factors (CPDs and potentials).
  - We assume that the structure of the graphical model is known.
  - Each sample $x^i = [x_O^i, x_H^i]$ is a vector that **some of its elements are latent/hidden/unknown.**
  - We assume a specific set of random variables are latent in all samples

Learning from Incomplete data

- Complete likelihood
  - Maximizing likelihood $p_\theta(\mathcal{D};\boldsymbol{\theta})$ for labeled data is straightforward

- Incomplete likelihood
  - Our objective becomes

$$p_\theta(\mathcal{D};\boldsymbol{\theta}) = p_\theta(x_O;\boldsymbol{\theta}) = \sum_{\mathcal{H}} p(x_O, x_{\mathcal{H}};\boldsymbol{\theta})$$

- Incomplete likelihood is the sum of likelihood functions, one for each possible joint assignment of the missing values.
- The number of possible assignments is exponential in the total number of latent variables.

EM algorithm

- General algorithm for finding MLE when data is incomplete (missing or unobserved data).

- An iterative algorithm in which each iteration is guaranteed to improve the log-likelihood function

- When hidden data, $\mathcal{H}$ is relevant to observed data $\mathcal{D}$ (in any way), we can hope to extract information about it from $\mathcal{D}$ assuming a specific parametric model on the data.

Expectation-maximization (EM) method

X: observed variables

Z: unobserved variables

$\boldsymbol{\theta}$: parameters

Expectation step (E-step): Given the current parameters, find soft completion of data using probabilistic inference

Maximization step (M-step): Treat the soft completed data as if it were observed and learn a new set of parameters

Choose an initial setting $\boldsymbol{\theta}^0, t = 0$

Iterate until convergence:

E Step: Use X and current $\boldsymbol{\theta}^t$ to calculate $P(Z|X, \boldsymbol{\theta}^t)$

M Step: $\boldsymbol{\theta}^{t+1} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} E_{Z \sim P(Z|X, \boldsymbol{\theta}^t)}[\log p(X, Z|\boldsymbol{\theta})]$

$t \leftarrow t + 1$

expectation of the log-likelihood evaluated using the current estimate for the parameters $\boldsymbol{\theta}^t$

$$E_{Z \sim P(Z|X, \boldsymbol{\theta}^{\text{old}})}[\log p(X, Z|\boldsymbol{\theta})]$$

$$= \sum_Z P(Z|X, \boldsymbol{\theta}^{\text{old}}) \times \log p(X, Z|\boldsymbol{\theta})$$

EM theoretical foundation

- Remember this equation from the last lecture

$$KL(q(Z) \parallel p(Z|X)) = KL\big(q(Z) \parallel p(Z,X)\big) + \log p(X)$$

- We have:

$$KL(q(Z) \parallel p(Z|X)) \geq 0 \rightarrow \log p(X) \geq -KL\big(q(Z) \parallel p(Z,X)\big)$$

$$\rightarrow \boldsymbol{q(Z) = p(Z|X) \rightarrow \ log\, p(x) = -KL\big(q(Z) \parallel p(Z,X)\big)}$$

- In **E-step** we set $q(Z)$ equal to $p(Z|X)$, therefore in the M-step we can maximize $-KL\big(q(Z) \parallel p(Z,X)\big)$ instead of $\log p(X)$:

$$\underset{\theta}{\operatorname{argmax}} \log p(x;\theta) \ = \underset{\theta}{\operatorname{argmax}} \mathrm{E}_{p(Z|X)}[p(Z|X)] - \mathrm{E}_{p(Z|X)}[p(Z,X;\theta)]$$

- The first term is fixed in the E-step and int the **M-step** is independent of θ, therefore in the maximization step we only maximize the second term:

$$\boldsymbol{\underset{\theta}{argmax} \ - E_{p(Z|X)}[p(Z,X;\theta)]}$$