Statistical Methods in Image Processing EE-048954

Homework 3: Contrastive Divergence and Noise Contrastive Estimation

Due Date: June 16, 2022

Submission Guidelines

- Submission only in **pairs** on the course website (Moodle).
- Working environment:
  - We encourage you to work in `Jupyter Notebook` online using Google Colab as it does not require any installation.
- You should submit two **separated** files:
  - A `.ipynb` file, with the name: `ee048954_hw3_id1_id2.ipynb` which contains your code implementations.
  - A `.pdf` file, with the name: `ee048954_hw3_id1_id2.pdf` which is your report containing plots, answers, and discussions.
  - **No handwritten submissions** and no other file-types (`.docx` , `.html` , ...) will be accepted.

Mounting your drive for saving/loading stuff

In [ ]:
```
from google.colab import drive
drive.mount('/content/drive')
```

Importing relevant libraries

In [6]:
```
## Standard libraries
import os
import math
import time
import numpy as np
import random
import copy

## Scipy optimization routines
from scipy.optimize import minimize

## Progress bar
import tqdm

## Imports for plotting
import matplotlib.pyplot as plt
import matplotlib.animation as animation
%matplotlib inline
```

```
import matplotlib
matplotlib.rcParams['lines.linewidth'] = 2.0
plt.style.use('ggplot')
```

Part I: Contrastive Divergence (50 points)

Consider the following Gaussian Mixture Model (GMM) distribution

$$p(x;\{\mu_i\}) = \sum_{i=1}^{N} \frac{1}{N} \frac{1}{2\pi} \exp\left\{-\frac{1}{2}||x-\mu_i||^2\right\},$$

where $x, \mu_i \in \mathbb{R}^2$. We will use $N = 4$, $\sigma = 1$, and $\{\mu_i\} = \{(0,0)^T, (0,3)^T, (3,0)^T, (3,3)^T\}$.

Sampling from GMM

Task 1. Direct sampling: Use your function from HW1 that accepts $\{\mu_i\}$, and returns a sample x from $p(x;\{\mu_i\})$. Draw $J = 1000$ samples $\{x\}$ from the distribution $p(x;\{\mu_i\})$ using this function. These will be our **real samples**.

Task 2. Sampling with MCMC: implement the MALA algorithm to draw samples from $p(x;\{\mu_i\})$. The function will get an initial guess $\{\hat{x}_i\}$ and will generate chains of length L. Use $\sqrt{2\varepsilon} = 0.1$ and $N \sim \mathcal{N}(0, I)$.

From now on, we will refer to $\{\mu_i\}$ **as unknowns** and we will estimate them using different algorithms.

Estimation of $\{\mu_i\}$

Task 3. Implement Maximum likelihood (ML) estimation of $\{\mu_i\}$ using direct sampling:

- Step 1: Randomly initialize $\{\tilde{\mu}_i\}$ from $U([0,3]^2)$.
- Step 2: Use your function from Task 1 to draw 100 samples $\tilde{x}$ from $p(x;\{\mu_i\})$ using $\{\tilde{\mu}_i\}$.
- Step 3: Update $\{\tilde{\mu}_i\}$ using the ML gradient descent step:

  $$\tilde{\mu}_i^{k+1} = \tilde{\mu}_i^k + \eta \left(\langle \nabla_{\mu_i} \log p(x;\{\mu_i\}) \rangle_x - \langle \nabla_{\mu_i} \log p(x;\{\mu_i\}) \rangle_{\tilde{x}} \right),$$

  where $\langle \cdot \rangle_x$ denotes averaging over the real samples from Section A and $\langle \cdot \rangle_{\tilde{x}}$ denotes averaging over the synthetically generated samples from Step 2. Use $\eta = 1$.
- Repeat Step 2 and Step 3 until convergence.

Task 4. Implement Maximum likelihood (ML) estimation of $\{\mu_i\}$ using MCMC:

- Step 1: Randomly initialize $\{\tilde{\mu}_i\}$ from $U([0,3]^2)$.
- Step 2: Use your function from Task 2 to draw 100 samples $\tilde{x}$ from $p(x;\{\mu_i\})$ using $\{\tilde{\mu}_i\}$. Initialize the chains with $\hat{x}_i \sim \mathcal{N}(1.5, 2)$ and use chains length of $L = 1000$.
- Step 3: Update $\{\tilde{\mu}_i\}$ using the ML gradient descent step:

$$\tilde{\mu}_i^{k+1} = \tilde{\mu}_i^k + \eta \left(\langle \nabla_{\mu_i} \log p(x; \{\mu_i\}) \rangle_x - \langle \nabla_{\mu_i} \log p(x; \{\mu_i\}) \rangle_{\tilde{x}} \right),$$

where $\langle \cdot \rangle_x$ denotes averaging over the real samples from Section A and $\langle \cdot \rangle_{\tilde{x}}$ denotes averaging over the synthetically generated samples from Step 2. Use $\eta = 1$.

- Repeat Step 2 and Step 3 until convergence.

 **Task 5**. Implement Contrastive Divergence (CD) estimation of $\{\mu_i\}$ using MCMC sampling:

- Step 1: Randomly initialize $\{\tilde{\mu}_i\}$ from $U([0, 3]^2)$.
- Step 2: Use your function from Task 2 to draw 100 samples $\tilde{x}$ from $p(x; \{\mu_i\})$ using $\{\tilde{\mu}_i\}$. Initialize the chains with **100 samples randomly chosen from the real set of examples from Task 1**, and use only $L = 10$ update steps.
- Step 3: Update $\{\tilde{\mu}_i\}$ using the CD gradient descent step:

$$\tilde{\mu}_i^{k+1} = \tilde{\mu}_i^k + \eta \left(\langle \nabla_{\mu_i} \log p(x; \{\mu_i\}) \rangle_x - \langle \nabla_{\mu_i} \log p(x; \{\mu_i\}) \rangle_{\tilde{x}} \right),$$

where $\langle \cdot \rangle_x$ denotes averaging over the 100 real samples used for initialization of the chains in Step 2 and $\langle \cdot \rangle_{\tilde{x}}$ denotes averaging over the MCMC generated samples from Step 3. Use $\eta = 1$.

- Repeat Step 2 and Step 3 until convergence.

 **Task 6**. Present the estimated $\{\mu_i\}$ and the final random samples $\{\tilde{x}_i\}$ generated with each of the three algorithms in Tasks 3-5. Discuss the differences in convergence.

Part II: Noise Contrastive Estimation (50 points)

Consider the distribution

$$p_m(x; \{\mu_i\}) = \frac{1}{Z} \sum_{i=1}^{N} \exp\left\{ -\frac{1}{2\sigma^2} ||x - \mu_i||^2 \right\},$$

where $Z \in \mathbb{R}$ is a normalization constant, and $x, \mu_i \in \mathbb{R}^2$.

Sampling from GMM

 **Task 7**. What is the value of Z?

 **Task 8**. Use $N = 4$, $\sigma = 1$, and $\{\mu_i\} = \{(0, 0)^T, (0, 3)^T, (3, 0)^T, (3, 3)^T\}$. Draw $J = 1000$ samples $\{x_j\}$ from the distribution $p_m(x; \{\mu_i\})$ using the function from Task 1.

From now on, we will refer to $\{\mu_i\}$ **as unknowns** and we will estimate them using the Noise Contrastive Estimation method.

Estimation of $\{\mu_i\}$

 **Task 9**. Implement Noise Contrastive Estimation of $\{\mu_i\}$:

- Step 1: Generating the artificial data-set of noise: Draw $J = 1000$ samples $\{y_j\}$ from

$$p_n(y;\mu_n) = \frac{1}{2\pi\sigma_n^2}\exp\left\{-\frac{1}{2\sigma_n^2}||y-\mu_n||^2\right\}$$

  using $\mu_n = (1,1)^T$ and $\sigma_n = 2$.
- Step 2: Randomly select an initial guess for the model means $\{\tilde{\mu}_i\}$ from $U([0,3]^2)$.
- Step 3: Update $\{\tilde{\mu}_i\}$ by **maximizing**:

$$\{\tilde{\mu}_i\} = \arg\max_{\{\mu_i\}} \sum_{j=1}^{J} \left[\ln(h(x_j;\{\mu_i\})) + \ln(1-h(y_j;\{\mu_i\}))\right],$$

  where

$$h(u;\{\mu_i\}) = \frac{p_m(u;\{\mu_i\})}{p_m(u;\{\mu_i\}) + p_n(u;\mu_n)}.$$

  Implementation Tip: This step can be executed using the function `scipy.optimize.minimize` which finds the **minimum** of an (unconstrained) optimization problem (e.g. using the `'BFGS'` method), given a function that calculates the objective and an initial guess (see scipy documentation for more details). In our case, for **maximization**, implement a function that calculates the **minus** of the objective above.

We will now regard both $\{\mu_i\}$ **and the normalization constant** Z as unknowns, and will estimate them using Noise Contrastive Estimation.

Task 10. Implement Noise Contrastive Estimation with an **un-normalized** probability model:

- Step 1: Generating the artificial data-set of noise: Draw $J = 1000$ samples $\{y_j\}$ from

$$p_n(y;\mu_n) = \frac{1}{2\pi\sigma_n^2}\exp\left\{-\frac{1}{2\sigma_n^2}||y-\mu_n||^2\right\}$$

  using $\mu_n = (1,1)^T$ and $\sigma_n = 2$.
- Step 2: Randomly select an initial guess for the model means $\{\tilde{\mu}_i\}$ from $U([0,3]^2)$, and for the normalization constant Z from $U([0.1,1])$
- Step 3: Update $\{\tilde{\mu}_i\}$ and Z by **maximizing**:

$$\{\tilde{\mu}_i\}, Z = \arg\max_{\{\mu_i\},Z} \sum_{j=1}^{J} \left[\ln(h(x_j;\{\mu_i\},Z)) + \ln(1-h(y_j;\{\mu_i\},Z))\right],$$

  where

$$h(u;\{\mu_i\},Z) = \frac{p_m(u;\{\mu_i\},Z)}{p_m(u;\{\mu_i\},Z) + p_n(u;\mu_n)}.$$

Evaluating the Results

Task 11. Visually: plot the estimates of $\{\mu_i\}$ of Tasks 9 and 10 (two separate figures). Include

the model samples, the noise samples, the initial guess for the model means, and the final estimates of $\{\tilde{\mu}_i\}$.

 **Task 12**. Quantitatively: repeat Tasks 9 and 10, this time with $J = 100 \times [1, 5, 10, 20, 30, 50]$. For each value of J repeat the estimation process for 50 times, each time with different realizations for $\{x_j\}$ and $\{y_j\}$ and initial guesses for the estimands ($\{\mu_i\}$ in Task 9 and $\{\mu_i\}, Z$ in Task 10.

For each value of J, calculate the MSE between the true parameter values and their estimates (the mean will be taken over the different realizations). Note that for the model means, the MSE should be calculated to the closest true μ_i for each estimation. If at the same run two estimated μ_is pick the same true μ_i, then this run should be declared as a failure and should be disregarded. Report the number of failure runs.

 **Task 13**. Discussion: How does the number of samples J affect the accuracy of the estimation? How does the addition of Z as an unknown affect the accuracy?