Mathematical Foundations of Deep Neural Networks, M1407.001200
E. Ryu
Fall 2021

Homework 8
Due 5pm, Tuesday, November 30, 2021

Problem 1: *1D flow to Gaussian.* Consider the flow

$$f_\theta(x) = \sum_{i=1}^{n} e^{w_i}(\Phi_{\mu_i,\exp(\tau_i)}(x) - 0.5),$$

where $\theta = (w_1, \ldots, w_n, \mu_1, \ldots, \mu_n, \tau_1, \ldots, \tau_n)$ and

$$\Phi_{\mu,\sigma}(x) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{x} \exp\left(-\frac{1}{2}\left(\frac{s-\mu}{\sigma}\right)^2\right) ds.$$

Note that $f_\theta \colon \mathbb{R} \to \mathbb{R}$. Download the starter code `normalizingFlow1d.py` and fit the flow model with $n = 5$ and $p_Z \sim \mathcal{N}(0,1)$.

Remark. Since p_Z is an unbounded distribution, we do not require $w_1, \ldots, w_n$ to be normalized.

Problem 2: Consider the optimization problem

$$\underset{\theta\in\Theta}{\text{maximize}} \quad f(\theta).$$

Informally assume f is an intractable function, i.e., evaluating $f(\theta)$ is difficult. However, assume there exists a decomposition

$$f(\theta) = g(\theta, \phi) + h(\theta, \phi) \qquad \forall\, \phi \in \Phi,$$

where g is tractable, i.e., evaluating $g(\theta, \phi)$ is easy, $h(\theta, \phi) \geq 0$ for all $\theta \in \Theta$ and $\phi \in \Phi$, and for any $\theta \in \Theta$ there exists a $\phi \in \Phi$ such that $h(\theta, \phi) = 0$, i.e., for any $\theta \in \Theta$,

$$\min_{\phi\in\Phi} h(\theta, \phi) = 0$$

and the minimum is attained. Now we consider the following problem with the tractable objective function

$$\underset{\theta\in\Theta,\,\phi\in\Phi}{\text{maximize}} \quad g(\theta, \phi).$$

Show that the two optimization problems are equivalent in the sense that

$$\operatorname{argmax} f = \{\theta \,|\, (\theta, \phi) \in \operatorname{argmax} g\}$$

Hint. Use the fact that

$$\sup_{\theta,\phi} g(\theta, \phi) = \sup_{\theta}\left(\sup_{\phi} g(\theta, \phi)\right).$$

Remark. Training variational autoencoders involves maximizing the variational lower bound (VLB/ELBO). If the encoder network is infinitely expressive (if the encoder network can represent any function), maximizing the VLB is equivalent to maximizing the log-likelihood. This problem abstracts the explanation of why that is the case.

Problem 3: *Inverse permutation.* Let S_n denote the group of length n permutations. Note that the map $i \mapsto \sigma(i)$ is a bijection. Define $\sigma^{-1} \in S_n$ as the permutation representing the inverse of this map, i.e, $\sigma^{-1}(\sigma(i)) = i$ for $i = 1, \ldots, n$. Describe an algorithm for computing σ^{-1} given σ.

Clarification. In this class, we defined σ as a list of length n containing the elements of $\{1, \ldots, n\}$ exactly once. The output of the algorithm, σ^{-1}, should also be provided as a list.

Clarification. For this problem, it is sufficient to describe the algorithm in equations or pseudocode. There is no need to submit a Python script for this problem.

Problem 4: *Permutation matrix.* Given a permutation $\sigma \in S_n$, the *permutation matrix* of σ is defined as

$$P_\sigma = \begin{bmatrix} e_{\sigma(1)}^\intercal \\ e_{\sigma(2)}^\intercal \\ \vdots \\ e_{\sigma(n)}^\intercal \end{bmatrix} \in \mathbb{R}^{n \times n},$$

where $e_1, \ldots, e_n \in \mathbb{R}^n$ are the standard unit vectors. Show

(a) $P_\sigma^\intercal = P_\sigma^{-1} = P_{\sigma^{-1}}$ and

(b) $|\det P_\sigma| = 1$.

HInt. If the rows of $U \in \mathbb{R}^{n \times n}$ are orthonormal, we say U is an orthogonal matrix. Orthogonal matrices satisfy $UU^\intercal = U^\intercal U = I$.

Problem 5: *Affine coupling layer with permutations.* Consider the affine coupling layer defined as follows. Let $\Omega \subseteq \{1, \ldots, n\}$ and $0 < |\Omega| < n$. Define $\Omega^\complement = \{1, \ldots, n\} \backslash \Omega$. For $x \in \mathbb{R}^n$, define

$$x_\Omega \in \mathbb{R}^{|\Omega|}, \qquad x_{\Omega^\complement} \in \mathbb{R}^{n-|\Omega|}$$

to be the sub-vectors of x with the indices within Ω and $\Omega^\complement$ selected. Define z_Ω and $z_{\Omega^\complement}$ analogously for $z \in \mathbb{R}^n$. The affine coupling layer is

$$\begin{aligned} z_\Omega &= x_\Omega \\ z_{\Omega^\complement} &= e^{s_\theta(x_\Omega)} \odot x_{\Omega^\complement} + t_\theta(x_\Omega), \end{aligned}$$

where $s_\theta \colon \mathbb{R}^{|\Omega|} \to \mathbb{R}^{n-|\Omega|}$ and $t_\theta \colon \mathbb{R}^{|\Omega|} \to \mathbb{R}^{n-|\Omega|}$. Show that

$$\log \left| \frac{\partial z}{\partial x} \right| = \mathbf{1}_{n-|\Omega|}^\intercal s_\theta(x_\Omega).$$

Clarification. We are not assuming $|\Omega| = n/2$.

Hint. Find a permutation σ such that

$$\frac{\partial z}{\partial x} = P_{\sigma^{-1}} \begin{bmatrix} I & 0 \\ * & \operatorname{diag}(e^{s_\theta(x_\Omega)}) \end{bmatrix} P_\sigma.$$

Problem 6: *Gambler's ruin.* You are a gambler at a casino with a starting balance of 100\$. You will play a game in which you bet 1\$ every game. With probability 18/37, you win and collect 2\$ (so you make a 1\$ profit). With probability 19/37, you lose and collect no money. You play until you reach a balance of 0\$ or 200\$ or until you play 600 games. Write a Monte Carlo simulation with importance sampling to estimate the probability that you leave the casino with 200\$. Specifically, simulate playing up to 600 games until you reach the balance of 0\$ or 200\$ and repeat this $N = 3000$ times.

Hint. Regardless of the outcome, simulate $K = 600$ games. The outcomes of the games form a sequence of Bernoulli random variables with probability mass function

$$f(X_1, \ldots, X_K) = \prod_{i=1}^{K} p^{X_i}(1-p)^{(1-X_i)}$$

and $p = 18/37$. For the sampling distribution, also use a sequence of Bernoulli random variables with probability mass function

$$g(Y_1, \ldots, Y_K) = \prod_{i=1}^{K} q^{Y_i}(1-q)^{(1-Y_i)}$$

but with $q > p$. Try using $q = 0.55$.

Hint. The answer is approximately 2×10^{-6}. Submit Python code that produces this answer.

Problem 7: Solve

$$\begin{array}{ll} \underset{\mu,\sigma\in\mathbb{R}}{\text{minimize}} & \mathbb{E}_{X\sim\mathcal{N}(\mu,\sigma^2)}[X\sin(X)] + \frac{1}{2}(\mu-1)^2 + \sigma - \log\sigma \\ \text{subject to} & \sigma > 0 \end{array}$$

using SGD combined with

(a) the log-derivative trick and

(b) the reparameterization trick.

Hint. Use the change of variables $\sigma = e^{\tau}$ to remove the constraint $\sigma > 0$.

Clarification. Implement SGD in Python and submit the code.

Problem 8: *Log-derivative trick for VAE.* Let $Z \in \mathbb{R}^k$ be a random variable. Let $q_\phi(z)$ be a probability density function for all $\phi \in \mathbb{R}^p$. Assume $q_\phi(z)$ is differentiable in ϕ for all fixed $z \in \mathbb{R}^k$. Let $h\colon \mathbb{R}^k \to \mathbb{R}$ satisfy $h(z) > 0$ for all $z \in \mathbb{R}^k$. Assume that the order of integration and differentiation can be swapped. Show

$$\nabla_\phi \mathbb{E}_{Z\sim q_\phi(z)}\left[\log\left(\frac{h(Z)}{q_\phi(Z)}\right)\right] = \mathbb{E}_{Z\sim q_\phi(z)}\left[(\nabla_\phi \log q_\phi(Z))\log\left(\frac{h(Z)}{q_\phi(Z)}\right)\right].$$

Hint. Since $q_\phi(z)$ is a probability density function,

$$\int \nabla_\phi q_\phi(z)\, dz = \nabla_\phi \int q_\phi(z)\, dz = \nabla_\phi 1 = 0.$$