6.790 Homework 7

This homework is for study purposes and will not be handed in or graded.

Contents

1 Generative model warmup **2**

2 VAE **2**

3 Mixture models **4**

4 Diffusion models **6**

5 Flow Matching **7**

1 Generative model warmup

1. (a) We are considering a variety of different generative models, trained on some dataset $\mathcal{D} = \{x^{(i)}\}_{x=1}^{n}$.
   For each of the following models, describe what parametric models (including neural networks) are learned and how we use them (and/or the training data) to obtain a value for the density $p(x)$, for a novel input x (or explain that it's difficult/impossible).
   i. kernel density estimator
   ii. auto-regressive model
   iii. Gaussian mixture
   iv. continuous-time probability flow (out of our scope, feel free to skip)
   v. variational auto-encoder
   vi. diffusion model (out of our scope, feel free to skip)

   (b) Now, for each of these same models, describe how to draw iid samples from the learned density.
   i. kernel density estimator
   ii. auto-regressive model
   iii. Gaussian mixture
   iv. continuous time probability flow
   v. variational auto-encoder
   vi. diffusion model

2. We can interpret logistic regression as determining a conditional distribution $\hat{p}(Y \mid X)$.
   (a) Explain how to sample from $\hat{p}(Y \mid X = x)$ for a novel value x.
   (b) Let's explore the ability of logistic regression to represent posterior distributions. Imagine a training set $\mathcal{D} = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n}$ where $x^{(i)} = (1)$ for all i, and where $y^{(i)} = +1$ for proportion p of the data (and 0 for proportion $1 - p$).
   Show that in the limit of large n, the learned likelihood $\hat{p}(Y = 1 \mid X = (1))$ converges to p.

2 VAE

This question closely follows the development in Kingma, Diederik P., and Max Welling. "An introduction to variational autoencoders." Foundations and Trends in Machine Learning.

3. (a) If our goal is to construct a model of a data distribution $\hat{p}_\theta(x)$, what advantage is there in turning it into an apparently harder problem of modeling $\hat{p}_\theta(x, z)$ for some latent variable z?
   (b) Consider a distribution over $x \in \mathbb{R}$ such that $p(x \leqslant v) = F(v)$ for some cdf F.
   i. How is $F(x)$ distributed if $x \sim p(x)$?

ii. If we wanted to make a latent variable model with $p(z) = \text{Unif}(0,1)$, which of the following choices for $p(x \mid z)$ would guarantee that $p(x) = \int_z p(x \mid z)p(z)dz$?
- ◯ A distribution that assigns probabilty 1 to $x = F^{-1}(z)$.
- ◯ A distribution that assigns probabilty 1 to $x = F^{-1}(p(z))$.
- ◯ A Gaussian $\mathcal{N}(F(z), 1)$.

iii. If we wanted to make a latent variable model with $p(z) = \mathcal{N}(0,1)(z)$, which of the following choices for $p(x \mid z)$ would guarantee that $p(x) = \int_z p(x \mid z)p(z)dz$?
- ◯ A distribution that assigns probabilty 1 to $x = F^{-1}(z)$.
- ◯ A distribution that assigns probabilty 1 to $x = F^{-1}(p(z))$.
- ◯ A distribution that assigns probabilty 1 to $x = F^{-1}(G(z))$, where G is the Gaussian cdf.
- ◯ A Gaussian $\mathcal{N}(F(z), 1)$.

iv. Now, let's say we want to train a neural network with parameters θ to represent $p(x \mid z)$, by maximizing the log likelihood of some training set $\mathcal{D} = \{x^{(i)}\}_{i=1}^n$. What loss function would we minimize, ignoring (for now) computational intractability?

v. Why is it hard to minimize, especially in high dimensions?

vi. Provide an approximation to the loss function, based on sampling.

vii. What problems might we have with this estimator if we sample $z \sim \mathcal{N}(0, I)$ in high dimensions?

(c) The strategy in a VAE is to learn a new distribution $q_\phi(z \mid x)$, called the *inference model* that will hopefully generate samples that will tend to have high values of $p_\theta(x^{(i)} \mid z)$. Let's focus on a single data-point x.

i. We observe that
$$\log p_\theta(x) = \log \frac{p_\theta(x, z)}{p_\theta(z \mid x)}$$
Verify this.

ii. If we are going to sample using the inference model, then it's useful to view this as
$$\log p_\theta(x) = \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p_\theta(x, z)}{p_\theta(z \mid x)}\right]$$
which we can (apparently gratuitously) rewrite as
$$\log p_\theta(x) = \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p_\theta(x, z)q_\phi(z \mid x)}{p_\theta(z \mid x)q_\phi(z \mid x)}\right]$$
But this lets us divide into two terms that are useful:
$$\log p_\theta(x) = \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{p_\theta(x, z)}{q_\phi(z \mid x)}\right] + \mathbb{E}_{q_\phi(z|x)}\left[\log \frac{q_\phi(z \mid x)}{p_\theta(z \mid x)}\right]$$
and they have names!
$$\log p_\theta(x) = \text{ELBO}_{\theta,\phi}(x) + \text{KL}\big(q_\phi(z \mid x) \,\|\, p_\theta(z \mid x)\big)$$
We are going to work on maximizing the ELBO rather than $\log p_\theta(x)$. Why is that more straightforward?

iii. Show that $\text{ELBO}_{\theta,\phi}(x) \leqslant \log p_\theta(x)$. When are they equal?

iv. Write an expression for the ELBO in terms of the data likelihood and $\text{KL}\big(q_\phi(z \mid x) \parallel p_\theta(z \mid x)\big)$. Assuming that our neural networks have infinite representational capacity and the optimization works perfectly, if we optimize $\text{ELBO}_{\theta,\phi}(x)$, what can we say about $p_\theta(x)$?

(d) It's time to maximize the ELBO via stochastic gradient descent!

i. First, with respect to θ. If we represent $p_\theta(x, z) = p_\theta(x \mid z)p(z)$ where $p(z)$ is a fixed spherical Gaussian, and $p_\theta(x \mid z) = \mathcal{N}(\text{NN}_\theta(z), \sigma^2)$ where $\text{NN}_\theta(z)$ is a deterministic neural network parameterized by θ and σ is a small fixed standard deviation, write an expression for

$$\nabla_\theta \text{ELBO}_{\theta,\phi}(x)$$

ii. Now, with respect to ϕ. This is harder because ϕ appears in the distribution that we're taking the expectation over:

$$\text{ELBO}_{\theta,\phi}(x) = \mathbb{E}_{q_\phi(z|x)}\big[\log p_\theta(x, z) - \log q_\phi(z \mid x)\big]$$

So we can't just push the gradient inside the expectation, tempting though it may be. Instead, we need to do the *reparameterization trick!* (See Murphy book 2 section 6.3.5) Instead of taking the expectation with respect to a distribution q over z, we'll define a new random variable $\epsilon \sim \mathcal{N}(0, 1)$ and define $z = g(\epsilon, \phi, x)$. Now,

$$\begin{aligned}\nabla_\phi \text{ELBO}_{\theta,\phi}(x) &= \nabla_\phi \mathbb{E}_\epsilon\big[\log p_\theta(x, z) - \log q_\phi(z \mid x)\big] \\ &= \mathbb{E}_\epsilon \nabla_\phi\big[\log p_\theta(x, z) - \log q_\phi(z \mid x)\big] \\ &\approx \nabla_\phi\big[\log p_\theta(x, z) - \log q_\phi(g(\epsilon, \phi, x) \mid x)\big]\end{aligned}$$

Write an expression for this gradient, assuming we represent $q_\phi(z \mid x) = \mathcal{N}(\text{NN}_\phi(x), \sigma^2)$ where $\text{NN}_\phi(x)$ is deterministic a neural network parameterized by ϕ and σ is a small fixed standard deviation. It will depend on g. (We won't go into strategies for choosing g, but the paper describes it nicely.)

3 Mixture models

4. Consider a simple mixture model involving two spherical Gaussians in two dimensions. So $x \in \mathcal{R}^2$ and

$$P(x|\theta) = \sum_{z=1}^{2} P(z|\theta)P(x|z, \theta) = \sum_{z=1}^{2} p_z \mathcal{N}(x; \mu_z, \sigma_z^2 I)$$

We will initialize the parameters of this mixture model as follows

$$p_1 = p_2 = 0.5, \quad \mu_1 = \mu_2, \quad \sigma_2^2 = 2\sigma_1^2$$

The initialization is also shown graphically in Figure 1 (top middle). The circles are drawn exactly one standard deviation (e.g., σ_1) away from the corresponding mean (e.g., μ_1). The larger dashed circle corresponds to the second component with larger variance.

Given the initialization above, which one of the figures a-d) of Figure 1 represents the mixture model that we get after one EM-iteration? Briefly justify your answer.

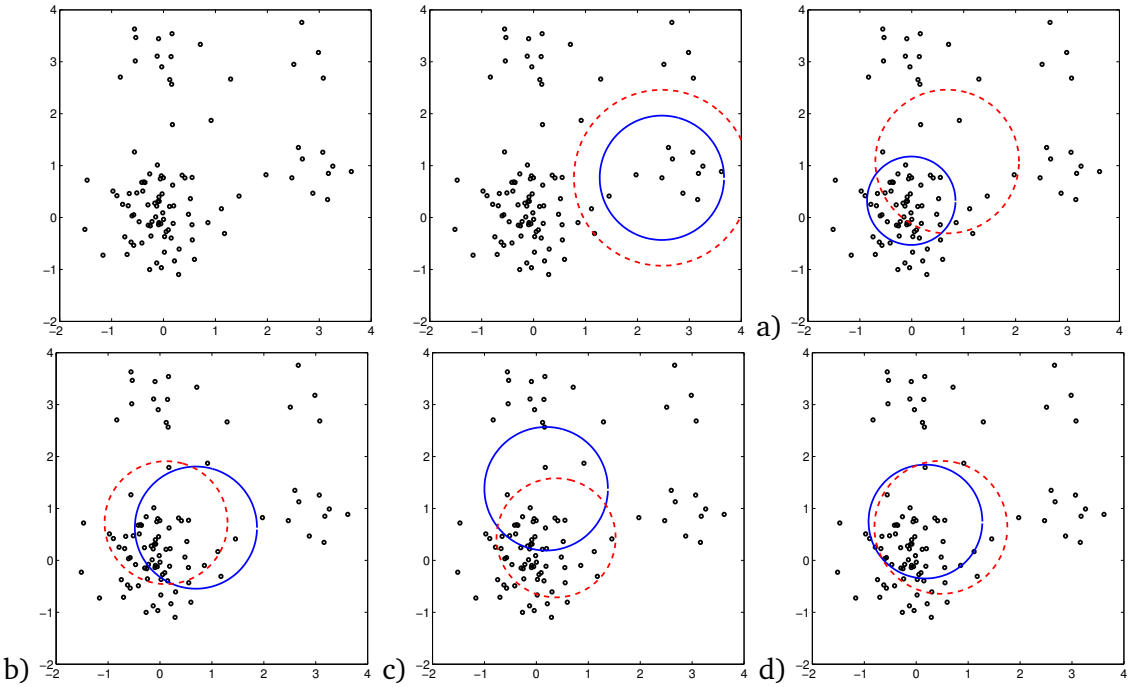

Figure 1: Top left) observed data; top right) initial mixture model; a-d) candidate mixture models resulting from one EM update.

5. Suppose we have a k-component mixture of spherical Gaussians model. Which of the following initializations of mixing proportions, means, and variances have a chance of recovering the underlying clusters assuming our assumption about the model family is correct? If the initialization is likely to fail, describe how. We use a shorthand $[k] = \{1, \ldots, k\}$.

    1. $p_j = 1/k$, $j \in [k]$; $\mu_j = \mu_0$, $j \in [k]$, for some common μ_0; $\sigma_j = \sigma_0$, $j \in [k]$, for some common σ_0.
    2. $p_j = 1/k$, $j \in [k]$; $\mu_j = \mu_0$, $j \in [k]$, for some common μ_0; σ_j, $j \in [k]$, are set to different values
    3. $p_j = 1/k$, $j \in [k]$; μ_j, $j \in [k]$, are set to randomly chosen data points; $\sigma_j = \sigma_0$, $j \in [k]$, for some common σ_0.
    4. p_j, $j \in [k]$ are randomized; $\mu_j = \mu_0$, $j \in [k]$, for some common μ_0; $\sigma_j = \sigma_0$, $j \in [k]$, for some common σ_0.

6. In this question and the next we revisit the ELBO introduced in Section 2, and show how it can be also used to view the EM algorithm for estimating a mixture of k spherical Gaussians model.

    Let $D = \{x^i\}_{i=1,\ldots,n}$ be our observed data where $x^i \in \mathbb{R}^d$. Given any choice of distributions $Q(z|x^i)$, provide explicit parameter estimates of the mixture model as a function of these

choices (M-step). In other words, solve $\hat{\theta} = \arg\max_\theta \text{ELBO}(Q;\theta)$ where

$$\text{ELBO}(Q;\theta) = \sum_{i=1}^{n} \left\{ \sum_{z=1}^{k} Q(z|x^i) \log \left[p_z N(x^i; \mu_z, \sigma_z^2 I) \right] + H(Q_{z|x^i}) \right\}$$

7. Recall that we can equivalently write the ELBO estimation criterion as

$$\text{ELBO}(Q;\theta) = \sum_{i=1}^{n} \left\{ \log P(x^i|\theta) - \text{KL}(Q_{z|x^i} \| P_{z|x^i,\theta}) \right\}$$

Show that when $Q(z|x^i) = P(z|x^i, \theta_0)$ for all $z \in [k]$ and $i = 1, \ldots, n$, then

$$\nabla_\theta \text{ KL}(Q_{z|x^i} \| P_{z|x^i,\theta})\big|_{\theta=\theta_0} = 0 \text{ (vector)}$$

for all $i = 1, \ldots, n$. This result ensures that $\nabla_\theta \text{ELBO}(Q;\theta)|_{\theta=\theta_0} = \nabla_\theta \sum_{i=1}^{n} \log P(x^i|\theta)|_{\theta=\theta_0}$ after each E-step. In other words, the lower bound criterion not only agrees in value at $\theta = \theta_0$ but it also has the same derivative as the log-likelihood.

4 Diffusion models

8. Let's consider a simple diffusion model in 2D. In other words, we are generating samples $x \in \mathbb{R}^2$. The dataset available to us consists of only two points, $[1,0]^T$ and $[0,1]^T$.

   (a) Let β_t, $t = 1, 2, \ldots, T$ refer to the noise variance we add at step t. In other words, at step t in the forward process we update the example according to $x_t = \sqrt{1-\beta_t} x_{t-1} + \sqrt{\beta_t}\epsilon_t$, where $\epsilon_t \sim N(0, I)$. Let $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$. What is the resulting forward model distribution at step t conditioned on x_0? Hint: you can start by writing x_2 as a linear combination of x_0 and Gaussian noise $\epsilon \sim N(0, I)$ and note I is the identity 2d matrix.

   (b) Since the forward process is applied the same to each example in our dataset, we can ask what the distribution is over x_t marginally across the examples. Write down an expression for this distribution. You can assume that the examples are selected with equal probability, i.e., $q(x_0) = 1/2$ for $x_0 = [1,0]^T$ or $x_0 = [0,1]^T$.

   (c) Suppose we use a simple estimation criterion for our reverse process, i.e., we find $\epsilon_\theta(x_t, t)$ that minimizes

$$E_{x_0,t,\epsilon} \left\{ \|\epsilon - \epsilon_\theta(x_t(x_0,\epsilon), t)\|^2 \right\}$$

   where $x_t(x_0,\epsilon) = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$ and $x_0 \sim q(x_0)$, $\epsilon \sim N(0, I)$. Consider a fixed $\hat{x}$. What is the resulting optimal estimate for $\epsilon_\theta(\hat{x}, t)$ if our reverse model can be arbitrarily complex? Write down the solution as an expression involving ϵ, $x_t(x_0,\epsilon)$ and $\hat{x}$.

   (d) To evaluate your answer to the previous question note that you can think of the problem in terms of a graphical model $x_0 \to x_t$, $\epsilon \to x_t$ where we know the marginal distributions over x_0 and ϵ and how they give rise to x_t through $x_t(x_0,\epsilon)$. We observe $x_t = \hat{x}$ and wish to calculate the resulting posterior over ϵ. What is this posterior?

   (e) Briefly describe how the optimal answer for the reverse process, i.e., our estimate $\epsilon_\theta(\hat{x}, t)$ for a fixed $\hat{x}$, behaves as t becomes very large.

5 Flow Matching

9. This question is based on lecture 23; for more background see Yaron Lipman, Ricky T.Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le, "Flow Matching for Generative Modeling."

   Given sample space $\mathbb{R}^d$ we can use a continuous probability "flow" to represent a target distribution $p_1(x)$, by learning an invertible transformation from some simple known distribution $p_0(x)$. We will use this method to model a data distribution $q(x) \sim \text{Unif}(\{x^{(1)}, \ldots, x^{(n)}\})$.

   (a) A *probability flow* is continuous time-indexed function, such that for all $t \in [0,1]$, p_t is a pdf over $\mathbb{R}^d$. We are going to think about the probability flow[1] induced by fixing p_0 and p_1, then defining a distribution of linear paths

   $$\begin{aligned} x_0 &\sim p_0(x) \\ x_1 &\sim p_1(x) \\ x_t &= (1-t)x_0 + tx_1 \end{aligned}$$

   We'll start by considering the case of a fixed x_1. If $x_0 \sim \mathcal{N}(0, I)$, what is the distribution $p_t(x_t \mid x_1)$ such that $x_t \sim p_t$?

   (b) What is $p_t(x \mid x_1)$ as $t \to 1$?

   (c) Now, more generally, if $x_1 \sim p_1(x)$ what is $p_t(x)$?

   (d) The whole reason we're trying to find alternative ways of thinking about learning p_1 is that complicated densities are hard to represent and learn directly. An alternative parameterization of the whole probability flow p_t is in terms of a time varying vector field $dx_t/dt = v_t(x_t)$, that intuitively has the property that if we start with p_0, and let the probability "flow" as specified by this vector field, the distributions p_t will match the ones we desire and, in particular, will converge to p_1 as $t \to 1$.

   The *continuity equation* from fluid flow tells us the relationship between this velocity field and the probability flow:

   $$\frac{d}{dt} p_t(x) = -\nabla_x \cdot (p_t(x) v_t(x))$$

   In one dimension, for intuition, this is simply[2]

   $$\frac{d}{dt} p_t(x) = -\frac{d}{dx} p_t(x) v_t(x)$$

   So now. We know what we want our probability flow to be: p_t. What is the v_t that will result in our desired p_t?

   $$v_t(x) = \int_{x_1} \frac{x_1 - x}{1-t} p(x_1 \mid x, t) dx_1$$

   Explain intuitively why this makes sense.

   (e) What is $p(x_1 \mid x, t)$?

[1] This is called a *probability path* in the paper.

[2] You may have forgotten the notation but $\nabla_x \cdot$ is the *divergence* operator. Go look it up.

(f) Show that our definitions of $v_t(x)$ and $p_t(x)$ satisfy the continuity equation, in 1D, and assuming $x_0 \sim \mathcal{N}(0, 1)$.

It's kind of tedious to do by hand (and Mathematica can do it!) so fine to use the fact that for a fixed x_1,

$$\frac{d}{dt} p_t(x \mid x_1) = -\frac{d}{dx}(p_t(x \mid x_1) v_t(x \mid x_1))$$
$$\frac{d}{dt} \mathcal{N}(tx_1, (1-t)^2) = -\frac{d}{dx} \mathcal{N}(tx_1, (1-t)^2) \frac{x_1 - x}{1-t}$$

(g) Whew! The key takeaway here was that we showed letting the velocities be

$$v_t(x) = \int_{x_1} \frac{x_1 - x}{1-t} p_t(x_1 \mid x) dx_1$$

would yield the right probability flow.

Use this insight to describe a stochastic gradient descent training procedure for learning a neural-network approximation $v_\theta(x, t)t$ to v_t from dataset $\mathcal{D} = \{x^{(1)}, \ldots, x^{(n)}\}$.

(h) Finally, once we have trained $v_\theta(x_t, t)$, how do we sample from $\hat{p_1}$?