EECS 182 Deep Neural Networks
Spring 2023 Anant Sahai

Review: Generative Models

1. Reparameterization Trick

Formally, a latent variable model p is a probability distribution over observed variables x and latent variables z (variables that are not directly observed but inferred), $p_\theta(x, z)$. Because we know z is unobserved, using learning methods learned in class (like supervised learning methods) is unsuitable. Indeed, our learning problem of maximizing the log-likelihood of the data turns from:

$$\theta \leftarrow arg \max_\theta \frac{1}{N} \Sigma_{i=1}^N \log[p_\theta(x_i)]$$

to:

$$\theta \leftarrow arg \max_\theta \frac{1}{N} \Sigma_{i=1}^N \log[\int p_\theta(x_i \mid z) p(z) dz]$$

where $p(x)$ has become $\int p_\theta(x_i \mid z)p(z)dz$.

(a) Instead of directly optimizing the likelihood of $p(x)$, we define the proxy likelihood as:

$$\mathcal{L}(x_i, \theta, \phi) = E_{z \sim q_\phi(z|x_i)} \Big[\log[p_\theta(x_i \mid z)] \Big] - D_{KL} \Big[q_\phi(z \mid x_i) || p(z) \Big]$$

This proxy term is a *lower bound* of the original likelihood. In order to optimize this variational lower bound, **which distribution do we sample from?**

Solution: We sample from $q_\phi(z \mid x_i)$

(b) How do we take gradients through samples? To do we, we need to show how sampling can be done as a deterministic and continuous function of the model parameters θ and the independent source of randomness (ie. the *prior*). Such an explicit representation of sampling is called **reparameterization**. Consider the case where the data x is sampled from a normal distribution with its mean parameterized by parameters θ and variance of 1, with our objective being a quadratic function of x:

$$\min_\theta E_q[x^2]$$

Write x as a function of ϵ, a vector sampled from a standard Normal $\mathcal{N}(0,1)$, and compute the gradient of the expectation term above:

Solution: We can first make the stochastic element in q independent of θ, and rewrite x as:

$$x = \theta + \epsilon, \epsilon \sim \mathcal{N}(0,1)$$

$$E_q[x^2] = E_\epsilon[(\theta + \epsilon)^2]$$

Hence we can write the derivative of $E_q[x^2]$ as:

$$\begin{aligned}\nabla_\theta E_q[x^2] &= \nabla_\theta E_\epsilon[(\theta+\epsilon)^2] \\ &= E_\epsilon[2(\theta+\epsilon)]\end{aligned}$$

2. Latent Variable Models

(a) **Describe what the encoder and decoder of the VAE are *respectively* doing** to capture and encode this information into a latent representation of space z. **Is the latent space dimension smaller that the input space? How is the information bottleneck created in VAE as opposed to Autoencoder.**

Solution:

i. **Encoder** - Encoder maps a high-dimensional input x (like the pixels of an image) and then (most often) outputs the parameters of a Gaussian distribution that specify the hidden variable z. In other words, they output $\mu_{z|x}$ and $\Sigma_{z|x}$. We will implement this as a deep neural network, parameterized by ϕ, which computes the probability $q_\phi(z|x)$. We could then sample from this distribution to get noisy values of the representation z.

ii. **Decoder** - Decoder maps the latent representation back to a high dimensional reconstruction, denoted as $\hat{x}$, and outputs the parameters to the probability distribution of the data. We will implement this as another neural network, parametrized by θ, which computes the probability $p_\theta(x|z)$. In the MNIST dataset example, if we represent each pixel as a 0 (black) or 1 (white), the probability distribution of a single pixel can be then represented using a Bernoulli distribution. Indeed, the decoder gets as input the latent representation of a digit z and outputs 784 Bernoulli parameters, one for each of the 784 pixels in the image.

(b) Once the VAE is trained, **how do we use it to generate a new fresh sample from the learned ap-**

proximation of the data-generating distribution?

Solution: We can now use only the Decoder network ($p_\theta(x \mid z)$). Here, instead of sampling z from the posterior that we had during training, we sample from our true generative process which is the prior that we had specified ($z \sim \mathcal{N}(0, I)$) and we proceed to use the network to sample $\hat{x}$ from there.

(c) In the previous question we have used a proxy likelihood:

$$\mathcal{L}(x_i, \theta, \phi) = E_{z \sim q_\phi(z|x_i)}\Big[\log[p_\theta(x_i \mid z)]\Big] - D_{KL}\Big[q_\phi(z \mid x_i)||p(z)\Big]$$

Please show that $\mathcal{L}(x_i, \theta, \phi)$ is always a lower bound to the true log likelihood for x_i.

Solution:

$$\begin{aligned}
\log p_\theta(x_i) &= E_{z \sim q_\phi(z|x_i)}\Big[\log p_\theta(x_i)\Big] \\
&= E_{z \sim q_\phi(z|x_i)}\Big[\log \frac{p_\theta(x_i \mid z) p_\theta(z)}{p_\theta(z \mid x_i)}\Big] \\
&= E_{z \sim q_\phi(z|x_i)}\Big[\log \frac{p_\theta(x_i \mid z) p_\theta(z)}{p_\theta(z \mid x_i)} \frac{q_\phi(z \mid x_i)}{q_\phi(z \mid x_i)}\Big] \\
&= E_{z \sim q_\phi(z|x_i)}\Big[\log p_\theta(x_i \mid z)\Big] - E_{z \sim q_\phi(z|x_i)}\Big[\log \frac{q_\phi(z \mid x_i)}{p_\theta(z)}\Big] + E_{z \sim q_\phi(z|x_i)}\Big[\log \frac{q_\phi(z \mid x_i)}{p_\theta(z \mid x_i)}\Big] \\
&= E_{z \sim q_\phi(z|x_i)}\Big[\log p_\theta(x_i \mid z)\Big] - D_{KL}(q_\phi(z \mid x_i)||p_\theta(z)) + D_{KL}(q_\phi(z \mid x_i)||p_\theta(z \mid x_i)) \\
&= \mathcal{L}(x_i, \theta, \phi) + D_{KL}(q_\phi(z \mid x_i)||p_\theta(z \mid x_i))
\end{aligned}$$

Because $D_{KL}(q_\phi(z \mid x_i)||p_\theta(z \mid x_i)) \geq 0$, and is not tractable due to $p_\theta(z \mid x_i)$ we can conclude that:
$\log p_\theta(x_i) \geq \mathcal{L}(x_i, \theta, \phi) = E_{z \sim q_\phi(z|x_i)}\Big[\log p_\theta(x_i \mid z)\Big] - D_{KL}(q_\phi(z \mid x_i)||p_\theta(z))$
Alternatively we could use Jensen's Inequality, which states, $\log E[X] \geq E[\log X]$ to show that:

$$\Sigma_{i=1}^N \log[p_\theta(x_i)] \geq \Sigma_{i=1}^N E_{q(z|x_i)}[\log\big(p_\theta(z)\big) - \log\big(p_q(z \mid x_i)\big) + \log\big(p_\theta(x_i \mid z)\big)]$$

That is:
We first write out the log-likelihood objective of a discrete latent variable model.

$$arg \max_\theta \frac{1}{N} \Sigma_{i=1}^N \log[p_\theta(x_i)] = arg \max_\theta \frac{1}{N} \Sigma_{i=1}^N log[\Sigma_z p_\theta(x_i \mid z) p_\theta(z)]$$

then,

$$\Sigma_{i=1}^N \log[p_\theta(x_i)] = \Sigma_{i=1}^N \Big(\Sigma_z \log[p_\theta(z) p_\theta(x_i \mid z)]\Big)$$

$$= \Sigma_{i=1}^{N}\left(\Sigma_z \log[\frac{q_\phi(z \mid x_i)}{q_\phi(z \mid x_i)} p_\theta(z) p_\theta(x_i \mid z)]\right)$$

$$= \Sigma_{i=1}^{N}\left(\Sigma_z \log E_{q_\phi(z|x_i)}[\frac{1}{q_\phi(z \mid x_i)} p_\theta(z) p_\theta(x_i \mid z)]\right)$$

$$\Sigma_{i=1}^{N} \log[p_\theta(x_i)] \geq \Sigma_{i=1}^{N} E_{q(z|x_i)}[\log\left(p_\theta(z)\right) - \log\left(p_q(z \mid x_i)\right) + \log\left(p_\theta(x_i \mid z)\right)]$$

3. Diffusion Models

In the previous question we considered sampling from a discrete distribution. Let's now see how iteratively adding Gaussian noise to a data point leads to a noisy sequence, and how the reverse process refines noise to generate realistic samples.

The classes of generative models we've considered so far (VAEs, GANs), typically introduce some sort of bottleneck (*latent representation* $\mathbf{z}$) that captures the essence of the high-dimensional sample space ($\mathbf{x}$). An alternate view of representing probability distributions $p(\mathbf{x})$ is by reasoning about the *score function* i.e. the gradient of the log probability density function $\nabla_{\mathbf{x}} \log p(\mathbf{x})$.

Given a data point sampled from a real data distribution $\mathbf{x}_0 \sim q(\mathbf{x})$, let us define a *forward diffusion process* iteratively adding small amount of Gaussian noise to the sample in T steps, producing a sequence of noisy samples $\mathbf{x}_1, ..\mathbf{x}_T$.

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t I) \qquad q(\mathbf{x}_{1:T}|x_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \tag{1}$$

The data sample $\mathbf{x}_0$ gradually loses its distinguishable features as the step t becomes larger. Eventually when $T \to \infty$, $\mathbf{x}_T$ is equivalent to an isotropic Gaussian distribution. (You can assume $\mathbf{x}_0$ is Gaussian).

To generative model is therefore the *reverse diffusion process*, where we sample noise from an isotropic Gaussian, and iteratively refine it towards a realistic sample by reasoning about $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$.

(a) **Anytime Sampling from Intermediate Distributions**

Given $\mathbf{x}_0$ and the stochastic process in eq. (1), **show that there exists a closed form distribution for sampling directly at the t^{th} time-step of the form**

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\alpha_t}\mathbf{x}_0, (1-\alpha_t)I)$$

Solution: Recall the reparameterization trick, where to sample from a Gaussian $\mathbf{x} \sim \mathcal{N}(\mu, \sigma^2)$, we could consider the following sampling process:

$$\mathbf{x} = \mu + \sigma\epsilon \qquad\qquad \text{where } \epsilon \sim \mathcal{N}(0,1)$$

Therefore, defining $\gamma_t = 1 - \beta_t$, we have

$$\begin{aligned}
\mathbf{x}_t &= \sqrt{\gamma_t}\mathbf{x}_{t-1} + \sqrt{(1-\gamma_t)}\epsilon_{t-1} && \text{where } \epsilon_{t-1} \sim \mathcal{N}(0, I) \\
&= \sqrt{\gamma_t}\Big(\sqrt{\gamma_{t-1}}\mathbf{x}_{t-2} + \sqrt{(1-\gamma_{t-1})}\epsilon_{t-2}\Big) + \sqrt{(1-\gamma_t)}\epsilon_{t-1} && \text{where } \epsilon_{t-2} \sim \mathcal{N}(0, I)
\end{aligned}$$

To simplify this, recall the following lemma, where mixing two Gausssians $\mathcal{N}(0, \sigma_1^2)$ and $\mathcal{N}(0, \sigma_2^2)$ gives a Gaussian $\mathcal{N}(0, \sigma_1^2 + \sigma_2^2)$. Therefore, mixing samples ϵ_1, ϵ_2. Building on this insight, we can combine the noise components ϵ_1, ϵ_2 into a new random variable:

$$\begin{aligned}
\hat{\epsilon}_{t-2} &\sim \mathcal{N}(0, (\gamma_t(1-\gamma_{t-1}) + (1-\gamma_t))I) \\
&\sim \mathcal{N}(0, (1-\gamma_t\gamma_{t-1})I) \\
\therefore \mathbf{x}_t &= \sqrt{\gamma_t\gamma_{t-1}}\mathbf{x}_{t-2} + \sqrt{(1-\gamma_t\gamma_{t-1})}\hat{\epsilon}_{t-2}
\end{aligned}$$

Unrolling this recursion, we would get the base case, where for $\mathbf{x}_0$ the samples are

$$\mathbf{x}_t = \sqrt{\Pi_{i=1}^t \gamma_i}\mathbf{x}_0 + \sqrt{1 - \Pi_{i=1}^t \gamma_i}\epsilon$$

Therefore, by introducting $\alpha_t = \Pi_{i=1}^t \gamma_i$ we get that

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\alpha_t}\mathbf{x}_0, (1-\alpha_t)I)$$

(b) **Reversing the Diffusion Process**

Reversing the diffusion process from *real* to *noise* would allow us to sample from the real data distribution. In particular, we would want to draw samples from $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$. **Show that given $\mathbf{x}_0$, the reverse conditional probability distribution is tractable and given by**

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \mu(\mathbf{x}_t, \mathbf{x}_0), \hat{\beta}_t I)$$

- *Hint: Use Bayes Rule on eq.* (1) *, assuming that $\mathbf{x}_0$ is drawn from Gaussian $q(\mathbf{x})$)*
- *Hint: When applying Bayes rule to compute $q(x_{t-1}|x_t, x_0)$, don't expand the entire Gaussion pdf. Instead just compute the exponent parts to simplify your work.*
- *Hint: Scalar form of Gaussian pdf is given as $f(z) = \frac{1}{\sigma\sqrt{2\pi}}\exp\Big\{-\frac{1}{2}(\frac{z-\mu}{\sigma})^2\Big\}$*

Solution: Applying Bayes rule on $q(x_t|x_{t-1}, x_0)$ we get the following expression

$$q(x_{t-1}|x_t, x_0) = q(x_t|x_{t-1}, x_0)\frac{q(x_{t-1}|x_0)}{q(x_t|x_0)}$$

From part (a) we know the densities as

$$q(x_t|x_0) \sim \mathcal{N}(\sqrt{\alpha_t}x_0, (1-\alpha_t)I)$$
$$q(x_t|x_{t-1}, x_0) \sim \mathcal{N}(\sqrt{1-\beta_t}x_{t-1}, \beta_t I)$$

Therefore by plugging into the Bayes rule, we recover (upto proportionality constants)

$$\begin{aligned}
q(x_{t-1}|x_t, x_0) &\propto \exp\Big(-\frac{1}{2}\Big\{\frac{(x_t - \sqrt{1-\beta_t}x_{t-1})^2}{\beta_t} + \frac{(x_{t-1} - \sqrt{\alpha_{t-1}}x_0)^2}{1-\alpha_{t-1}} - \frac{(x_t - \sqrt{\alpha_t}x_0)^2}{1-\alpha_t}\Big\}\Big) \\
&\propto \exp\Big(-\frac{1}{2}\Big\{\frac{x_t^2 - 2\sqrt{1-\beta_t}x_{t-1}x_t + (1-\beta_t)x_{t-1}^2}{\beta_t} + \\
&\qquad \frac{x_{t-1}^2 - 2\sqrt{\alpha_{t-1}}x_0x_{t-1} + \alpha_{t-1}x_0^2}{1-\alpha_{t-1}} - \frac{(x_t - \sqrt{\alpha_t}x_0)^2}{1-\alpha_t}\Big\}\Big)
\end{aligned}$$

Simplifying the expression we get

$$q(x_{t-1}|x_t, x_0) \propto \exp\Big(-\frac{1}{2}\Big\{(\frac{1-\beta_t}{\beta_t} + \frac{1}{1-\alpha_t})x_{t-1}^2 - (\frac{2\sqrt{1-\beta_t}}{\beta_t}x_t + \frac{2\sqrt{\alpha_t}}{1-\alpha_t}x_0)x_{t-1} + H(x_t, x_0)\Big\}\Big)$$

where $H(x_t, x_0)$ is independent of x_{t-1} and therefore would be normalized out. Comparing to the expression for Gaussian $\mathcal{N}(\mu, \sigma^2)$

$$\mathcal{N}(\mu, \sigma^2) \propto \exp\Big(-\frac{1}{2}\Big\{\frac{x^2 - 2\mu x + \mu^2}{\sigma^2}\Big\}\Big)$$

we recover the expression for mean, variance of $q(x_{t-1}|x_t, x_0)$ as

$$\begin{aligned}
\hat{\beta}_t &= 1/\Big(\frac{1-\beta_t}{\beta_t} + \frac{1}{1-\alpha_t}\Big) \\
&= \frac{1-\alpha_{t-1}}{1-\alpha_t}\beta_t \qquad \Big(\text{recall } \alpha_t = \prod_{i=1}^{T}(1-\beta_t)\Big) \\
\mu(x_t, x_0) &= \Big(\frac{\sqrt{1-\beta_t}}{\beta_t}x_t + \frac{\sqrt{\alpha_t}}{1-\alpha_t}x_0\Big)\Big/\Big(\frac{1-\beta_t}{\beta_t} + \frac{1}{1-\alpha_t}\Big) \\
&= \frac{\sqrt{1-\beta_t}(1-\alpha_t)}{1-\alpha_t}x_t + \frac{\beta_t\sqrt{\alpha_{t-1}}}{1-\alpha_t}x_0
\end{aligned}$$

Therefore, under our assumptions, the distribution of $q(x_{t-1}|x_t, x_0) \sim \mathcal{N}(\mu(x_t, x_0), \hat{\beta}_t I)$. □

