Exercises
Computational Intelligence Lab
SS 2020

Machine Learning Institute
Dept. of Computer Science, ETH Zürich
Prof. Dr. Thomas Hofmann
Web http://da.inf.ethz.ch/cil

Series 9, May 7-8, 2020 (Generative Models)

Problem 1 (Variational lower bound for generative models):

1. We transform the likelihood as follows to derive the ELBO:

$$\begin{aligned}\log p_\theta(\mathbf{x}^{(i)}) &= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}[\log p_\theta(\mathbf{x}^{(i)})] \\ &= \mathbb{E}_{\mathbf{z}}\left[\log \frac{p_\theta(\mathbf{x}^{(i)}|\mathbf{z})p_\theta(\mathbf{z})}{p_\theta(\mathbf{z}|\mathbf{x}^{(i)})}\right] \\ &= \mathbb{E}_{\mathbf{z}}\left[\log \frac{p_\theta(\mathbf{x}^{(i)}|\mathbf{z})p_\theta(\mathbf{z})}{p_\theta(\mathbf{z}|\mathbf{x}^{(i)})}\frac{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}\right] \\ &= \mathbb{E}_{\mathbf{z}}[\log p_\theta(\mathbf{x}^{(i)}|\mathbf{z})] - \mathbb{E}_{\mathbf{z}}\left[\log \frac{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}{p_\theta(\mathbf{z})}\right] + \mathbb{E}_{\mathbf{z}}\left[\log \frac{q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}{p_\theta(\mathbf{z}|\mathbf{x}^{(i)})}\right] \\ &= \mathbb{E}_{\mathbf{z}}[\log p_\theta(\mathbf{x}^{(i)}|\mathbf{z})] - D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})||p_\theta(\mathbf{z})) + D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})||p_\theta(\mathbf{z}|\mathbf{x}^{(i)}))\end{aligned}$$

   In the first identity we used that $\mathbf{z} \sim q_\Phi(\mathbf{z}|\mathbf{x}^{(i)})$ does not depend on $\log p_\theta(\mathbf{x}^{(i)})$, so we can introduce the expectation, (*as given in the hint* $\mathbb{E}_X(f(Y)) = f(Y)$ if X does not depend on Y). The second equality follows from using Bayes rule in reverse to introduce the desired probabilities. Lastly we multiply the expression by a constant to introduce the variational distribution q_Φ.

2. We can group the expression into two terms as:

$$\log p_\theta(\mathbf{x}^{(i)}) = \mathcal{L}(\theta, \Phi, \mathbf{x}^{(i)}) + \underbrace{D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})||p_\theta(\mathbf{z}|\mathbf{x}^{(i)}))}_{\geq 0}$$

   where the ELBO is $\mathcal{L}(\theta, \Phi, \mathbf{x}^{(i)}) = \mathbb{E}_{\mathbf{z}}[\log p_\theta(\mathbf{x}^{(i)}|\mathbf{z})] - D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})||p_\theta(\mathbf{z}))$.

3. The term $-D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})||p_\theta(\mathbf{z}))$ encourages the posterior distribution to be close to the prior distribution on the latent variables $\mathbf{z}$, which acts as a regularizer. The first term $\mathbb{E}_{\mathbf{z}}[\log p_\theta(\mathbf{x}^{(i)}|\mathbf{z})]$ maximizes the likelihood of the generated data $\mathbf{x}^{(i)}$, which is related to reconstruction quality.

4. Using the reparameterization trick, $\mathbf{z} = g_\phi(\boldsymbol{\epsilon}, \mathbf{x})$ with , we can sample $\boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon})$ and use Monte Carlo estimate. The ELBO then becomes

$$\begin{aligned}\mathcal{L}(\theta, \phi; \mathbf{x}^{(i)}) &= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x}^{(i)})}[-\log q_\phi(\mathbf{z}|\mathbf{x}^{(i)}) + \log p_\theta(\mathbf{x}^{(i)}, \mathbf{z})] \\ &\approx \frac{1}{L}\sum_{l=1}^{L}[\log p_\theta(x^{(i)}, \mathbf{z}^{(i,l)}) - \log q_\phi(\mathbf{z}^{(i,l)}|\mathbf{x}^{(i)})]\end{aligned}$$

   where $\mathbf{z}^{(i,l)} = g_\phi(\boldsymbol{\epsilon}^{(i,l)}, \mathbf{x}^{(i)})$ and $\boldsymbol{\epsilon}^{(l)} \sim p(\boldsymbol{\epsilon})$.
   Doing so we express the random variable $\mathbf{z}$ as a deterministic variable where $\boldsymbol{\epsilon}$ is an auxiliary variable with independent marginal $p(\boldsymbol{\epsilon})$ and $g_\phi(\cdot)$ is some vector-valued function parameterized by ϕ. Hence the Monte Carlo estimates is now differentiable w.r.t. ϕ and backpropagation can be used in order to tune the parameters of the model.

5. A valid reparameterization would be $z = \mu + \sigma\epsilon$. Indeed z follows a Gaussian distribution and the mean is μ, while the variance is $\sigma^2 Var(\epsilon) = \sigma^2$. Hence we obtain

$$\mathbb{E}_{\mathcal{N}(z;\mu,\sigma^2)}[f(z)] = \mathbb{E}_{\mathcal{N}(\epsilon;0,1)}[f(\mu + \sigma\epsilon)] \approx \frac{1}{N}\sum_{l=1}^{L} f(\mu + \sigma\epsilon^{(l)})$$

   where $\epsilon^{(l)} \sim \mathcal{N}(0, 1)$.

6. Both networks are jointly trained to solve the optimization problem $\theta^*, \Phi^* = \text{argmax}_{\theta,\Phi} \mathcal{L}(x^{(i)}, \theta, \Phi)$. In a VAE the encoder network optimizes the regularizer term, by making the approximte posterior distribution q_Φ close to the latent variable prior. We then sample a latent sample $\mathbf{z} \sim q_\Phi$ and pass it to the decoder network. The decoder network then produces samples $\hat{x}$ that maximize the reconstruction quality with respect to the original input. As both terms of the ELBO are differentiable, we can use a training method like SGD to train the VAE model end-to-end.
   Classical autoencoders learn a deterministic function that compresses the data while variational autoencoders (VAEs) learn the parameters of a probability distribution representing the data. Since VAEs learn to model the data, we can sample from the distribution and generate new input data samples. Hence, VAEs are generative models.

7. We proceed by integrating the two terms of the difference under the integral separately, and then plugging back into the definition of the Kullback-Leibler divergence D_{KL}:

$$\begin{aligned}
\int q_\phi(z) \log q_\phi(z) dz &= \int \mathcal{N}(z; \mu, \Sigma) \log \mathcal{N}(z; \mu, \Sigma) dz \\
&= \mathbb{E}_z[\log \mathcal{N}(z; \mu, \Sigma)] \\
&= \mathbb{E}_z\left[\log \frac{1}{\sqrt{(2\pi)^J |\Sigma|}} \exp\left\{-\frac{1}{2}(z-\mu)^T \Sigma^{-1}(z-\mu)\right\}\right] \\
&= -\frac{J}{2}\log 2\pi + \sum_{j=1}^{J}\left(-\frac{1}{2}\log \sigma_j^2 - \frac{1}{2\sigma_j^2}\mathbb{E}_z[(z-\mu_j)^2]\right) \\
&= -\frac{J}{2}\log 2\pi + \sum_{j=1}^{J}\left(-\frac{1}{2}\log \sigma_j^2 - \frac{1}{2}\right) \\
&= -\frac{J}{2}\log 2\pi - \frac{1}{2}\sum_{j=1}^{J}\left(\log \sigma_j^2 + 1\right)
\end{aligned}$$

Since the posterior approximation is a multivariate Gaussian distribution with diagonal covariance (Σ), each dimension is independent and the log-likelihood becomes a sum of log-likelihoods per dimension parameterized by μ_j and σ_j^2

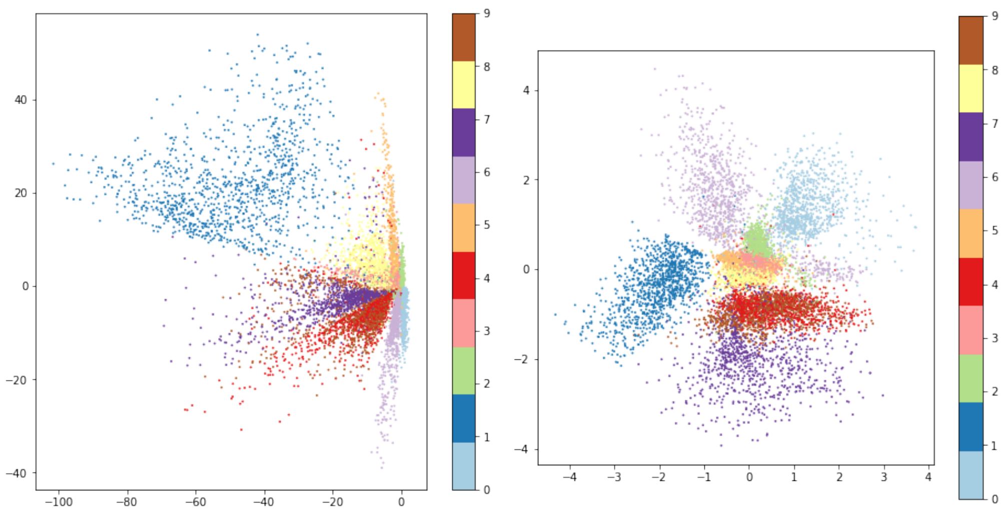
$$\begin{aligned}
\int q_\phi(z) \log p_\theta(z) dz &= \int \mathcal{N}(z; \mu, \sigma^2) \log \mathcal{N}(z; 0, \mathbb{I}) dz \\
&= \mathbb{E}_z[\log \mathcal{N}(z; 0, \mathbb{I})] \\
&= \mathbb{E}_z\left[\log \frac{1}{\sqrt{(2\pi)^J}} \exp(-\frac{1}{2}z^2)\right] \\
&= -\frac{J}{2}\log 2\pi - \frac{1}{2}\mathbb{E}_z[z^2] \\
&= -\frac{J}{2}\log 2\pi - \frac{1}{2}\sum_{j=1}^{J}(\mu_j^2 + \sigma_j^2) \\
-D_{KL} &= \int q_\phi(z)(\log p_\theta(z) - \log q_\phi(z)) dz \\
&= \frac{1}{2}\sum_{j=1}^{J}(1 + \log((\sigma_j)^2) - (\mu_j)^2 - (\sigma_j)^2)
\end{aligned}$$

which yields the desired result.

Problem 2 (Variational autoencoder):

The solution is provided in the notebook `solution09.ipynb`.

1. See code.
2. Figure 1a shows the latent space of a standard autoencoder. The space looks somewhat structured (similar digits are close to each other in the latent space), but it is unclear how to sample from this space, as some clusters are very scattered.
3. We just need to add some extra outputs to the encoder, for the variances (assuming a diagonal Gaussian). For D dimensions, we need $2D$ outputs (D means and D variances). The decoder does not need to be modified, as it takes an individual sample as input.
4. We can formulate the reparameterization trick as $\mathbf{z} = \mu(\mathbf{x}) + \sigma(\mathbf{x})\epsilon$, where ϵ is a sample from $\mathcal{N}(0, \mathbf{I})$, and $\mu(\mathbf{x})$, $\sigma(\mathbf{x})$ are the outputs of the encoder. This way, we can backpropagate through μ and σ. Predicting the full covariance matrix would require $\mathcal{O}(D^2)$ parameters (for D dimensions), which is impractical. Instead, if we restrict ourselves to a diagonal covariance matrix we only need $\mathcal{O}(D)$ parameters.
5. See code.
6. The latent space is depicted in Figure 1b. Compared to the standard autoencoder, the latent space of the variational version is more structured and compact, and makes it easier to sample from it.
7. To generate, we just sample a random $\mathbf{z}$ from $\mathcal{N}(0, \mathbf{I})$ and decode it.



(a) Autoencoder (b) Variational autoencoder

Figure 1: Comparison between the latent space of a standard autoencoder and the latent space of a variational autoencoder. The MNIST digits are compressed to 2 dimensions.