Time Series Analysis & Recurrent Neural Networks

Lecturer: Daniel Durstewitz
Tutors: Georgia Koppe, Manuel Brenner, Daniel Kramer, Leonard Bereska
WS2020/2021

Exercise 12

To be uploaded before the exercise group on 14th July, 2021.

Variational Autoencoder
The file `vae_template.py` is a code template for a variational autoencoder (VAE)[1] applied to handwritten digits of the MNIST dataset[2]. The MNIST dataset consists of images of 28×28 pixels, where each pixel value is specified in binary code: 1 for white, 0 for black. However, the reparametrization trick (see 2.) and the loss function (see 5.) are still missing in the implementation.
The VAE is maximizing the evidence lower bound (ELBO) (cf. Eq. 1), and assumes the approximate posterior to be Gaussian $q(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mu, \Sigma)$, with diagonal covariance Σ. The prior is assumed to be unit Gaussian: $p(z) = \mathcal{N}(0, I)$.

$$\text{ELBO} = \mathbb{E}_{z \sim q(z|x)} \log(p(x|z) - D_{\text{KL}}(q(z|x)||p(z)). \tag{1}$$

The parameters μ, Σ are the output of a neural network (also called *encoder*). The distribution $p(x|z)$ is also parametrized by a neural network (*decoder*). As the pixels are in binary format, we assume a Bernoulli distribution for each pixel of $p(\mathbf{x}|\mathbf{z})$. The decoder output is a vector, each dimension characterizing the probability of a pixel being white.

1. Show mathematically that minimizing the Kullback-Leibler divergence $D_{\text{KL}}(q(z|x)||p(z|x))$ between the approximate posterior $q(z|x)$ and the true posterior $p(z|x)$ is equivalent to maximizing the evidence lower bound (ELBO, Eq. 1),

2. The reparametrization trick allows for differentiable sampling from a Gaussian. For $z \sim \mathcal{N}(\mu, \sigma)$ can be reformulated to $z = \mu + \sigma\epsilon$ with $\epsilon \sim \mathcal{N}(0, 1)$. Implement the reparametrization trick for a multidimensional z in the given code template.

3. Derive the $D_{\text{KL}}(q(z|x)||p(z))$ in the case of $q(z|x) = \mathcal{N}(\mu, \Sigma)$, where Σ is diagonal and assuming $p(z) = \mathcal{N}(0, I)$.[3]

4. Convince yourself, that maximizing the likelihood of a Bernoulli distribution $p(x|z) = \phi^x(1 - \phi)^{(1-x)}$ (where ϕ is the probability vector, with the probability of a pixel being white) is giving rise to the *cross-entropy loss* $\mathcal{L}(x, \phi) = x\log(\phi) + (1 - x)\log(1 - \phi)$, for targets x and predicted probabilities ϕ.

5. Implement the ELBO as a loss function for the VAE in the given code template. Use results from 3. for the KL-divergence term and from 4. for the reconstruction term (Hint: Use `torch.nn.functional.binary_cross_entropy`).

6. Train the VAE. After training, plot samples from the learned distribution $p(x) = p(x|z)p(z)$.

[1] The original papers are by Rezende et al. (2014), `https://arxiv.org/abs/1401.4082`, and Kingma et al. (2014), `https://arxiv.org/abs/1312.6114`.
[2] `https://en.wikipedia.org/wiki/MNIST_database`
[3] You may use results from exercise 5.