Assignment 4 (practical part)

Deadline: Friday, June 14th (23:59)

May 29, 2019

NOTE: The assignment consists of a theoretical and a practical part. This is the practical part, make sure you first read the theoretical part and answer the questions there. Then, work on this practical part, using the provided Jupyter Notebook.

Question 1: Variational Autoencoder (7pt)

Task 1.1: Encoder architecture

(a) Choose and implement a suitable architecture for the encoder. The encoder should have two outputs, which we will interpret as the mean and log variance of the approximate posterior distribution $q(\boldsymbol{z}|\boldsymbol{x})$ (a Gaussian, as defined in the theoretical assignment). In this task, use a latent space dimension of 2 such that we can easily plot the latent space.

(b) Motivate your choice of architecture.

(c) What is the reason to model the logarithm of the variance, instead of the variance (or standard deviation) itself?

Task 1.2: Decoder architecture

(a) Choose and implement a suitable architecture for the decoder. Input is a sample of latent variables, the output represents the parameters of the generative distribution $p(x|z)$, i.e. the mean of a Gaussian distribution (for continuous data) or Bernoulli distribution (for binary data).

(b) Motivate your choice of architecture.

Task 1.3: Reparametrisation trick

To implement sampling with the reparametrisation trick, we will define a custom lambda layer. It takes the mean and log variance of $q(z|x)$ as input, and outputs a sample from $q(z|x)$. This is done by first sampling from a standard Gaussian (Normal) distribution, and then applying the proper transformation to obtain a sample from $q(z|x)$ with the given mean and variance. Implement this transformation.

Task 1.4: Loss function

Now we define the loss function, which you have derived in the theoretical part of the assignment. Note: we wish to maximise the ELBO, but Keras formulates training objectives as a loss function to minimise, so the loss function is the negative ELBO: $-\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] + KL(q_\phi(z|x)||p(z))$.

We split this into two terms, the reconstruction loss and the KL loss.

Reconstruction loss

We start with the first term: $-\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})]$.

We approximate the expectation with a single Monte Carlo sample (using the reparametrisation trick such that the loss function is differentiable), which we can readily obtain from the sampling layer of our encoder model (output $\mathbf{z}$). So all you need to implement is the loss function $-\log p_\theta(\boldsymbol{x}|\boldsymbol{z}_{sample})$. You've derived the formula for this in the theoretical part of the assignment, for both a Bernoulli distribution (for discrete data) and a Gaussian distribution with fixed standard deviation (for continuous data). Implement both versions. Choose a suitable standard deviation value for the Gaussian.

KL loss

The second term of the loss is the KL Divergence $KL(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z}))$, which as we saw has a closed form for our choice of posterior and prior (both Gaussians): $\frac{1}{2}\sum_{k=1}^{K}(m_k^2 + s_k^2 - \log s_k^2 - 1)$ (see the theoretical part of the assignment for explanation of the notations). Implement it.

Task 1.5: Training & evaluating the model

Train the model for 50 epochs with batch size 100. Choose whether to use the Bernoulli or Gaussian generative distribution.

Use the code provided in the notebook to generate plots of the latent space (note that they only work for a 2-dimensional latent space).

The first plot shows the mean representations in latent space for data points from the test set. Although VAEs are unsupervised, we do have label information for the MNIST data, so we can use this give data points a different colour depending on their label.

The second plot takes linearly spaced coordinates in latent space, decodes them into data space representations, and plots them in a grid according to the latent space coordinates. So each of the images are generated, not reconstructed from data.

These latent space plots can give an insight into the (latent) representations that are learned by the VAE. Evaluate what you see:

(a) Did you successfully train a generative model for this data? Motivate your answer.

(b) Discuss how the latent space is populated by the test data. What happens in "gaps" in the latent space? I.e. areas in latent space near the origin (so with high prior likelihood) but without any data points being mapped to them. What do images generated from such latent points look like?

Question 2: FashionMNIST VAE, semi-supervised learning (10pt)

In this question, we will investigate the usefulness of VAE latent representations in semi-supervised learning. Since the VAE is an unsupervised method, we don't need any labels to learn latent representations. Often, it is easy to obtain a lot of unlabelled data, but labelling this data is expensive. Thus, methods that can leverage unsupervised training to learn a supervised goal (such as classification) may be very powerful. This is the idea behind semi-supervised learning.

You will evaluate this on the FashionMNIST dataset. Although all labels are available for this dataset, we can "hide" some labels from a model, simply by not using all of them.

Besides representations learned with a VAE, you will also compare with another representation learning method: a denoising autoencoder.

Task 2.1: Unsupervised training

(a) Train two representation learning methods on the FashionMNIST dataset; a variational autoencoder (VAE) and a denoising autoencoder. Choose a suitable architecture, and encoding/latent dimension (*hint: for good results a dimension of 2 will likely be too small*). For fairness, use similar architectures for each of the models.

Use each of the models to obtain encoded representations for the full dataset (training and test set).

(b) Motivate your architectural choices.

Task 2.2: Qualitative evaluation

Plotting the population of the latent space only works for a 2-dimensional latent space. For higher dimensions, we need different ways to qualitatively evaluate the models.

(a) Reconstruct some images from the test set with both the VAE and DAE. Plot the reconstructions alongside the original images. Briefly discuss the results.

(b) The VAE is a generative model; generate some images with the VAE and visualise them. Also try to generate images with the DAE (even though it is not intended as a generative model) and visualise them. Discuss the results, in particular the difference between VAE and DAE.

Task 2.3: Semi-supervised learning

Perform a thorough evaluation of semi-supervised learning for representations learned with your variational and denoising autoencoders.
Guidelines:

- For various suitable values of l, randomly select l instances from the training dataset, these will represent your labelled data and are the only data points that may be used for supervised learning. Make sure to include $l = 60000$ (i.e. the entire dataset).
- For each value of l, train a few off-the-shelf methods from scikit-learn (e.g. random forest, SVM) as well as a simple multilayer perception (MLP) on representations from both the VAE and DAE, using only the l available labels.
- For each value of l, also train an MLP on the original image data, using only the l available labels.
- Visualise your results in a clear way, motivate your methods, and report your conclusions. Was it beneficial to use unlabelled data as well as labelled data? Which representations worked best?