Mini-project 1

Advanced Machine Learning (02460)
Technical University of Denmark
Jes Frellsen

February 2024
(Version 1.1)*

1 Formalities

This is the project description for the first mini project in *Advanced Machine Learning* (02460). The project is part of the course exam and will count towards your final grade.

Deadline You must submit your report as a group electronically via DTU Learn by 28 February 2024 at 12:00 (noon).

Groups You must do the project in groups of 3–4 people. You need an exception to deviate from this group size. You do not need to document individual contributions.

What should be handed in? You must hand in a single report in PDF format and your code in a single file (either a zip or tar archive).

Length The report must consist of:

1. A single page with the main text, including figures and tables. This page must include names, student numbers, course number and the title "Mini-project 1" (so do not include a front page).
2. Unlimited pages of references.
3. A single page of well-formatted code snippets.

You must use at least font size 10pt and margins of at least 2cm. Any content violating the space limitation will not be evaluated.

*Version 1.1: Font size of the report was change to 10pt, and it was added that you briefly need to describe architectures.

Code You may use all code you were given during weeks 1–3 in the course. If you use code from elsewhere, it must be documented in the report.

2 Project description

In this project, you will train and evaluate various deep generative models on binarized MNIST and MNIST. The project is divided into three parts.

Part A: Priors for variational autoencoders (VAEs)

We will consider different priors for VAEs with a product of Bernoulli likelihood, $p(\mathbf{x}|\mathbf{z})$, trained on binarized MNIST, where pixels with values over 0.5 are set to 1, and pixels with values less than 0.5 are set to 0. You need to train VAEs with all the following priors:

- A standard Gaussian prior.
- A mixture of Gaussian (MoG) prior.
- A Flow-based prior.

For each of the three priors, you have to:

- Show a plot of the prior and the aggregate posterior.
- Discuss and compare across priors the match between the priors and the aggregate posteriors.
- Evaluate the *test set* log-likelihood as approximated by the ELBO.
- Discuss and compare the test set log-likelihood across priors.

You must report the mean and standard derivation over multiple training runs when reporting the test set log-likelihood. You also need to briefly describe the architectures used for the encoder and decoder.

Optional 1: Instead of the MoG prior, you may also implement another prior that is not a standard Gaussian or a Flow, e.g., a hierarchical prior or a VampPrior.

Optional 2: Instead of the ELBO, you may report a better approximation of the log-likelihood, such as the importance-weighted IWAE bound (Burda et al. 2016).

Part B: Sampling quality of generative models

In the second part, we consider generative models trained on standard MNIST (i.e., non-binarized). You need to train:

- A Flow-based model.
- A Denoising Diffusion Probabilistic Model (DDPM) using a U-Net.

In the report, you need to:

- Show four representative samples for each of the two models and four samples from a VAE of you choice (including on binarized MNIST) .
- Discuss and compare the sampling quality across the three generative models.

You only need to do a qualitative comparison of the sampling quality. You also need to briefly describe the architectures used for the Layers in the Flow-based model and the DDPM.

Optional: You may also compare the sampling quality quantitively by evaluating, e.g., the Fréchet inception distance (FID, Heusel et al. 2017).

Part C: Code snippets

In the code part of the report, you must include code snippets from your implementation showing the following:

- How you evaluate the VAE ELBO for a non-Gaussian prior, e.g., a MoG prior or a Flow-based prior.
- The central parts of your Flow-based prior implementation.
- Your implementation of the DDPM ELBO.
- Your implementation of the DDPM sampling algorithm.

References

Burda, Y, RB Grosse, and R Salakhutdinov (2016). "Importance weighted autoencoders". In: *International Conference on Learning Representations.*

Heusel, M, H Ramsauer, T Unterthiner, et al. (2017). "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium". In: *Advances in Neural Information Processing Systems.* Vol. 30.