Mini-project 2

Advanced Machine Learning (02460)
Technical University of Denmark
Søren Hauberg

March 2025
(Version 1.0)

1 Formalities

This is the project description for the second 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 3 April 2025 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 follow the published LaTeX template and 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 2" (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.

Code You may use all code you were given during weeks 1–7 in the course. If you use code from elsewhere, it must be documented in the report.

2 Project description

In this project, you will estimate geometries using variational autoencoders (VAEs) on a subset of (non-binarized) MNIST. The project is divided into three parts. You must document your work in the report and provide relevant plots (example plots generated with our implementation are available in the report template for reference).

Part A: Pull-back geodesics

We will consider a subset of MNIST containing 3 classes and a total of 2048 observations. You should train a VAE with a standard Gaussian prior and a Gaussian likelihood, $p(\mathbf{x}|\mathbf{z})$. The latent space should be two-dimensional to ease plotting. The code hand-out provides a starting point. Note that the hand-out training loop adds a bit of noise to the data as this improves the ensemble studied in Part B.

You must implement an algorithm to compute geodesics under the pull-back metric associated with the mean of the Gaussian decoder. The report should include:

- A plot of the latent variables alongside geodesics between, at least, 25 random latent variable pairs.
- A code snippet of the algorithm for computing the energy of a curve.
- A code snippet of the algorithm used for computing geodesics.
- A discussion of the qualitative behavior of the computed geodesics, e.g. are they reliable across training runs?

Note that code-snippets can be shortened in the report to save space as long as they preserve the main points of the code.

Part B: Ensemble VAE geometry

In the second part, we will take model uncertainty into account to increase the robustness of the learned geometry. You must train a series of VAEs with an ensemble of decoders with up to M ensemble members. You must further implement an algorithm for computing the *model-average curve energy*, and compute geodesics by minimizing this. Specifically, the model-average curve energy can be computed with the following approximation

$$\mathcal{E}(c) \approx \sum_{i=0}^{N} \mathbb{E}_{l,k} \left\| f_l(c(t_i)) - f_k(c(t_{i+1})) \right\|^2 , \tag{1}$$

where f_l and f_k denotes decoder ensemble members drawn uniformly. This expectation can be approximated using Monte Carlo.

You must investigate the impact of using ensembles on the reliability of geodesic distances compared with Euclidean distances. For this, you must measure the coefficient of variation (CoV) of both Euclidean and geodesic distances as a function of number of ensemble decoders. The CoV is a unit-less measure of variability that can be used to determine if one distance measure is more reliable than another. Evaluating the CoV requires training multiple VAEs and evaluating the variability of distances across models. Specifically, let $(\mathbf{y}_i, \mathbf{y}_j)$ denote a fixed pair of test points (these should be the same across different models) and let $(\mathbf{x}_i^{(m)}, \mathbf{x}_j^{(m)})$ denote the means of the associated latent variables under the m^{th} VAE. Let $d_{ij}^{(m)}$ denote a distance between these latent variables (either Euclidean or geodesic). The CoV associated with $(\mathbf{y}_i, \mathbf{y}_j)$ is then

$$\mathrm{CoV}_{ij} = \frac{\sigma\left(\left\{d_{ij}^{(1)}, \dots, d_{ij}^{(M)}\right\}\right)}{\mu\left(\left\{d_{ij}^{(1)}, \dots, d_{ij}^{(M)}\right\}\right)}, \tag{2}$$

where μ and σ are the mean and standard deviation functions, respectively. You should consider, at least, $M = 10$ VAE retrainings, and $1, 2, 3$ decoders.

The report should include:

- A plot of the latent variables alongside geodesics between the same random latent variable pairs as in Part A computed using the ensemble decoder VAE.
- A plot of the coefficient of variation (CoV) of both Euclidean and geodesic distances as a function of number of ensemble decoders. The final plot should contain the average CoV across, at least, 10 point pairs.
- Relevant code snippets.

3 Remarks

- The final experiments can easily take a couple of hours to run, so be sure to plan your time. Starting the day before the deadline will, most likely, result in a incomplete report. So, don't do that :-)
- A significant practical hurdle can be to tune the optimizer used for computing geodesics, so expect that you will have to fiddle with learning rates. I recommend getting reliable results with a single decoder before moving on to ensembles.
- Note that it is essential to use the *same* point pairs across models when computing the CoV.