Deep Gaussian Process for Unsupervised Learning

Semester project, Spring 2017

Simone Rossi
Advisor Prof. Maurizio Filippone

June 22, 2017

EURECOM, Ecole d'Ingénieur et Centre de Recherche en Telecommunications

Table of contents

Introduction

Why unsupervised learning

Unsupervised learning is more subjective than supervised learning, as there is no simple goal for the analysis. But techniques for unsupervised learning are of growing importance in a number of fields:

Why unsupervised learning

Unsupervised learning is more subjective than supervised learning, as there is no simple goal for the analysis. But techniques for unsupervised learning are of growing importance in a number of fields:

- visualize and draw trends of high dimensional problems,

Why unsupervised learning

Unsupervised learning is more subjective than supervised learning, as there is no simple goal for the analysis. But techniques for unsupervised learning are of growing importance in a number of fields:

- visualize and draw trends of high dimensional problems,
- subgroups of breast cancer patients grouped by their gene expression measurements,

Why unsupervised learning

Unsupervised learning is more subjective than supervised learning, as there is no simple goal for the analysis. But techniques for unsupervised learning are of growing importance in a number of fields:

- visualize and draw trends of high dimensional problems,
- subgroups of breast cancer patients grouped by their gene expression measurements,
- groups of shoppers characterized by their browsing and purchase histories,
- movies grouped by the ratings assigned by movie viewers.

Examples of unsupervised learning

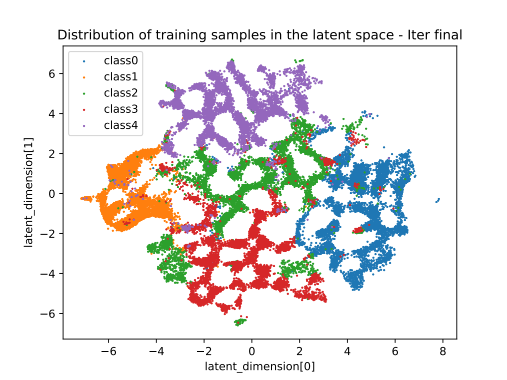


Figure 1: Feature projection of the `MNIST dataset` (5 digits)

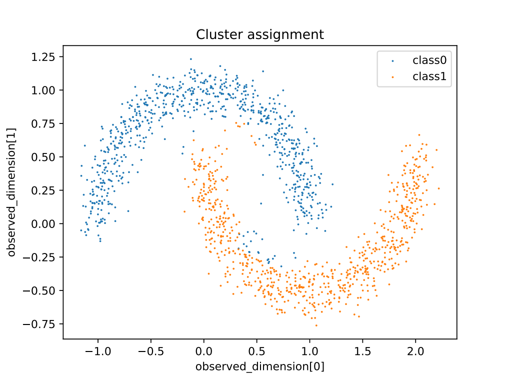


Figure 2: Clustering assignment of the `sklearn` moon dataset

Deep Gaussian Processes

Gaussian Process - Weight space

A Gaussian Process can be seen as a Bayesian linear regression with possibly infinite basis functions.

$$\bar{f}(\mathbf{x}_*) = \phi(\mathbf{x}_*)^\top \mathbf{w}. \tag{1}$$

Gaussian Process - Weight space

A Gaussian Process can be seen as a Bayesian linear regression with possibly infinite basis functions.

$$\bar{f}(\mathbf{x}_*) = \phi(\mathbf{x}_*)^\top \mathbf{w}. \tag{1}$$

Introducing the covariance function $k(\mathbf{x}, \mathbf{x}')$, it can be proved that the equation above can be written as follows

$$\bar{f}(\mathbf{x}_*) = \mathbf{k}(\mathbf{x}_*)^\top \boldsymbol{\alpha}, \tag{2}$$

where $\boldsymbol{\alpha} = K^{-1}\mathbf{y}$ and $\mathbf{k}(\mathbf{x}_*)$ denote the vector of covariances between the point $\mathbf{x}_*$ and the n training points.

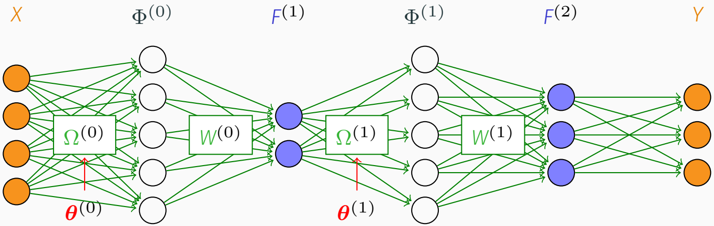
Fourier expansion

The popular RBF kernel can be approximated as follows

$$k_{\mathrm{rbf}}(\mathbf{x}_i, \mathbf{x}_j) \approx \frac{1}{N_{\mathrm{RF}}} \sum_{r=1}^{N_{\mathrm{RF}}} \mathbf{z}(\mathbf{x}_i | \tilde{\boldsymbol{\omega}}_r)^\top \mathbf{z}(\mathbf{x}_j | \tilde{\boldsymbol{\omega}}_r), \tag{3}$$

where $\mathbf{z}(\mathbf{x}|\boldsymbol{\omega}) = [\cos(\mathbf{x}^\top \boldsymbol{\omega}), \sin(\mathbf{x}^\top \boldsymbol{\omega})]^\top$ and with $\tilde{\boldsymbol{\omega}}_r \sim p(\boldsymbol{\omega})$.

Deep Architecture



This is the approximation of DGP where

$$\Phi_{\text{rbf}}^{(l)} = \sqrt{\frac{(\sigma^2)^{(l)}}{N_{\text{RF}}^{(l)}}} \left[\cos\left(F^{(l)}\Omega^{(l)}\right), \sin\left(F^{(l)}\Omega^{(l)}\right)\right], \tag{4}$$

and

$$F^{(l+1)} = \Phi_{\text{rbf}}^{(l)} W^{(l)}. \tag{5}$$

Latent Variable Models

Probabilistic Principal Component Analysis

One of the most important problems in unsupervised learning is to represent the observed data $\mathbf{Y}$ (with $\mathbf{y}_i \in R^{D_{\text{obs}}}$) in some lower dimensional embedded space $\mathbf{X}$ (with $\mathbf{x}_i \in R^{D_{\text{lat}}}$), such that

$$\mathbf{y}_i = \mathbf{W}\mathbf{x}_i + \varepsilon_i, \tag{6}$$

where $\varepsilon_i \sim \mathcal{N}(0, \sigma^2\mathbf{I})$.

Probabilistic Principal Component Analysis

One of the most important problems in unsupervised learning is to represent the observed data $\mathbf{Y}$ (with $\mathbf{y}_i \in R^{D_{\text{obs}}}$) in some lower dimensional embedded space $\mathbf{X}$ (with $\mathbf{x}_i \in R^{D_{\text{lat}}}$), such that

$$\mathbf{y}_i = \mathbf{W}\mathbf{x}_i + \varepsilon_i, \tag{6}$$

where $\varepsilon_i \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$.

The model is defined **probabilistically**, the latents are **marginalized** and the parameters are computed through **maximization**.

Probabilistic Principal Component Analysis (cont.)

Let's define the likelihood as follows

$$p(\mathbf{y}_i|\mathbf{x}_i, \sigma^2) = \mathcal{N}(\mathbf{W}\mathbf{x}_i, \sigma^2\mathbf{I}) \tag{7}$$

Probabilistic Principal Component Analysis (cont.)

Let's define the likelihood as follows

$$p(\mathbf{y}_i|\mathbf{x}_i, \sigma^2) = \mathcal{N}(\mathbf{W}\mathbf{x}_i, \sigma^2\mathbf{I}) \tag{7}$$

We can now specify a simple prior over $\mathbf{x}_i$ and integrate out the latent variable

$$p(\mathbf{y}_i|\mathbf{W}, \sigma^2) = \int p(\mathbf{y}_i|\mathbf{x}_i, \mathbf{W}, \sigma^2)p(\mathbf{x}_i)d\mathbf{x}_i$$

Probabilistic Principal Component Analysis (cont.)

Let's define the likelihood as follows

$$p(\mathbf{y}_i|\mathbf{x}_i, \sigma^2) = \mathcal{N}(\mathbf{W}\mathbf{x}_i, \sigma^2\mathbf{I}) \tag{7}$$

We can now specify a simple prior over $\mathbf{x}_i$ and integrate out the latent variable

$$p(\mathbf{y}_i|\mathbf{W}, \sigma^2) = \int p(\mathbf{y}_i|\mathbf{x}_i, \mathbf{W}, \sigma^2)p(\mathbf{x}_i)d\mathbf{x}_i = \mathcal{N}(\mathbf{0}, \mathbf{W}\mathbf{W}^\top + \sigma^2\mathbf{I}) \tag{8}$$

Probabilistic Principal Component Analysis (cont.)

Let's define the likelihood as follows

$$p(\mathbf{y}_i|\mathbf{x}_i, \sigma^2) = \mathcal{N}(\mathbf{W}\mathbf{x}_i, \sigma^2\mathbf{I}) \tag{7}$$

We can now specify a simple prior over $\mathbf{x}_i$ and integrate out the latent variable

$$p(\mathbf{y}_i|\mathbf{W}, \sigma^2) = \int p(\mathbf{y}_i|\mathbf{x}_i, \mathbf{W}, \sigma^2)p(\mathbf{x}_i)d\mathbf{x}_i = \mathcal{N}(\mathbf{0}, \mathbf{W}\mathbf{W}^\top + \sigma^2\mathbf{I}) \tag{8}$$

Thanks to point independence, the marginal likelihood on the whole dataset is

$$p(\mathbf{Y}|\mathbf{W}, \sigma^2) = \prod_{i=0}^{n-1} p(\mathbf{y}_i|\mathbf{W}, \sigma^2) \tag{9}$$

Probabilistic Principal Component Analysis (cont.)

Let's define the likelihood as follows

$$p(\mathbf{y}_i|\mathbf{x}_i, \sigma^2) = \mathcal{N}(\mathbf{W}\mathbf{x}_i, \sigma^2\mathbf{I}) \tag{7}$$

We can now specify a simple prior over $\mathbf{x}_i$ and integrate out the latent variable

$$p(\mathbf{y}_i|\mathbf{W}, \sigma^2) = \int p(\mathbf{y}_i|\mathbf{x}_i, \mathbf{W}, \sigma^2)p(\mathbf{x}_i)d\mathbf{x}_i = \mathcal{N}(\mathbf{0}, \mathbf{W}\mathbf{W}^\top + \sigma^2\mathbf{I}) \tag{8}$$

Thanks to point independence, the marginal likelihood on the whole dataset is

$$p(\mathbf{Y}|\mathbf{W}, \sigma^2) = \prod_{i=0}^{n-1} p(\mathbf{y}_i|\mathbf{W}, \sigma^2) \tag{9}$$

Finally,

$$\mathbf{W} = \arg\max_{\mathbf{W}} p(\mathbf{Y}|\mathbf{W}, \sigma^2) \tag{10}$$

Dual Probabilistic Principal Component Analysis

Differently, we can **marginalize** the parameters and compute the latents are computed through **maximization**. To do so, let's specify a prior over $\mathbf{W}$:

$$p(\mathbf{W}) = \prod_{i=0}^{D_{\text{obs}}-1} \mathcal{N}(\mathbf{w}_i|\mathbf{0}, \mathbf{I}) \tag{11}$$

Dual Probabilistic Principal Component Analysis

Differently, we can **marginalize** the parameters and compute the latents are computed through **maximization**. To do so, let's specify a prior over $\mathbf{W}$:

$$p(\mathbf{W}) = \prod_{i=0}^{D_{\text{obs}}-1} \mathcal{N}(\mathbf{w}_i|\mathbf{0}, \mathbf{I}) \tag{11}$$

The marginal likelihood has the form

$$p(\mathbf{Y}|\mathbf{X}, \sigma^2) = \prod_{i=0}^{D_{\text{obs}}-1} \int p(\mathbf{y}_i|\mathbf{x}_i, \mathbf{W}, \sigma^2) p(\mathbf{W}) d\mathbf{W}$$

Dual Probabilistic Principal Component Analysis

Differently, we can **marginalize** the parameters and compute the latents are computed through **maximization**. To do so, let's specify a prior over $\mathbf{W}$:

$$p(\mathbf{W}) = \prod_{i=0}^{D_{\text{obs}}-1} \mathcal{N}(\mathbf{w}_i|\mathbf{0}, \mathbf{I}) \tag{11}$$

The marginal likelihood has the form

$$p(\mathbf{Y}|\mathbf{X}, \sigma^2) = \prod_{i=0}^{D_{\text{obs}}-1} \int p(\mathbf{y}_i|\mathbf{x}_i, \mathbf{W}, \sigma^2)p(\mathbf{W})d\mathbf{W} = \prod_{i=0}^{D_{\text{obs}}-1} \mathcal{N}(\mathbf{y}_{:,i}|\mathbf{0}, \mathbf{X}\mathbf{X}^\top + \sigma^2\mathbf{I}). \tag{12}$$

Dual Probabilistic Principal Component Analysis

Differently, we can **marginalize** the parameters and compute the latents are computed through **maximization**. To do so, let's specify a prior over $\mathbf{W}$:

$$p(\mathbf{W}) = \prod_{i=0}^{D_{\text{obs}}-1} \mathcal{N}(\mathbf{w}_i|\mathbf{0}, \mathbf{I}) \tag{11}$$

The marginal likelihood has the form

$$p(\mathbf{Y}|\mathbf{X}, \sigma^2) = \prod_{i=0}^{D_{\text{obs}}-1} \int p(\mathbf{y}_i|\mathbf{x}_i, \mathbf{W}, \sigma^2) p(\mathbf{W}) d\mathbf{W} = \prod_{i=0}^{D_{\text{obs}}-1} \mathcal{N}(\mathbf{y}_{:,i}|\mathbf{0}, \mathbf{X}\mathbf{X}^\top + \sigma^2\mathbf{I}). \tag{12}$$

The corresponding loglikelihood is the following

$$\log p(\mathbf{Y}|\mathbf{X}, \sigma^2) = -\frac{nD_{\text{obs}}}{2} \ln(2\pi) - \frac{D_{\text{obs}}}{2} \ln|\mathbf{K}| - \frac{1}{2}\mathrm{Tr}\left(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^\top\right) \tag{13}$$

Dual Probabilistic Principal Component Analysis (cont.)

$$\mathcal{L} = \log p(\mathbf{Y}|\mathbf{X}, \sigma^2) = -\frac{nD_{\text{obs}}}{2}\ln(2\pi) - \frac{D_{\text{obs}}}{2}\ln|\mathbf{K}| - \frac{1}{2}\text{Tr}\left(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^{\top}\right) \tag{14}$$

Since $\mathbf{K} = \mathbf{X}\mathbf{X}^{\top} + \sigma^2\mathbf{I}$, this is a product of D_{obs} independent Gaussian processes with linear covariance function.

Dual Probabilistic Principal Component Analysis (cont.)

$$\mathcal{L} = \log p(\mathbf{Y}|\mathbf{X}, \sigma^2) = -\frac{nD_{\text{obs}}}{2}\ln(2\pi) - \frac{D_{\text{obs}}}{2}\ln|\mathbf{K}| - \frac{1}{2}\mathrm{Tr}\left(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^\top\right) \tag{14}$$

Since $\mathbf{K} = \mathbf{X}\mathbf{X}^\top + \sigma^2\mathbf{I}$, this is a product of D_{obs} independent Gaussian processes with linear covariance function.

The solution of the maximization problem is

$$\mathbf{X} = \mathbf{U}\mathbf{L}\mathbf{V}^\top \tag{15}$$

where $\mathbf{U}$ is an $n \times D_{\text{lat}}$ matrix whose columns are the first D_{lat} eigenvectors of $\mathbf{Y}\mathbf{Y}^\top$, $\mathbf{L}$ is the associated diagonal eigenvalue matrix and $\mathbf{V}$ is eventually a $D_{\text{obs}} \times D_{\text{obs}}$ rotation matrix.

Gaussian Process Latent Variable Model

We can now replace the inner product kernel with a covariance function so that it allows non-linear transformation to obtain a non-linear latent variable model.

$$k_{\mathrm{rbf}}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left[-\frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_j\|^2\right] \tag{16}$$

Gaussian Process Latent Variable Model

We can now replace the inner product kernel with a covariance function so that it allows non-linear transformation to obtain a non-linear latent variable model.

$$k_{\mathrm{rbf}}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left[-\frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_j\|^2\right] \tag{16}$$

Unfortunately there is not closed form solution of the maximization of the likelihood and therefore the resulting models will not be optimizable through an eigenvalue problem.

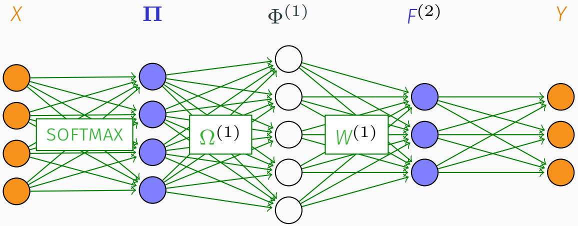
Clustering

From latents to cluster assignment

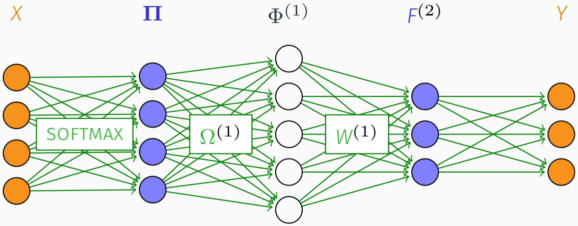
To actually making assignment to clusters, latents have to be discretized in order to give for each point a probability of being assigned to specific cluster.

From latents to cluster assignment

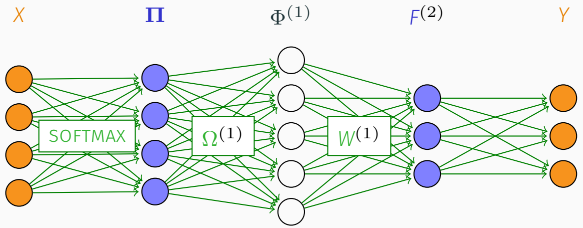
To actually making assignment to clusters, latents have to be discretized in order to give for each point a probability of being assigned to specific cluster.



From latents to cluster assignment (cont.)



From latents to cluster assignment (cont.)



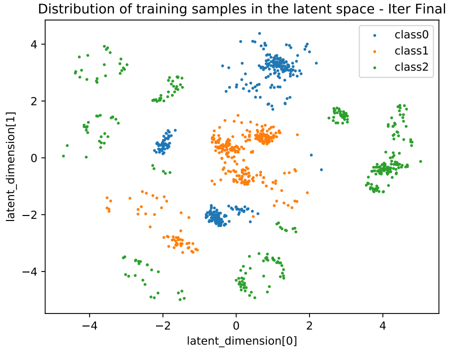
The SOFTMAX layer simply implements a softmax function on each samples of the latent space $\mathbf{X}$; in practice:

$$\mathbf{\Pi} = \frac{\exp(\mathbf{X})}{\sum_{j=0}^{Dlat-1} \exp(\mathbf{X}_{:,j})} \tag{17}$$

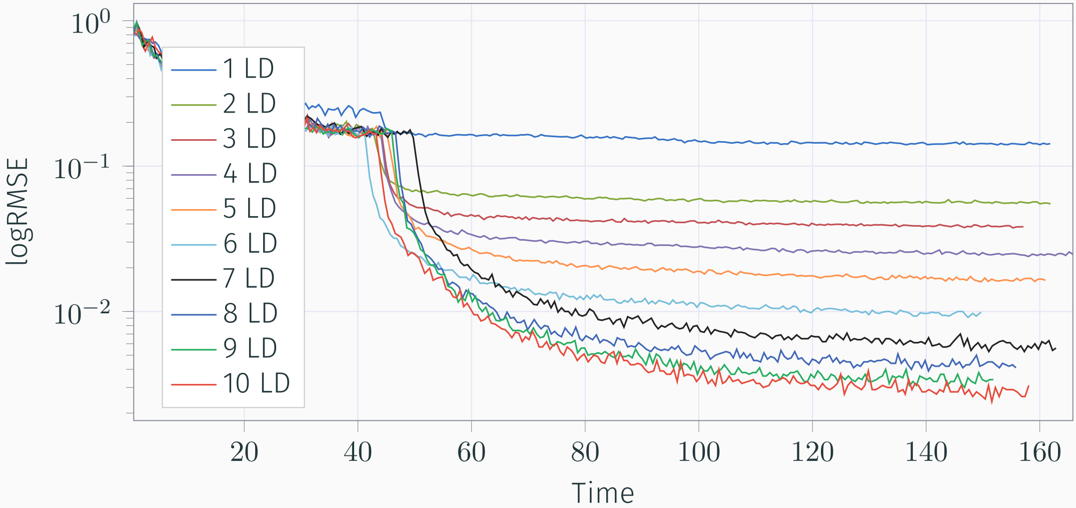
Experiments

Oil dataset

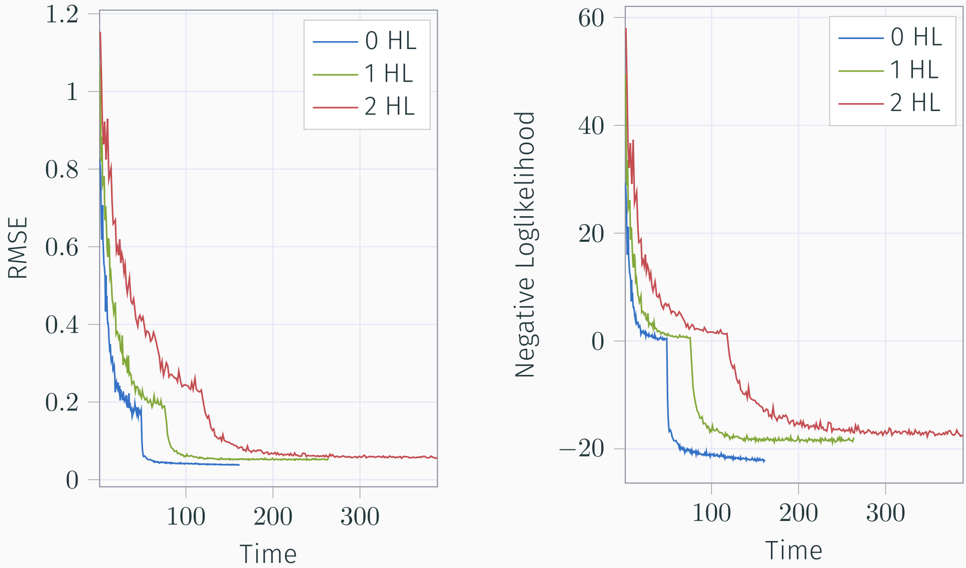
It is dataset modeling non-intrusive measurements on a oil. The flow in the pipe can be **horizontally stratified**, **nested annular** or **homogeneous**. The data lives in a 12-dimensional measurement space, but is known to live on a reduced dimensionality.



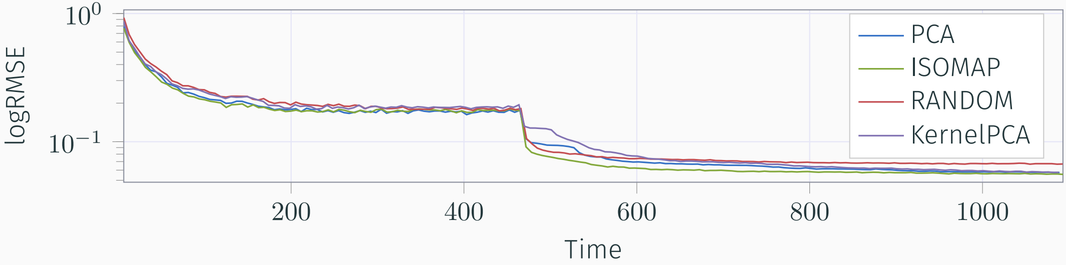
Experiment on Number of Latent Dimensions

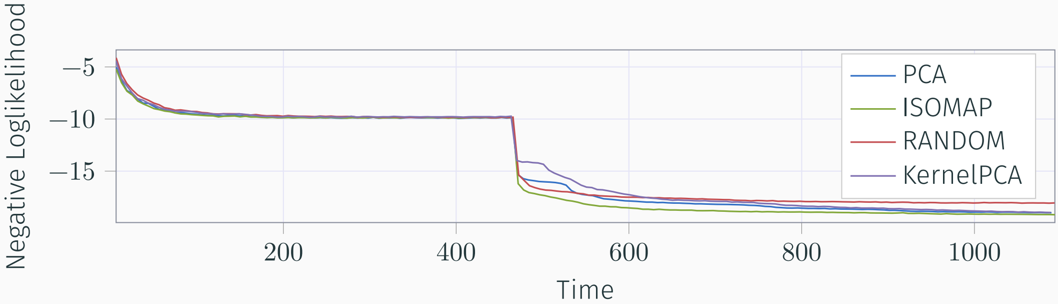


Experiment on Number of Hidden Layers

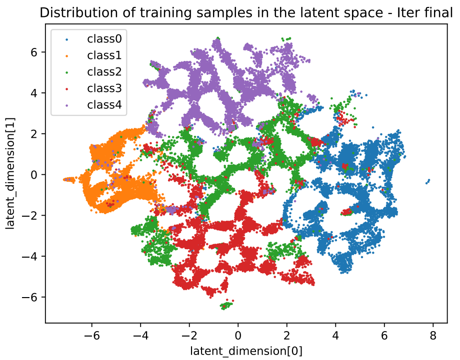


Experiment on Latent Space Initilisation

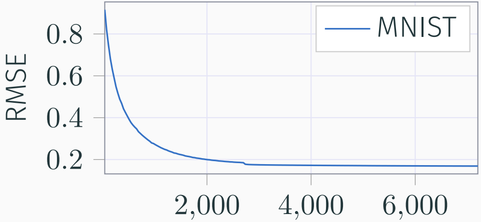




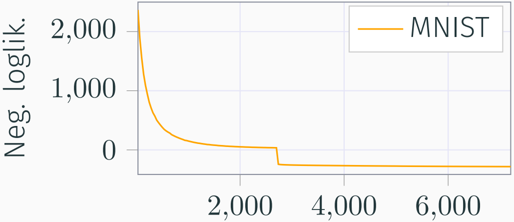
MNIST



Distribution of training samples in the latent space - Iter final
class0
class1
class2
class3
class4
latent_dimension[1]
latent_dimension[0]
−6
−4
−2
0
2
4
6
8

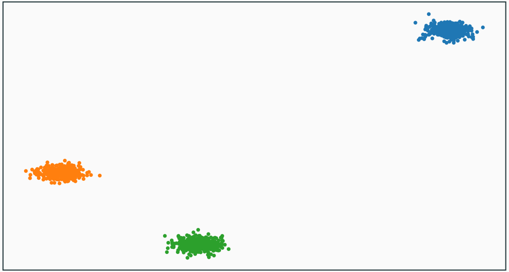


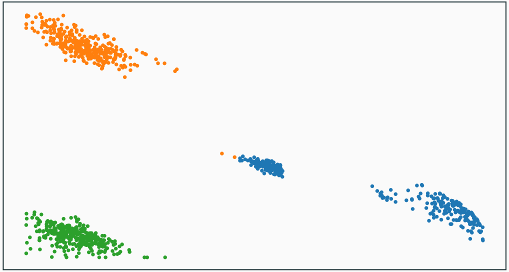
MNIST
RMSE
0.8
0.6
0.4
0.2
2,000
4,000
6,000

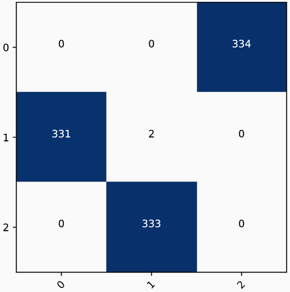


MNIST
Neg. loglik.
2,000
1,000
0
2,000
4,000
6,000

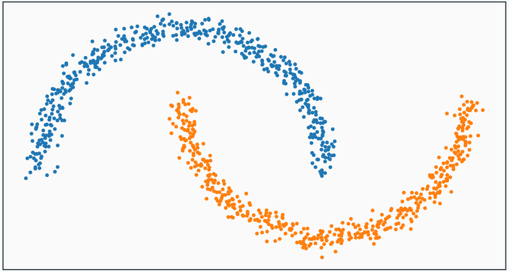
Experiments on Clustering - Blobs

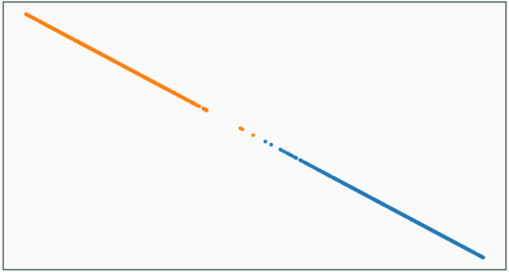


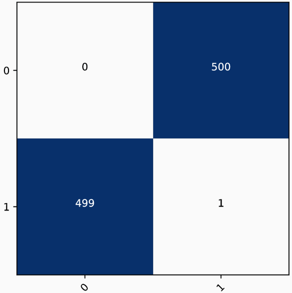


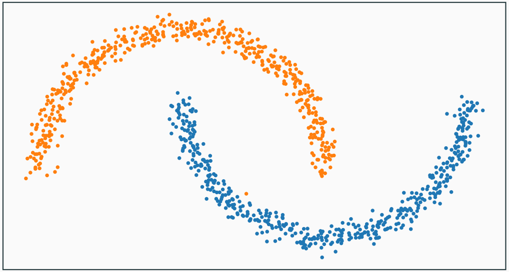


Experiments on Clustering - Moons

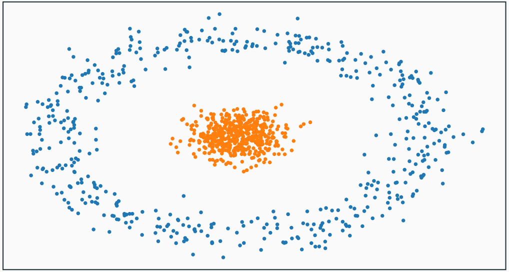


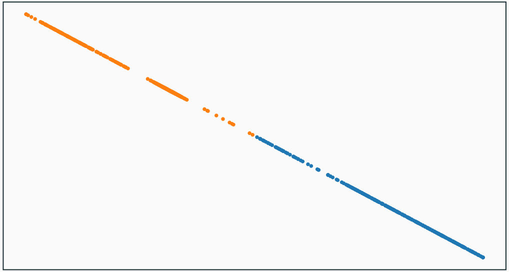


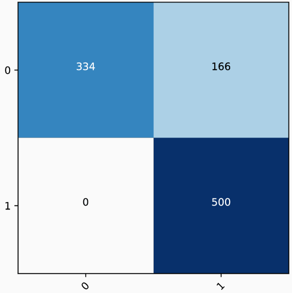


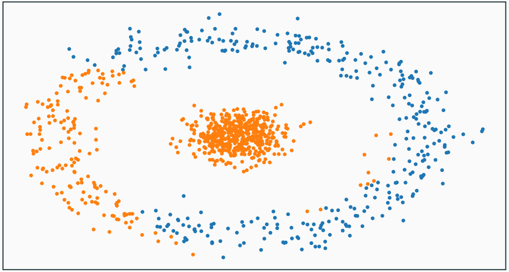


Experiments on Clustering - Circles









Conclusions

Things done i

What has been done so far?

- Migration of existing code from TensorFlow 0.12 to TensorFlow 1.1
- Extension for DGPLVM for both dimensionality reduction and clustering in a new dedicated class
- Experiments on dimensionality reduction and clustering with both real and synthetic datasets

Things done ii

Problems spotted

- Due to TensorFlow computational graph's engine, it's impossible to optimize over `placeholders`, making the minibatch-based learning not straightforward
- The resulting model seems to be too much sensible to initialization pf hyperparameters (in particular, the `lengthscale`)

Things to be done

Future works

- Scalability extension through the use of minibatch-based learning
- Further experiments on clustering with real dataset and some comparisons with state-of-the-art algorithms.

Questions?

Angelova, A., Krizhevsky, A., and Vanhoucke, V.
Real-time pedestrian detection with deep network cascades.
Paper from Google.

Dalal, N., and Triggs, B.
Histograms of oriented gradients for human detection.
Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (2005).

Bibliography ii

Felzenszwalb, P. F., Girshick, R. B., McAllester, D., and Ramanan, D.
Object detection with discriminatively trained part-based models.
IEEE transactions on pattern analysis and machine intelligence 32, 9 (September 2010), 1627–1645.

Ouyang, W., Zeng, X., and Wang, X.
Partial occlusion handling in pedestrian detection with a deep model.
IEEE transactions on circuits and systems for video technology 26, 11 (November 2016), 2123–2137.

Tome, D., Monti, F., and Baroffio, L.
Deep convolutional neural networks for pedestrian detection.
Elsevier Journal of Signal Processing: Image Communication (2016).