[Re] Gaussian Process Latent Variable Models for Visualisation of High Dimensional Data

Group 56

Carles Balsells Rodas
carlesbr@kth.se

Oleguer Canal Anton
oleguer@kth.se

Núria Marzo i Grimalt
nuriamig@kth.se

Federico Taschin
taschin@kth.se

Abstract

In this paper we attempt to develop a re-implementation of the Gaussian Process Latent Variable Model (GPLVM) algorithm. This consist on a formulation for Principal Component Analysis (PCA) which allows nonlinear mapping on higher dimensional spaces of the observable features. Similarly to the original work regarding this method [14], we evaluate the performance of our latent variable model as an approach for the visualization of high dimensional data in a variety of complex datasets. Overall, we see that in comparison to other latent variable decomposition methods, such as linear PCA or kernel PCA, our implementation of the GPLVM yields slightly better results. However, for our implementation of GPLVM, the computational time was much higher than PCA.

1 Introduction

1.1 Description

This work further develops on the Lawrence et al. papers [14] and [12]. We first replicate the algorithm and evaluate it on the same datasets as the original author. In addition, we apply it to new data and provide a more extensive assessment of its performance against similar methods such as PCA and Kernel PCA.

The original formulation of the Gaussian Process Latent Variable Model (GPLVM) can be regarded as an extension of the Probabilistic Principal Component Analysis (PPCA) introduced by Tipping and Bishop [16] some years earlier. Thus, the GPLVM consists on the introduction of the *kernel trick* intuition in this probabilistic reformulation of PCA, that leads to a decomposition of the data represented in higher dimensional spaces.

1.2 Related work

The original formulation shows promising results when applying it to data represented in high dimensional spaces, such as images [14]. However, the GPLVM framework can also be applied to more complex domains: human pose estimation [5], facing the task of determining the physical location of a mobile device from wireless signal strengths [6], etc. Although the GPLVM approach is considered to be used as a form of unsupervised learning, we can find contributions where they try to extend it to the field of supervised learning [7].

In this document, we will see that, although this first approach of GPLVM shows us promising results when dealing with high dimensional spaces, it has some drawbacks regarding computational costs and convergence. In order to overcome these issues, several extensions have been proposed in the recent years. We can find many prominent contributions that boost GPLVM performance in different ways: the Shared GPLVM (SGPLVM) which aims to learn a discriminative manifold shared by multiple input data [4], the introduction of a hierarchical approach to express conditional independencies in the data [15], a Bayesian approach so as to estimate the posterior distribution using variational inference [17], and finally a proposal of another variational inference procedure presenting a spike and slab prior [3].

1.3 Outline

The scope and objectives of the project are the following:

1. Formulate the GPLVM algorithm, self-implement it and optimize it with the informative vector machines.
2. Reproduce the results of the original paper on the oil and digits dataset and apply the algorithm to a similar dataset (genes of mice dataset).
3. Reproduce the results of the original paper on the faces dataset and apply the algorithm to a similar dataset (video showcase).
4. Evaluate and compare the GPLVM method with others like Kernel PCA on the selected datasets.
5. Quantitatively compare the different algorithms using a performance evaluation protocol.

2 Methods

2.1 Formulation

2.1.1 Gaussian Process Latent Variable Models (GPLVMs)

As mentioned in previous sections: GPLVMs can be understood as the dual formulation of PPCA. Given a set of high dimensional data $\{Y_n\}_n$, both methods attempt to provide a lower dimensional projection $\{X_n\}_n$ with the following difference: PCA achieves it by first marginalizing over the latent variables $\{X_n\}_n$ and later optimizing the transformation parameters W. Instead, GPLVM approach first marginalizes over the projection parameters W and then optimizes over both the latent variables $\{X_n\}_n$. It also allows for non-linear transformations through the use of kernels, which also optimizes its parameters. To do so, the following formulation is proposed [14].

First choose a prior over the parameters:

$$Pr(W) = \prod_{i=1}^{D} \mathcal{N}(w_i|0, \alpha^{-1}I) \tag{1}$$

Then, we have that the data follows:

$$Pr(Y|X,\theta) = \frac{\exp\left(-\frac{1}{2}tr\left(K^{-1}YY^T\right)\right)}{(2\pi)^{\frac{D\cdot N}{2}}|K|^{\frac{D}{2}}} \tag{2}$$

Where K refers to the matrix from the chosen kernel, N the number of data samples and D their dimension. As usual, we will work with the negative log-likelihood, which in this case is:

$$-L = \frac{D\cdot N}{2}\log(2\pi) + \frac{D}{2}\log|K| + \frac{1}{2}tr\left(K^{-1}YY^T\right) \tag{3}$$

This expression could already be used to find the projection in latent space X. Nevertheless, for computation limitations these optimization can be decoupled and performed in a two-step iterative manner as follows:

- **Kernel Optimization:** Optimize equation 3 w.r.t kernel parameters using a subset of the available data provided by 2.1.2 called active set.
- **Latent Variable Optimisation:** Taking a new active set following 2.1.2, from equation 1 we can take:

  $$Pr(y_j|x_j,\theta) = \mathcal{N}\left(y_j|f_j,\sigma_j^2 I\right) \tag{4}$$

  Where $f_j = Y_I^T K_{II}^{-1} k_{Ij}$, $\sigma^2 = kernel(x_j,x_j) - k_{Ij}^T K_{II}^{-1} Y_{Ij}$, Y_I is the active subset of data, K_{II} the kernel matrix from the active latent variables and k_{Ij} j-th column of active set rows from kernel matrix. We can thus, optimize over each latent x_j from the inactive set maximizing this probability.

Notice how PCA can be understood as a case of GPLVM where the process prior is based on the inner product of matrix X.

2.1.2 Informative Vector Machine

Each evaluation of Eq. 3 implies computing $K^{-1}YY^T$. Since K is a n x n matrix, its inversion has complexity O(n^c) with $c > 2$ depending on the chosen inversion algorithm. The same applies for the computation of $K^{-1}YY^T$, and therefore each evaluation of L has complexity O(n^c) with $c > 2$. For big datasets its therefore critical to reduce it to become tractable.

Informative Vector Machines [10] can be applied to Gaussian Process regression tasks [13] in order to obtain a representative subset, called the active set, of size d of the dataset. This allows us to obtain a sparse representation of the GP posterior in O(d^2N) and perform optimization of kernel parameters, active set, and inactive set — points not in the active set— separately, reducing the computation cost of kernel optimization to O(d^3).

In IVM, points are selected greedily by the change of entropy given by their inclusion in the active set in the posterior process. Therefore, we build an active set of size d by incrementally insert a point from the inactive set that induces the highest ΔH as described by Lawrence and Platt [13].

2.2 Implementation

Full implementation can be found in this repo. Nevertheless, here are the main steps needed for an efficient computation of the projection:

Algorithm 1: Efficient GPLVM Computation

```
Given: N observations of D dimensions (Y), an active set size d, and a max number of iterations T
Compute first approximation of the latent projection X using PCA on Y
for T iterations do
    Select active set I using IVM algorithm described in section 2.1.2
    Perform kernel parameters optimization of eq. 3 using only active set data as described in 2.1.1.
    Select new active set I using IVM algorithm described in section 2.1.2
    for Each point in the inactive set J do
        Perform latent variable optimization maximizing 4 as described in 2.1.1.
    end
end
```

3 Experiments

This section describes the experiments that we have performed regarding the performance of our implementation of GPLVM in comparison to other latent variable decomposition methods. We have selected 5 different data sets.

3.1 Oil dataset

The data used for this experiment is with the oil flow dataset [2]. This dataset consist on 1000 12-dimensional data points that correspond to the phase of flow in an oil pipeline: stratified, annular and homogeneous. In Figure 1 we can see the plots of the PCA, Kernel PCA and GPLVM of a subset of 200 data points of the oil dataset. Each color represents a class in the dataset. For speeding-up purposes, we have resized the input images to 60x90.

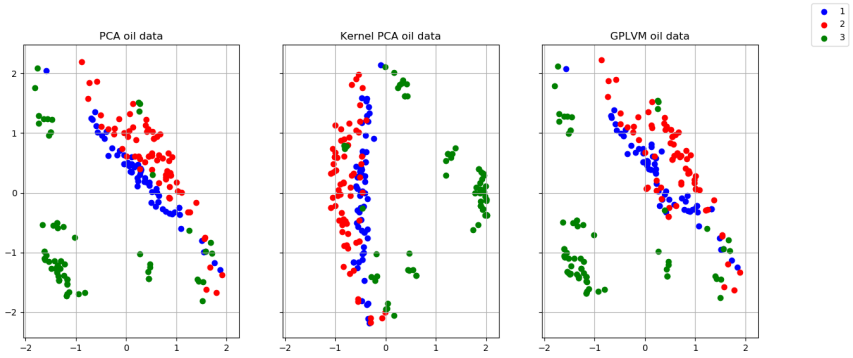


Figure 1: PCA versus KernelPCA versus GPLVM results on a subset of 200 samples of the oil dataset. The kernel used is radius base function.

3.2 Digits

The data used for this experiment is the USPS Digits Dataset [11]. It consist on a set images (16x16 pixels) of handwritten digits from 0 to 9. We used a subset of 500 digits to perform the experiment and in Figure 2 there are the plots of the PCA results compared to the GPLVM results (using a polynomial kernel of degree 5).

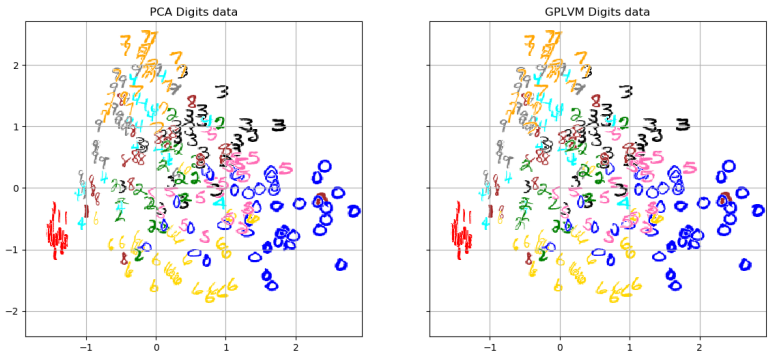


Figure 2: PCA versus GPLVM results on a set of the 500 samples of the digits dataset.

3.3 Genes

The data used in this experiment is extracted from [9] where they analyzed 48 genes from 437 cells of mice, so the data has 437 points of 48 dimensions each. The cells are divided into 10 different types. In Figure 3 there is a plot of the PCA and GPLVM results from the genes dataset and each color represents a different type of cell.

3.4 Video showcase

This experiment is inspired by the original work of GPLVM [14] in order to recreate a similar setting than the one for the *Fantasy faces* dataset. Unfortunately, we were not able to find such dataset, but we will

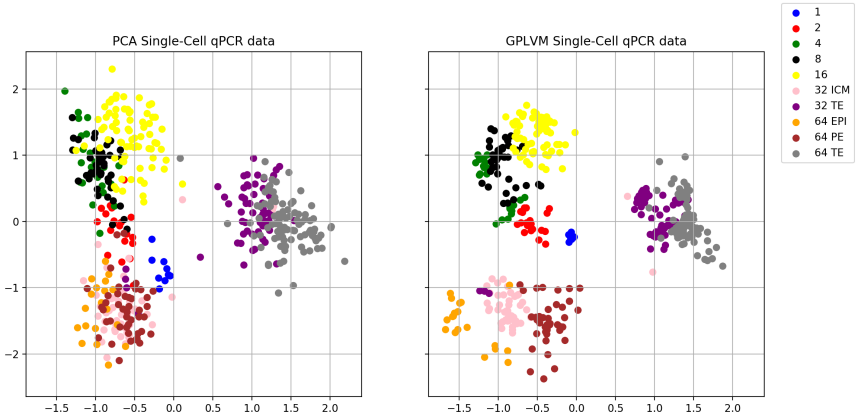
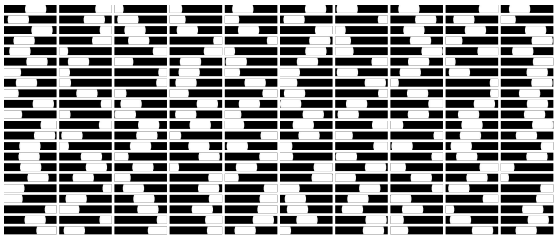


Figure 3: PCA versus GPLVM results on the 437 samples of the genes dataset.

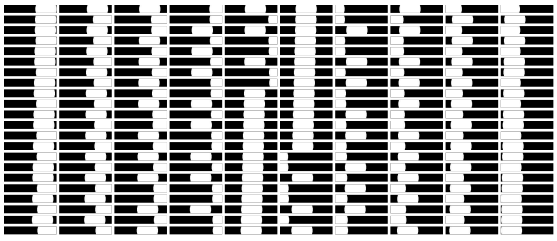
construct a similar version that fits our purposes. The one which we have recreated consists of a video composed by 220 frames, and each frame is an image of size 200x30.

The objective is to test if the latent variables resulting from a decomposition gives us information about the causality in a video sequence. Therefore, we will generate some video data where we expect to infer some causal behaviour between frames so that the decomposition into a single latent variable gives information about the ordering of each frame in the sequence. The video case that we study consists of a white ball that moves from left to right in a black background.

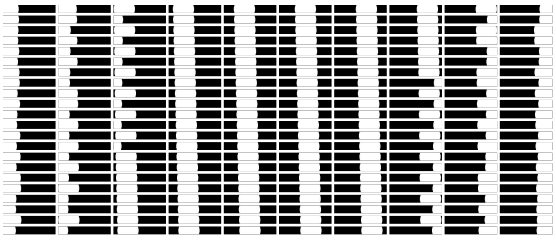
The decomposition of the latent variables is performed over the frames after being randomly shuffled. An example of the input forwarded to the algorithm is shown in figure 4a. Figures 4b, 4c and 4d show the results when performing the decomposition using a linear PCA, a kernel PCA with a sigmoid kernel and $\gamma = 15$, and our implementation of the GPLVM respectively. For qualitative evaluation purposes, the ordering of the frames goes from top to bottom and from left to right. We only show the results of the kernel PCA as using other kernels gives worse results.



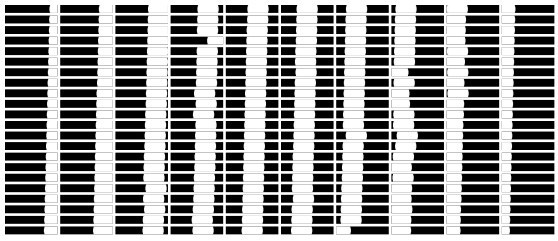
(a) Input data.



(b) Linear PCA.



(c) Kernel PCA with a sigmoid kernel ($\gamma = 15$).



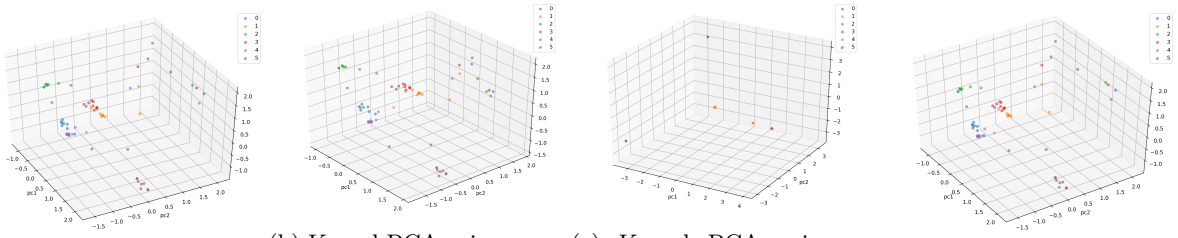
(d) GPLVM.

Figure 4: Results on our video data example showcase. The ordering of the sequences of frames can be understood by reading them from top to bottom, and from left to right.

3.5 Yale Face Database

The purpose of this section is to show the performance of our implementation of the GPLVM in combination with the IVM in the context of complex domplex domains. In this experiment, we are facing the task of face recognition. To do so, we will use a subset of the well-known Yale Face Database [8]. This database contains 165 grayscale images in GIF format of 15 individuals. There are 11 images per subject, one per different facial expression or configuration: center-light, w/glasses, happy, left/right-light, etc.

We will perform our experiments on a subset of the whole data set, only considering 6 different subjects. In order to account for the performance of our approach, we will test it in comparison to a linear PCA, a kernel PCA that uses a RBF kernel and a cubic one. Figure 5 shows the results of the decomposition of the input data of the subjects that we have considered into three principal components, so that we can perform a qualitative evaluation of the results.



(a) Linear PCA.

(b) Kernel PCA using a cubic kernel.

(c) Kernel PCA using a RBF kernel with $\gamma = 1$.

(d) GPLVM.

Figure 5: Results on the subset of Yale Face Database.

4 Discussion

Quantitative performance evaluation protocol. For each tested dataset and different kernel we compare the quality of the results with PCA, KernelPCA, and GPLVM. A frequent situation in which these methods are applied is in unsupervised learning, where the classes are unknown. The quality of a dimensionality reduction is therefore related on how well it groups together points belonging to the same class in the latent space, without knowing their class. We thus compute the latent space projection of variables for all these algorithms and perform a Gaussian Mixture Models clustering for a number of clusters equal to the number of classes. Each cluster k is assigned to a class Ck by a majority vote of the real classes of its points. We compute the confusion matrix and, for each class, the precision and recall. The metrics we use to compare the algorithms are therefore the average precision and recall over the classes in the dataset. Precision and recall measures are defined following the standard definitions [1]. An example of this clustering evaluation process on gene's dataset can be seen in figure 6.

In section 3.1, 3.2 and 3.3 we compare the PCA algorithm and the GPLVM algorithm in similar datasets. The oil and the digits are the same used in Lawrence's paper [14] and we added a new dataset from [9] about genes in mice cells.

In the results of the oil dataset showed in Figure 1 we compared the PCA, the Kernel PCA and the GPLVM algorithms performance. All the algorithms manage to separate the clusters in a two-dimensional latent space but we can see some difference in the separation of the class 1 and class 2 whereas the class 3 is always very differentiated.Qualitatively, the results seem better than the ones presented by Lawrence's paper [14] as we can see a more clear separation of the classes but the difference between PCA and GPLVM is minimal. The metrics of the algorithm are showed in Table 1 and we can see that the PCA is the one with best results followed very closely by the Kernel PCA and GPLVM.

For the digits dataset the results are showed in Figure 2. Similarly to the oil dataset, we can seen that the difference between the PCA and the GPLVM results is minimal. The results are not as good as in [14] paper. The main reasons can be that in their paper they use 10 times more data points than we did, also the digit images are very complex data and with only 200 samples the algorithm does as good as it can. The metrics of the algorithm are showed in Table 2 and we can see that the GPLVM is the one that gives best results. Nevertheless, they are very similar to the PCA ones.

In addition, we evaluated the result on a different dataset that the ones used in the Lawrence's paper [14]. The genes dataset gave us very interesting and good results, specially when we compare the PCA results and the GPLVM results showed in Figure 3. In the plots we can see how good the separation of the clusters is. An example of how good was the GPLVM is with the classes "32 ICM", "64 EPI" and "64 PE". If we compare the separation of the three classes in linear PCA and GPLVM we can see that GPLVM's results are much better. The metrics of the algorithm are showed in Table 3 and we can see that the GPLVM is the one that gives best results with a significance different between the two algorithms.

In section 3.4 we presented the case of a video decomposition so as to see if our model is able to capture the causal relations between frames. As shown in figure 4d, we can see that GPLVM has successfully sorted the majority of the video frames. The result is a video sequence of a ball that is translated along the image smoothly, except for a small number of frames. By inspection of the rest of the results (for both PCA and kernel PCA) shown in figures 4b and 4c we observe that GPLVM outperforms both the rest of the methods. Furthermore, notice that the ordering of the frames obtained using the kernel PCA are

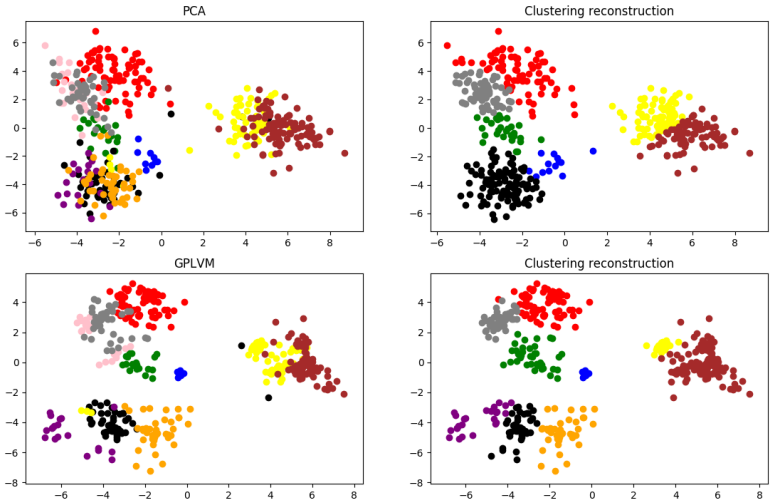


Figure 6: PCA and GPLVM latent space projections (left) and corresponding GMM clustering reconstruction (right) for genes dataset

flipped along the vertical axis. This result should be expected as the models have no prior information of the initial point of the ball (left side or right side).

Finally, we have explored the performance of GPLVM considering more high dimensional datasets (see Sec. 3.5). As it was indicated previously, we only consider 6 different faces in our setting be able to give a better evaluation by inspection of the figures (see fig. 5). In general, it can be observed that the different methods that we have considered perform similarly. However, we can see that the decomposition performed by the kernel PCA using a RBF kernel yields very bad results, as almost of the data points are located at the zero position. The reason why we tested the usage of this kernel is because we use it in our implementation of the GPLVM, which surprisingly shows much better results. For the rest of the cases, we can see that the majority of the input data is separated by means of the face, except for some outliers. In figure 7, we show that in fact these samples correspond to the same category of a given face (right-light and left-light). In order to provide a quantitative evaluation perspective, we have performed the evaluation protocol described in the beginning of this section (see table 4). We can see that our implementation of the GPLVM gives the best results according to both of the metrics that we have defined.

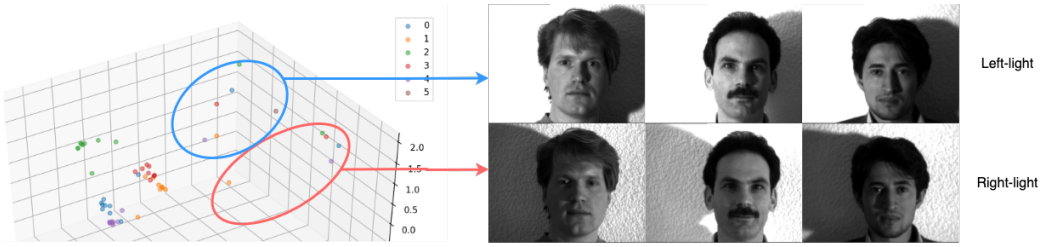


Figure 7: Outliers encountered in the latent variable decompositions that have been performed.

5 Conclusions

We have presented a successful re-implementation and amplification the the Gaussian Process Latent Variable Models for Visualisation of High Dimensional Data paper by Lawrence et al [14]. We have been able to re-implement the algorithm and the optimization techniques such as IVMs. We have tested the latent variable model in many different datasets, some used in the original paper and some others that we have found of interest. We have obtained good results, with GPLVM performing better in almost all datasets. However, in our implementation, GPLVM was much slower: to compute the latent space of 500 digits (256 dimensions) GPLVM took 41 seconds, while PCA took less then a second. In many applications this is not desirable, and the improvement in our evaluation metrics would most likely be not enough to justify it.

Algorithm	Avg precision	Avg recall
PCA	0.**2220**	0.**2000**
Kernel PCA RBF	0.2147	0.1900
GPLVM	0.2144	0.1950

Table 1: Oil Dataset

Algorithm	Avg precision	Avg recall
PCA	0.0416	0.0526
GPLVM	0.**0483**	**0.0532**

Table 2: Digits Dataset

Algorithm	Avg precision	Avg recall
PCA	0.0565	0.0641
GPLVM	**0.0753**	**0.0764**

Table 3: Genes Dataset

Algorithm	Avg precision	Avg recall
PCA	0.0897	0.1010
PCA RBF $\gamma = 1$	0.1163	0.0479
PCA Cubic	0.1187	0.1060
GPLVM	**0.1211**	**0.1187**

Table 4: Yale Faces

We developed a quantitative evaluation performance protocol to compare the results. An "eye" evaluation cannot be considered acceptable, especially for complex datasets such as the oil flow. Moreover, visually evaluating the quality of the latent space is too susceptible to interpretation biases. By comparing the performances of the latent space generated by GPLVM and other techniques in a real-world situation such unsupervised clustering we provide a quantitative evaluation of these techniques. We believe this approach should be further investigated.

References

[1] Precision and recall definition. `https://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html`.

[2] Christopher M Bishop and Gwilym D James. Analysis of multiphase flows using dual-energy gamma densitometry and neural networks. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 327(2-3):580–593, 1993.

[3] Zhenwen Dai, James Hensman, and Neil Lawrence. Spike and slab gaussian process latent variable models. *arXiv preprint arXiv:1505.02434*, 2015.

[4] Carl Henrik Ek, Peter Jaeckel, Neill Campbell, Neil Lawrence, and Chris Melhuish. Shared gaussian process latent variable models for handling ambiguous facial expressions. *AIP Conference Proceedings*, 1107, 03 2009.

[5] Carl Henrik Ek, Philip HS Torr, and Neil D Lawrence. Gaussian process latent variable models for human pose estimation. In *International workshop on machine learning for multimodal interaction*, pages 132–143. Springer, 2007.

[6] Brian Ferris, Dieter Fox, and Neil D Lawrence. Wifi-slam using gaussian process latent variable models. In *IJCAI*, volume 7, pages 2480–2485, 2007.

[7] Xinbo Gao, Xiumei Wang, Dacheng Tao, and Xuelong Li. Supervised gaussian process latent variable model for dimensionality reduction. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 41(2):425–434, 2010.

[8] A Georghiades, P Belhumeur, and D Kriegman. Yale face database. *Center for computational Vision and Control at Yale University, http://cvc. yale. edu/projects/yalefaces/yalefa*, 2(6):33, 1997.

[9] Guoji Guo, Mikael Huss, Guo Qing Tong, Chaoyang Wang, Li Li Sun, Neil D Clarke, and Paul Robson. Resolution of cell fate decisions revealed by single-cell gene expression analysis from zygote to blastocyst. *Developmental cell*, 18(4):675–685, 2010.

[10] Ralf Herbrich, Neil D Lawrence, and Matthias Seeger. Fast sparse gaussian process methods: The informative vector machine. In *Advances in neural information processing systems*, pages 625–632, 2003.

[11] Jonathan J. Hull. A database for handwritten text recognition research. *IEEE Transactions on pattern analysis and machine intelligence*, 16(5):550–554, 1994.

[12] Neil Lawrence. Probabilistic non-linear principal component analysis with gaussian process latent variable models. *Journal of machine learning research*, 6(Nov):1783–1816, 2005.

[13] Neil Lawrence and John Platt. Learning to learn with the informative vector machine. 09 2004.

[14] Neil D Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In *Advances in neural information processing systems*, pages 329–336, 2004.

[15] Neil D Lawrence and Andrew J Moore. Hierarchical gaussian process latent variable models. In *Proceedings of the 24th international conference on Machine learning*, pages 481–488. ACM, 2007.

[16] Michael E Tipping and Christopher M Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, 1999.

[17] Michalis Titsias and Neil D Lawrence. Bayesian gaussian process latent variable model. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 844–851, 2010.