Discriminative Gaussian Process Latent Variable Model for Classification

Raquel Urtasun RURTASUN@CSAIL.MIT.EDU
Trevor Darrell TREVOR@CSAIL.MIT.EDU
Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology

Abstract

Supervised learning is difficult with high dimensional input spaces and very small training sets, but accurate classification may be possible if the data lie on a low-dimensional manifold. Gaussian Process Latent Variable Models can discover low dimensional manifolds given only a small number of examples, but learn a latent space without regard for class labels. Existing methods for discriminative manifold learning (e.g., LDA, GDA) do constrain the class distribution in the latent space, but are generally deterministic and may not generalize well with limited training data. We introduce a method for Gaussian Process Classification using latent variable models trained with discriminative priors over the latent space, which can learn a discriminative latent space from a small training set.

1. Introduction

Conventional classification methods suffer when applied to problems with high dimensional input spaces, very small training sets, and no relevant unlabeled data. If, however, the high dimensional data in fact lie on a low-dimensional manifold, accurate classification may be possible with a small amount of training data if that manifold is discovered by the classification method. Existing techniques for discovering such manifolds for discriminative classification are generally deterministic and/or require a large amount of labeled data. We introduce here a new method that learns a discriminative probabilistic low dimensional latent space.

We exploit Gaussian Processes Latent Variable Models, which can discover low dimensional manifolds in high dimensional data given only a small number of examples (Lawrence, 2004). Such methods have been developed to date in a generative setting for visualization and regression applications, and learn a latent space without regard for class labels[1]. These models are not “discriminative”; nothing in the GPLVM encourages points of different classes to be far in latent space, especially if they are close in data space, or discourages points of the same class from being far in latent space if they are far in input space. As a result, the latent space is not optimal for classification.

In contrast, discriminative latent variable methods, such as Linear Discriminant Analysis (LDA), and its kernelized version Generalized Discriminant Analysis (GDA), try to explicitly minimize the spread of the patterns around their individual class means, and maximize the distance between the mean of the different classes. However, these methods are generally not probabilistic and may not generalize well with limited training data.

In this paper, we develop a discriminative form of GPLVM by employing a prior distribution over the latent space that is derived from a discriminative criterion. We specifically adopt GDA constraints but the proposed model is general and other criteria could also be used. Our model has the desirable generalization properties of generative models, while being able to better discriminate between classes in the latent space.

Gaussian Process Classification (GPC) methods have seen increasing recent interest as they can accurately and efficiently model class probabilities in many recognition tasks. Since generalization to test cases inherently involves some level of uncertainty, it is desirable to make predictions in a way that reflects these uncertainties (Rasmussen & Williams, 2006). In general, GPC is defined by a covariance function, and one must optimize this function (i.e. hyper-parameters) with respect to the best classification rates or class probabilities (i.e. confidence). In the standard GPC formulation, the only freedom for the covariance to become discriminative is in the choice of the value of its hyper-parameters. Here, we will show that the covariance matrix estimated by a discriminative GPLVM dramatically improves GPC classification when the training data

Appearing in *Proceedings of the 24^{th} International Conference on Machine Learning*, Corvallis, OR, 2007. Copyright 2007 by the author(s)/owner(s).

[1] GPLVMs can be considered as a generalization of Probabilistic PCA to the non-linear case.

is small, even when the number of examples is smaller than the dimensionality of the data space.

Several authors have proposed methods to take advantage of the low dimensional intrinsic nature of class labeled data. (Iwata et al., 2005) proposed Parametric Embedding (PE), a technique based on Stochastic Neighbor Embedding (SNE) (Hinton & Roweis, 2002), to simultaneously embed objects and their classes. This was extended to the semi-supervised case by modeling the pairwise relationships between the objects and the embedding (Zien & Quiñonero-Candela, 2005). But these methods do not work in practice when the training set is small. Probably the closest work to ours is the covariance kernels proposed by Seeger (Seeger, 2001), where a Bayesian mixture of factor analyzers is used for semi-supervised classification. The formalism we propose is different and works well with no unlabeled data and relatively few training examples.

In following sections we review GPC and GPLVM, introduce discriminative GPLVM, and present the use of discriminative GPLVM in the context of GPC. We then show comparative results on a variety of datasets which demonstrate significantly improved performance when the amount of training data is limited. We finally discuss extensions of our method to semi-supervised tasks, and to different discriminative criteria.

2. Gaussian Process for Classification

In this section we review the basics of Gaussian Processes for Binary Classification. Since the classification problem (i.e. the probability of a label given an input) cannot be directly modeled as a Gaussian Process, in GPC a latent function is introduced. A Gaussian Process (GP) prior is placed over the latent functions, and their results are "squashed" through a logistic function to obtain a prior on the class probabilities (given the inputs).

More formally, let $\mathbf{Y} = [\mathbf{y}_1, ..., \mathbf{y}_N]^T$ be a matrix representing the input data and $\mathbf{Z} = [z_1, ..., z_N]^T$ denote the vector representing the labels associated with the training data, where $z_i \in \{-1, 1\}$ denotes the class label of input $\mathbf{y}_i$. Gaussian process classification (GPC) discriminately models $p(\mathbf{z}|\mathbf{y})$ as a Bernouilli distribution. The probability of success is related to an unconstrained intermediate[2] function, $f_i = f(\mathbf{y}_i)$, which is mapped to the unit interval by a sigmoid function (e.g. logit, probit) to yield a probability. Let $\mathbf{f} = [f_1, ..., f_N]^T$ be the values of the intermediate function. The joint likelihood factorizes to

$$p(\mathbf{Z}|\mathbf{f}) = \prod_{i=1}^{N} p(\mathbf{z}_i|f_i) = \prod_{i=1}^{N} \Phi(\mathbf{z}_i f_i) \ , \tag{1}$$

where Φ is the sigmoid function. Following (Rasmussen & Williams, 2006) we use a zero-mean Gaussian Process prior over the intermediate functions f with covariance $k(\mathbf{y}_i, \mathbf{y}_j)$. The posterior distribution over latent functions becomes

$$p(\mathbf{f}|\mathbf{Z}, \mathbf{Y}, \theta) = \frac{\mathcal{N}(\mathbf{f}|0, \mathbf{K})}{p(\mathbf{Z}, \mathbf{Y}|\theta)} p(\mathbf{Z}|\mathbf{f}) \tag{2}$$

with

$$p(\mathbf{Z}, \mathbf{Y}|\theta) = \int p(\mathbf{Z}|\mathbf{f}) p(\mathbf{f}|\mathbf{Y}, \theta) d\mathbf{f} \ , \tag{3}$$

where $K_{ij} = k(\mathbf{y}_i, \mathbf{y}_j)$, and θ are the hyper-parameters of the covariance function k. Unlike the regression case, neither the posterior, the marginal likelihood $p(\mathbf{Z}|\mathbf{f})$, nor the predictions can be computed analytically. A discriminative GPC either approximates the posterior with a Gaussian, or employs Markov chain Monte Carlo sampling. In this paper we take the former approach, and use the Laplace and Expectation Propagation (EP) methods. For a detailed description of such methods, and a comparison between them, we refer the reader to (Rasmussen & Williams, 2006; Kuss & Rasmussen, 2006).

The functional form of the covariance function k encodes assumptions about the intermediate function. For example, one might use a Radial Basis Function (RBF) if we expect the latent function to be smooth. When doing inference, the hyper-parameters of the covariance function have to be estimated, choosing them so that the covariance matrix is as "discriminative" as possible. But not many degrees of freedom are typically left for the covariance to be discriminative. For example, in the case of an RBF, only two hyper-parameters are estimated: the support width and the output variance.

In theory, one could optimize the whole covariance, but this is unfeasible in practice as it requires N^2 parameters to be estimated, subject to the constraint that the covariance matrix has to be positive definite. In the following section we review GPLVM models, which provide a covariance function with a significantly richer parameterization than typical hyper-parameters, yet which is sufficiently constrained to allow estimation.

3. Gaussian Process Latent Variable Model (GPLVM)

Let $\mathbf{Y} = [\mathbf{y}_1, ..., \mathbf{y}_N]^T$ be a matrix representing the training data, with $\mathbf{y}_i \in \Re^D$. Similarly, let $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_N]^T$

[2] We are using the term intermediate function rather than latent function here, to avoid confusion with the latent variable space in GPLVM.

denote the matrix whose rows represent corresponding positions in latent space, $\mathbf{x}_i \in \Re^d$. The Gaussian Process Latent Variable Model relates a high-dimensional data set, $\mathbf{Y}$, and a low dimensional latent space, $\mathbf{X}$, using a Gaussian process mapping from the latent space to the data space. Given a covariance function for the Gaussian process, $k_Y(\mathbf{x}, \mathbf{x}')$, the likelihood of the data given the latent positions is,

$$p(\mathbf{Y} \,|\, \mathbf{X}, \bar{\beta}) \;=\; \frac{1}{\sqrt{(2\pi)^{ND}|\mathbf{K}_Y|^D}} \exp\left(-\frac{1}{2}\mathrm{tr}\left(\mathbf{K}_Y^{-1}\mathbf{Y}\mathbf{Y}^T\right)\right), \quad (4)$$

where elements of the kernel matrix $\mathbf{K}_Y$ are defined by the covariance function, $(\mathbf{K}_Y)_{i,j} = k_Y(\mathbf{x}_i, \mathbf{x}_j)$. We use a kernel that is the sum of an RBF, a bias or constant term, and a noise term.

$$k_Y(\mathbf{x}, \mathbf{x}') \;=\; \theta_1 \exp\left(-\frac{\theta_2}{2}||\mathbf{x} - \mathbf{x}'||^2\right) + \theta_3 + \frac{\delta_{\mathbf{x},\mathbf{x}'}}{\theta_4}, \quad (5)$$

where $\theta = \{\theta_1, \theta_2, ...\}$ comprises the kernel hyperparameters that govern the output variance, the RBF support width, the bias, and the variance of the additive noise, respectively. The posterior can be written as

$$p(\mathbf{X}, \bar{\beta} \,|\, \mathbf{Y}) \;\propto\; p(\mathbf{Y} \,|\, \mathbf{X}, \bar{\beta})\, p(\mathbf{X})\, p(\theta)\,. \quad (6)$$

Learning in the GPLVM consists of minimizing the log posterior with respect to the latent space configuration, $\mathbf{X}$, and the hyper parameters, θ,

$$\mathcal{L} \;=\; \mathcal{L}_r \;+\; \sum_i \ln\theta_i \;+\; \sum_i \frac{1}{2}||\mathbf{x}_i||^2, \quad (7)$$

where we have introduced uninformative priors over the kernel hyper-parameters, and simple priors over the latent positions. These priors prevent the GPLVM from placing latent points infinitely far apart, i.e. latent positions close to the origin are preferred. The log likelihood associated with (4) is,

$$\mathcal{L}_r \;=\; \frac{D}{2}\ln|\mathbf{K}_Y| \;+\; \frac{1}{2}\mathrm{tr}\left(\mathbf{K}_Y^{-1}\mathbf{Y}\mathbf{Y}^T\right) \quad (8)$$

A key property of the model is its use of (closed form) Bayesian model averaging (Lawrence, 2004), both to mitigate problems due to over-fitting with small data sets, and to remove the need to select parameters of the function approximators.

To preserve topological structure, the back-constrained GPLVM (Lawrence & Quiñonero-Candela, 2006) constrains the latent positions to be a smooth function of the data space. As a result, points that are close in data space will be close in latent space.

However, the GPLVM (back-constrained or not) is purely generative: nothing in the model encourages latent positions of different classes to be far, nor latent positions of the same class to be close. In the following section we propose the discriminative GPLVM to address this limitation. The discriminative GPLVM explicitly models the intrinsic low-dimensional representation of the data, resulting in good classification rates, even when the number of training examples is smaller than the input space dimensionality.

4. Discriminative GPLVM

The GPLVM is a generative model of the data, where a simple spherical Gaussian prior is placed over the latent positions (7). In this section we develop a Discriminative Gaussian Process Latent Variable Model, using an informative prior that encourages latent positions of the same class to be close and those of different classes to be far. While several discriminative criterion are possible, we have used a prior based on Generalized Discriminant Analysis (GDA).

4.1. LDA-GDA Revisited

LDA and GDA are discriminative methods which find a transformation that maximizes between-class separability and minimizes within-class variability. This transformation is linear in the LDA and non-linear (kernelized) in the GDA. The transformation projects to a space of dimension at most $L - 1$, where L is the number of classes, and is distribution-free, i.e. no assumption is made regarding the distribution of the data. These techniques are generally combined with a classifier in the low dimensional space.

More formally, let $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_N]^T$ be the desired low dimensional representation of the input data $\mathbf{Y} = [\mathbf{y}_1, ..., \mathbf{y}_N]^T$. LDA and GDA try to maximize between-class separability and minimize within-class variability by maximizing

$$J(\mathbf{X}) = \mathrm{tr}\left(\mathbf{S}_w^{-1}\mathbf{S}_b\right), \quad (9)$$

where $\mathbf{S}_w$ and $\mathbf{S}_b$ are the within- and between- class matrices:

$$\mathbf{S}_w = \sum_{i=1}^{L} \frac{N_i}{N}(\mathbf{M}_i - \mathbf{M}_0)(\mathbf{M}_i - \mathbf{M}_0)^T \quad (10)$$

$$\mathbf{S}_b = \sum_{i=1}^{L} \frac{N_i}{N}\left[\frac{1}{N_i}\sum_{k=1}^{N_i}(\mathbf{x}_k^{(i)} - \mathbf{M}_i)(\mathbf{x}_k^{(i)} - \mathbf{M}_i)^T\right] \quad (11)$$

where $\mathbf{X}^{(i)} = [\mathbf{x}_1^{(i)}, \cdots, \mathbf{x}_{N_i}^{(i)}]$ are the N_i training points of class i, $\mathbf{M}_i$ is the mean of the elements of class i, and $\mathbf{M}_0$ is the mean of all the training points of all classes. In the linear case (LDA) the maximization problem can be solved in closed form. In the non-linear case the kernel "trick" is typically used to obtain a closed form solution.

4.2. Discriminative GPLVM (D-GPLVM)

The LDA-GDA energy function in (9) is a function of the latent positions, $\mathbf{X}$, and can be interpreted as a prior over latent configurations that forces the latent points of the same class to be close together and far from those of other classes.

$$p(\mathbf{X}) = \frac{1}{Z_d} \exp \left\{ -\frac{1}{\sigma_d^2} J^{-1} \right\} , \tag{12}$$

where Z_d is a normalization constant, and σ_d^2 represents a global scaling of the prior.

Learning the D-GPLVM is then equivalent to minimizing

$$\mathcal{L}_S \;=\; \mathcal{L}_r \;+\; \sum_i \ln \theta_i \;+\; \frac{1}{\sigma_d^2} \mathrm{tr}\left(\mathbf{S}_b^{-1} \mathbf{S}_w \right) \;, \tag{13}$$

with $\mathcal{L}_r$ defined in (8). This is minimized using Scaled Conjugate Gradient (SCG) technique. We have replaced the spherical Gaussian prior over the latent positions in (7) with a discriminative prior based on GDA[3]. As the GPLVM, discriminative GPLVM relies on MAP estimates of the kernel hyperparameters and the latent locations.

Note that (13) can be interpreted as a regularized GPLVM, where the regularizer is a discriminative GDA-based criterion, or as a regularized GDA, where the regularizer is a GP. In the limit ($\sigma_d \to 0$), Equation (13) has a closed form solution and is equivalent to GDA. If we use a linear kernel instead of (5) then our method is equivalent to LDA.

The choice of the σ_d reflects a tradeoff between our method's ability to discriminate (small σ_d) and its ability to generalize (large σ_d). Fig. 1 shows models learned with different values of σ_d for the oil database. Fig. 1(i) shows the classification error as a function of $1/\sigma_d^2$ in logaritmic scale. The leftmost point corresponds to classic GPLVM (i.e. $\sigma_d^2 = \infty$) and the rightmost point to $\sigma_d^2 = 10^{-6}$. The latter produces results similar to GDA. In this example as σ_d increases the model becomes less discriminative and has increasing classification error; to achieve minimum error σ_d cannot be ∞, confirming GPLVM is not optimal for classification. When the number of examples is smaller than the dimensionality of the data space the minimum occurs in the middle of the curve, as is shown in Fig. 3 for the USPS database. In general, the optimal value σ_d is a function of the amount of training data and the input space dimensionality. Larger input space dimensionality and smaller amounts of training examples both imply larger values of σ_d should be used, since generalization becomes more important.

5. Discriminative GPLVM for Gaussian Process Classification

Recall that in GPC the intermediate function is modeled with a zero-mean Gaussian Process prior. The discriminative GPLVM learns a covariance that is a function of the low dimensional representation of the training data, $\mathbf{X}$, and the kernel hyper-parameters, θ. The covariance, $\mathbf{K}_Y$, can directly be used for the Gaussian Process prior over the intermediate functions, $\mathbf{K}$ in (2). The kernel matrices obtained by our method are discriminative and more flexible than the ones used in classical GPC, since they are learned based on a discriminative criterion, and more degrees of freedom are estimated than classic kernel hyper-parameters[4].

5.1. Inference with New Test Points

When given a new test point $\mathbf{y}'$ we need to estimate its low dimensional representation $\mathbf{x}'$. This can be done by maximizing $p(\mathbf{y}', \mathbf{x}' | \mathbf{X}, \mathbf{Y}, \theta)$, or equivalently by minimizing its negative log likelihood. This is (up to an additive constant) equal to

$$\mathcal{L}_{inf} = \frac{\|\mathbf{y}' - \mu_Y(\mathbf{x}')\|^2}{2\sigma^2(\mathbf{x}')} + \frac{D}{2} \ln \sigma^2(\mathbf{x}') + \frac{1}{2} \|\mathbf{x}'\|^2 \tag{14}$$

where the mean and variance are given by

$$\begin{aligned} \mu_Y(\mathbf{x}') &= \mu + \mathbf{Y}^T \mathbf{K}_Y^{-1} \mathbf{k}_Y(\mathbf{x}') \,, & (15) \\ \sigma^2(\mathbf{x}') &= k_Y(\mathbf{x}', \mathbf{x}') - \mathbf{k}_Y(\mathbf{x}')^T \mathbf{K}_Y^{-1} \mathbf{k}_Y(\mathbf{x}') \,, & (16) \end{aligned}$$

and $\mathbf{k}_Y(\mathbf{x})$ is the vector with elements $k_Y(\mathbf{x}, \mathbf{x}_j)$ for all other latent positions $\mathbf{x}_j$ in the model, with k_Y as in (5). For inference we use an isotropic spherical prior over the new latent positions, since the class labels of the test data are unknown.

Fast inference To speed up this process, we can exploit GPLVM backconstraints (Lawrence & Quiñonero-Candela, 2006), minimizing (13) subject to

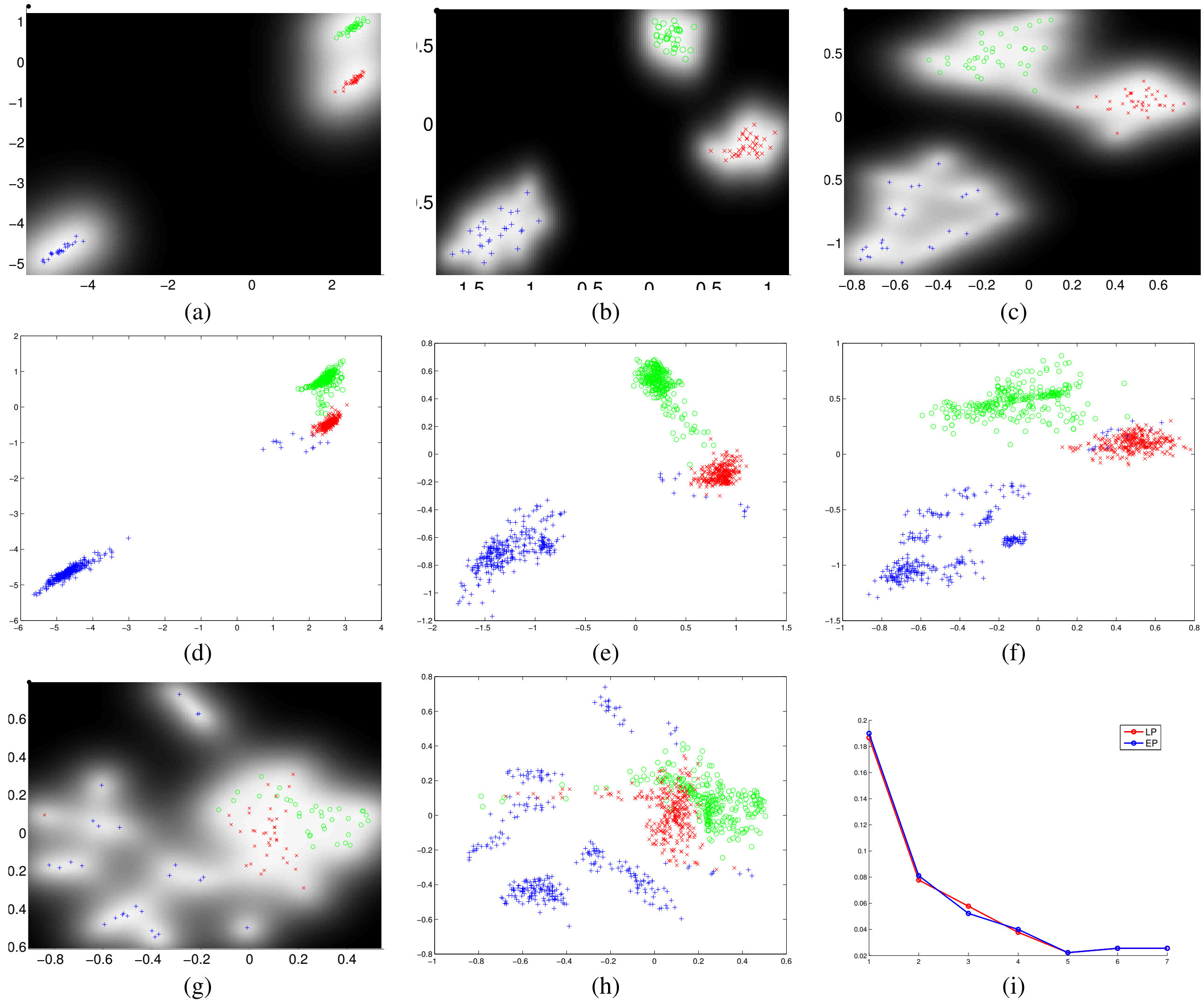
$$x_{ij} = g_j(\mathbf{y}_i; \mathbf{a}) = \sum_{m=1}^{N} a_{jm} k_{bc}(\mathbf{y}_i - \mathbf{y}_m) \,, \tag{17}$$

where x_{ij} is the j-th dimension of $\mathbf{x}_i$. This allows for different backconstraints, g_j, to be used for the different dimensions. In particular, since we want the inverse mapping to be smooth, we use an RBF kernel for the backconstraint in each dimension

$$k_{bc}(\mathbf{y}_i - \mathbf{y}_m) = \exp(-\frac{\gamma}{2} ||\mathbf{y}_i - \mathbf{y}_m||^2) \,. \tag{18}$$

[3] Note that since (9) is a maximization criterion and we are minimizing the log likelihood, we use the inverse of J.

[4] The number of degrees of freedom is the number of hyper-parameters N_θ for GPC and $N_\theta + N * d$ for discriminative GPLVM, with d the dimensionality of the latent space, and N the total number of training points.


(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 1. 2D latent spaces learned by D-GPLVM on the oil dataset are shown, with 100 training examples and different values of σ_d. The grayscale plots (a,b,c,g) show the low dimensional representation of the training examples, and the grayscale represents $-\frac{D}{2}\ln\sigma^2(\mathbf{x}') + \frac{1}{2}\|\mathbf{x}'\|^2$. (d,e,f,h) depict the latent coordinates of the test examples computed using backconstraints. In particular, (a,d) $\sigma_d^2 = 10^{-7}$, (b,e) $\sigma_d^2 = 10^{-5}$, (c,f) $\sigma_d^2 = 10^{-4}$ and (g,h) is equivalent to GPLVM ($\sigma_d^2 = \infty$). (i) depicts the classification error for the Laplace and EP methods as a function of $1/\sigma_d^2$ in logarithmic scale. Note that as $1/\sigma_d^2$ increases the model becomes more discriminative but has worse generalization.

The low dimensional representation of a test point is then obtained by evaluating the inverse mapping learned by the backconstraint at the test point $x'_j = g_j(\mathbf{y}'; \mathbf{a})$. No minimization is required, and $\mathbf{x}'$ can be computed directly for the new test point.

6. Experimental Results

For all databases, we test the performance of our algorithm across varying training set sizes. Each test trial was repeated with varying model parameters $\sigma_d = 10^3, 10^4, 10^5$ and $\gamma = 0.1, 0.01, 0.001$. The setting which resulted in minimum mean error performance over 10 random trials was used. For multi-class problems we used a one-vs-all GPC formulation. The D-GPLVM is used in a supervised setting, and the low dimensional representation is learned based only on the small training set. The baseline GPCs in the original space and in the low dimensional representations are also trained using only the labeled examples.

In the first experiment we compare the performance of our algorithm to GPC[5] in the original space, using both the Laplace and EP methods. Results on the oil database[6] are shown in Fig. 2. This database has three classes, input dimensionality 12 and 1000 examples; results from a two dimensional D-GPLVM are also shown in the figure, where 10 to 100 examples were used for training. For small amounts of training data, the performance of our method is much higher than GPC in the original space.

In the second experiment we consider the case where the dimensionality of the input is higher than the number of examples, and evaluate on the task of discriminating between 3's and 5's in the USPS database. The dimensionality of the input space is 256, and 1540 examples are available. In our experiments between 10 to 100 examples were used for training. As shown in Fig. 3, GPC in the original space performs at chance levels as it can not learn from so few examples. In contrast the D-GPLVM takes advantage of the fact that the data lies on a low dimensional manifold and produces low classification error rates; with 100 examples the error rates are less than 5%. The dimensionality of the latent space is 1. In Table 1, we compare our algorithm to GPC and SVMs both in the input space and in the spaces learned by LDA and GDA. As expected, GDA overfits and performs at chance levels. LDA discovers some of the structure of the manifold, but has more classification errors than D-GPLVM. Note that all classifiers in the input space perform no better than chance, and GPC performs approximately as well as SVMs.

The last example (Fig. 4) shows mean error rates for the UCI wine database. The dimensionality of the input space is 13, and 178 examples of three different classes are available. A D-GPLVM of dimension 2 outperforms GPC in the original space by 2 or 3%, slightly increasing with the number of training points. This is not an expected behavior, as we expect that the difference in performance will decrease with the number of training points.

7. Future Work and Extensions

In this section we discuss possible extensions of the D-GPLVM.

Semi-supervised D-GPLVM The D-GPLVM can be extended to the semi-supervised case by simply applying the prior only over the labeled training data. In this case, the use of the backconstraints is critical, since they will place each unlabeled data in latent space close to the labeled points that are similar in input space. The inverse mapping will be learned using all the labeled and unlabeled data, resulting in a better approximation for the fast inference. If no backconstraints are used, then we need to use a prior over the unlabeled data to preserve at least their local topological structure. For example one can use for the unlabeled data a prior based on LLE technique (Roweis & Saul, 2000).

Using other discriminative criteria In this paper we have presented a general framework to learn the covariance matrix for GPC in a discriminative way, by means of augmenting the GPLVM with an informative prior distribution over the latent space. In particular, we have used a prior that is based on GDA. The proposed framework can be used with other discriminative criterions such as Local Fisher discriminant analysis (Sugiyama, 2006), or the the discriminative distance functions of (Globerson & Roweis, 2005; Xing et al., 2003).

Classifying dynamical sequences When dealing with dynamical sequences, the difference between classes might not be in terms of distances in the input space but in their dynamics. It is well known, for example in gait analysis, that it is easier to discriminate between sets of poses than individual poses (Urtasun et al., 2006). The D-GPLVM can be easily extended to take into account dynamics, by learning a GPDM (Wang et al., 2005) instead of a GPLVM. The GPDM is a latent variable model with a nonlinear probabilistic mapping from latent positions to input space, and a nonlinear dynamical mapping on the latent space. Learning the D-GPDM is equivalent to minimize

$$\mathcal{L}_{\mathcal{S}} \;=\; \mathcal{L} + \frac{1}{\sigma_d^2}\mathrm{tr}\left(\mathbf{S}_b^{-1}\mathbf{S}_w\right) + \mathcal{L}_d\,, \tag{19}$$

[5] We use the GPC implementation available on the GPML website http://www.GaussianProcess.org/gpml/code

[6] http://www.ncrg.aston.ac.uk/GTM/3PhaseData.html

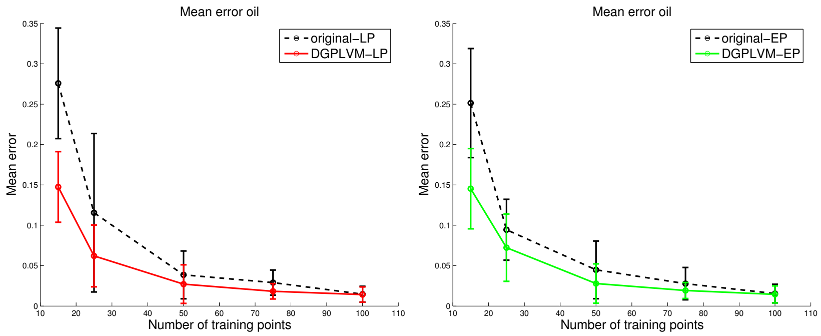


Figure 2. Mean error rate as a function of the number of training points for the "oil" database, using GPC in the original input-space and with a D-GPLVM. The Laplace (left) and EP (right) results are depicted. Our method outperforms GPC, specially when the amount of training data is small.

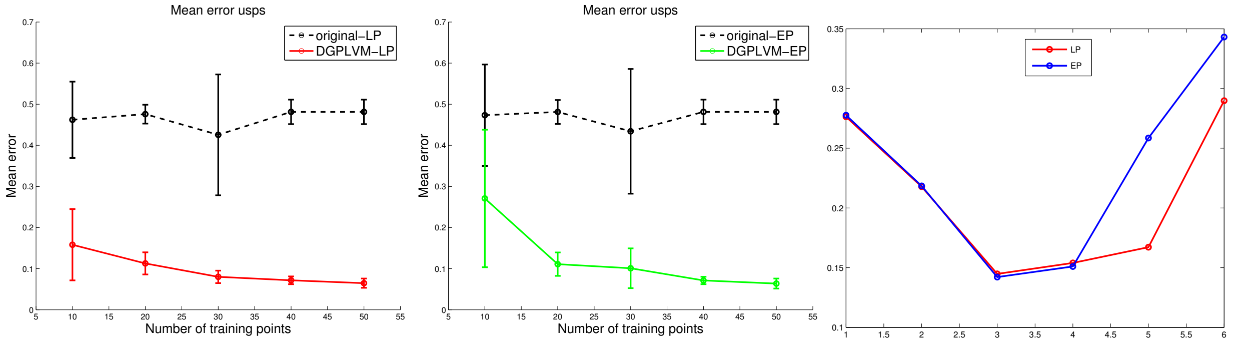


Figure 3. Mean error rate as a function of the number of training points for the "usps" database, where we are trying to discriminate between 3's and 5's. Results using GPC with Laplace (left) and EP (center) in the original input-space and with a D-GPLVM of dimension 2 are shown. (right) Classification error as a function of the number of $1/\sigma_d^2$. The dimensionality of the D-GPLVM latent space is 1.

with

$$\mathcal{L}_d = \frac{d}{2}\ln|\mathbf{K}_X| + \frac{1}{2}\mathrm{tr}\left(\mathbf{K}_X^{-1}\mathbf{X}_{out}\mathbf{X}_{out}^T\right) \qquad (20)$$

where $\mathbf{X}_{out} = [\mathbf{x}_2, ..., \mathbf{x}_N]^T$, $\mathbf{K}_X$ is the $(N-1) \times (N-1)$ kernel matrix constructed from $\mathbf{X}_{in} = [\mathbf{x}_1, ..., \mathbf{x}_{N-1}]$ to model the dynamics. The inference in (14) might also be modified to include the dynamics term.

Optimizing parameters As described above, the hyperparameters of the covariance matrix learned using a D-GPLVM are optimized during GPC. The latent coordinates can also be estimated during this process, resulting in latent positions which optimize the GPC criterion. This might increase the classification performance.

8. Conclusion

In this paper, we have developed a discriminative GPLVM by employing a prior distribution over the latent space derived from a discriminative criterion. This has the desirable generalization properties of generative models, while being able to better discriminate between classes. In contrast to previous Gaussian Process Classification techniques, our method provides a richer parameterization of the covariance function based on the low dimensional structure of the data. Our method empirically outperforms other classification techniques, especially in cases when the dimensionality of the input space is high and the training set is small.

References

Globerson, A., & Roweis, S. (2005). Metric learning by collapsing classes. *Advances in Neural Information Processing Systems (NIPS)*. Cambridge, MA: MIT Press.

Hinton, G., & Roweis, S. (2002). Stochastic neighbor embedding. *Advances in Neural Information Processing Systems (NIPS)*. Cambridge, MA: MIT Press.

Iwata, T., Saito, K., Ueda, N., Stromsten, S., Griffiths, T. L., & Tenenbaum, J. B. (2005). Parametric embedding for class visualization. *Advances in Neural Infor-*

Table 1. Classification accuracies on the USPS database with different subsets, using our algorithm, GPC and SVMs both in the input space and in the spaces learned by LDA and GDA. GDA overfits and performs at chance levels. LDA discovers some of the structure of the manifold, but has more classification errors than D-GPLVM. All classifiers in the input space perform no better than chance.

	10	20	30	40	50	100
DGPLVM LP	**0.16 ± 0.087**	**0.11 ± 0.027**	**0.08 ± 0.015**	0.072 ± 0.0095	0.065 ± 0.011	**0.045 ± 0.0069**
DGPLVM EP	0.27 ± 0.17	**0.11 ± 0.028**	0.1 ± 0.048	**0.071 ± 0.009**	**0.064 ± 0.012**	**0.045 ± 0.0068**
LP	0.46 ± 0.093	0.48 ± 0.023	0.43 ± 0.15	0.48 ± 0.03	0.48 ± 0.03	0.49 ± 0.035
EP	0.47 ± 0.12	0.48 ± 0.029	0.43 ± 0.15	0.48 ± 0.03	0.48 ± 0.03	0.49 ± 0.035
SVM	0.49 ± 0.036	0.47 ± 0.001	0.48 ± 0.033	0.48 ± 0.03	0.48 ± 0.03	0.49 ± 0.035
LDA SVM	0.39 ± 0.045	0.31 ± 0.069	0.27 ± 0.059	0.22 ± 0.061	0.16 ± 0.053	0.099 ± 0.026
GDA SVM	0.49 ± 0.036	0.47 ± 0.001	0.48 ± 0.033	0.48 ± 0.03	0.48 ± 0.03	0.48 ± 0.032
LDA LP	**0.16 ± 0.057**	0.13 ± 0.037	0.13 ± 0.033	0.1 ± 0.02	0.084 ± 0.021	0.071 ± 0.012
LDA EP	**0.16 ± 0.056**	0.13 ± 0.038	0.13 ± 0.033	0.1 ± 0.02	0.085 ± 0.021	0.072 ± 0.012
GDA LP	0.51 ± 0.036	0.48 ± 0.03	0.48 ± 0.033	0.48 ± 0.03	0.48 ± 0.03	0.49 ± 0.035
GDA EP	0.51 ± 0.036	0.47 ± 0.001	0.48 ± 0.033	0.48 ± 0.03	0.48 ± 0.03	0.49 ± 0.035

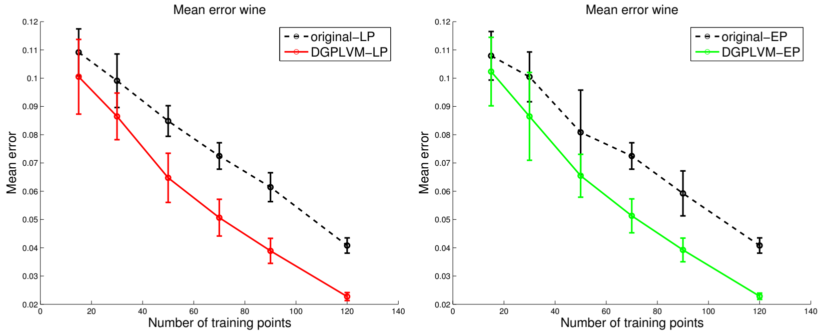


Figure 4. Mean error rate as a function of the number of training points for the UCI wine database. A D-GPLVM of dimension 2 outperforms GPC in the original space by 2 or 3%.

mation Processing Systems (NIPS). Cambridge, MA: MIT Press.

Kuss, M., & Rasmussen, C. E. (2006). Assessing approximations for gaussian process classification. *Advances in Neural Information Processing Systems (NIPS)* (pp. 699 – 706). MIT Press.

Lawrence, N. D. (2004). Gaussian process models for visualisation of high dimensional data. *Advances in Neural Information Processing Systems (NIPS)*. Cambridge, MA: MIT Press.

Lawrence, N. D., & Quiñonero-Candela, J. (2006). Local distance preservation in the gp-lvm through back constraints. *International Conference in Machine Learning* (pp. 96–103).

Rasmussen, C. E., & Williams, C. K. (2006). *Gaussian process for machine learning*. MIT Press.

Roweis, S., & Saul, L. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science*, *290*, 2323–2326.

Seeger, M. (2001). Covariance kernels from bayesian generative models. *Advances in Neural Information Processing Systems (NIPS)*. Cambridge, MA: MIT Press.

Sugiyama, M. (2006). Local fisher discriminant analysis for supervised dimensionality reduction. *International Conference in Machine Learning*.

Urtasun, R., Fleet, D. J., & Fua, P. (2006). Temporal motion models for monocular and multiview 3d human body tracking. *Computer Vision and Image Understanding*.

Wang, J., Fleet, D. J., & Hertzman, A. (2005). Gaussian process dynamical models. *Advances in Neural Information Processing Systems (NIPS)*. Cambridge, MA: MIT Press.

Xing, E., Ng, A., Jordan, M., & Russell, S. (2003). Distance metric learning, with application to clustering with side-information. *Advances in Neural Information Processing Systems (NIPS)*. Cambridge, MA: MIT Press.

Zien, A., & Quiñonero-Candela, J. (2005). Large margin non-linear embedding. *International Conference in Machine Learning*.