Deep Gaussian Processes

Andreas C. Damianou **Neil D. Lawrence**
Dept. of Computer Science & Sheffield Institute for Translational Neuroscience,
University of Sheffield, UK

Abstract

In this paper we introduce deep Gaussian process (GP) models. Deep GPs are a deep belief network based on Gaussian process mappings. The data is modeled as the output of a multivariate GP. The inputs to that Gaussian process are then governed by another GP. A single layer model is equivalent to a standard GP or the GP latent variable model (GP-LVM). We perform inference in the model by approximate variational marginalization. This results in a strict lower bound on the marginal likelihood of the model which we use for model selection (number of layers and nodes per layer). Deep belief networks are typically applied to relatively large data sets using stochastic gradient descent for optimization. Our fully Bayesian treatment allows for the application of deep models even when data is scarce. Model selection by our variational bound shows that a five layer hierarchy is justified even when modelling a digit data set containing only 150 examples.

1 Introduction

Probabilistic modelling with neural network architectures constitute a well studied area of machine learning. The recent advances in the domain of deep learning [Hinton and Osindero, 2006, Bengio et al., 2012] have brought this kind of models again in popularity. Empirically, deep models seem to have structural advantages that can improve the quality of learning in complicated data sets associated with abstract information [Bengio, 2009]. Most deep algorithms require a large amount of data to perform learning, however, we know that humans are able to perform inductive reasoning (equivalent to concept generalization) with only a few examples [Tenenbaum et al., 2006]. This provokes the question as to whether deep structures and the learning of abstract structure can be undertaken in *smaller* data sets. For smaller data sets, questions of generalization arise: to demonstrate such structures are justified it is useful to have an objective measure of the model's applicability.

The traditional approach to deep learning is based around binary latent variables and the restricted Boltzmann machine (RBM) [Hinton, 2010]. Deep hierarchies are constructed by stacking these models and various approximate inference techniques (such as contrastive divergence) are used for estimating model parameters. A significant amount of work has then to be done with annealed importance sampling if even the *likelihood*[1] of a data set under the RBM model is to be estimated [Salakhutdinov and Murray, 2008]. When deeper hierarchies are considered, the estimate is only of a lower bound on the data likelihood. Fitting such models to smaller data sets and using Bayesian approaches to deal with the complexity seems completely futile when faced with these intractabilities.

The emergence of the Boltzmann machine (BM) at the core of one of the most interesting approaches to modern machine learning is very much a case of a the field going back to the future: BMs rose to prominence in the early 1980s, but the practical implications associated with their training led to their neglect until families of algorithms were developed for the RBM model with its reintroduction as a product of experts in the late nineties [Hinton, 1999].

The computational intractabilities of Boltzmann machines led to other families of methods, in particular kernel methods such as the support vector machine (SVM), to be considered for the domain of data classification. Almost contemporaneously to the SVM, Gaussian process (GP) models [Rasmussen and Williams, 2006] were introduced as a fully probabilistic substitute for the multilayer perceptron (MLP), inspired by the observation [Neal, 1996] that, under certain conditions, a GP *is* an MLP with *infinite* units in the hidden layer. MLPs also relate to deep learning models: deep learning algorithms have been used to pretrain autoencoders for dimensionality reduction [Hinton and Salakhut-

Appearing in Proceedings of the 16^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2013, Scottsdale, AZ, USA. Volume 31 of JMLR: W&CP 31.

[1] We use emphasis to clarify we are referring to the model likelihood, not the marginal likelihood required in Bayesian model selection.

dinov, 2006]. Traditional GP models have been extended to more expressive variants, for example by considering sophisticated covariance functions [Durrande et al., 2011, Gönen and Alpaydin, 2011] or by embedding GPs in more complex probabilistic structures [Snelson et al., 2004, Wilson et al., 2012] able to learn more powerful representations of the data. However, all GP-based approaches considered so far do not lead to a principled way of obtaining truly deep architectures and, to date, the field of deep learning remains mainly associated with RBM-based models.

The conditional probability of a single hidden unit in an RBM model, given its parents, is written as

$$p(y|\mathbf{x}) = \sigma(\mathbf{w}^\top \mathbf{x})^y (1 - \sigma(\mathbf{w}^\top \mathbf{x}))^{(1-y)},$$

where here y is the output variable of the RBM, $\mathbf{x}$ is the set of inputs being conditioned on and $\sigma(z) = (1 + \exp(-z))^{-1}$. The conditional density of the output depends only on a linear weighted sum of the inputs. The representational power of a Gaussian process in the same role is significantly greater than that of an RBM. For the GP the corresponding likelihood is over a continuous variable, but it is a nonlinear function of the inputs,

$$p(y|\mathbf{x}) = \mathcal{N}\left(y|f(\mathbf{x}), \sigma^2\right),$$

where $\mathcal{N}\left(\cdot|\mu, \sigma^2\right)$ is a Gaussian density with mean μ and variance σ^2. In this case the likelihood is dependent on a mapping function, $f(\cdot)$, rather than a set of intermediate parameters, $\mathbf{w}$. The approach in Gaussian process modelling is to place a prior directly over the classes of functions (which often specifies smooth, stationary nonlinear functions) and integrate them out. This can be done analytically. In the RBM the model likelihood is estimated and maximized with respect to the parameters, $\mathbf{w}$. For the RBM marginalizing $\mathbf{w}$ is not analytically tractable. We note in passing that the two approaches can be mixed if $p(y|\mathbf{x}) = \sigma(f(\mathbf{x}))^y (1 - \sigma(f(\mathbf{x}))^{(1-y)}$, which recovers a GP classification model. Analytic integration is no longer possible though, and a common approach to approximate inference is the expectation propagation algorithm [see e.g. Rasmussen and Williams, 2006]. However, we don't consider this idea further in this paper.

Inference in deep models requires marginalization of $\mathbf{x}$ as they are typically treated as *latent* variables[2], which in the case of the RBM are binary variables. The number of the terms in the sum scales exponentially with the input dimension rendering it intractable for anything but the smallest models. In practice, sampling and, in particular, the contrastive divergence algorithm, are used for training. Similarly, marginalizing $\mathbf{x}$ in the GP is analytically intractable, even for simple prior densities like the Gaussian. In the GP-LVM [Lawrence, 2005] this problem is solved through maximizing with respect to the variables (instead of the parameters, which are marginalized) and these models have been combined in stacks to form the hierarchical GP-LVM [Lawrence and Moore, 2007] which is a maximum a posteriori (MAP) approach for learning deep GP models. For this MAP approach to work, however, a strong prior is required on the top level of the hierarchy to ensure the algorithm works and MAP learning prohibits model selection because no estimate of the marginal likelihood is available.

There are two main contributions in this paper. Firstly, we exploit recent advances in variational inference [Titsias and Lawrence, 2010] to marginalize the latent variables in the hierarchy variationally. Damianou et al. [2011] has already shown how using these approaches two Gaussian process models can be stacked. This paper goes further to show that through variational approximations any number of GP models can be stacked to give truly deep hierarchies. The variational approach gives us a rigorous lower bound on the *marginal* likelihood of the model, allowing it to be used for model selection. Our second contribution is to use this lower bound to demonstrate the applicability of deep models even when data is scarce. The variational lower bound gives us an objective measure from which we can select different structures for our deep hierarchy (number of layers, number of nodes per layer). In a simple digits example we find that the best lower bound is given by the model with the deepest hierarchy we applied (5 layers).

The deep GP consists of a cascade of hidden layers of latent variables where each node acts as output for the layer above and as input for the layer below—with the observed outputs being placed in the leaves of the hierarchy. Gaussian processes govern the mappings between the layers.

A single layer of the deep GP is effectively a Gaussian process latent variable model (GP-LVM), just as a single layer of a regular deep model is typically an RBM. [Titsias and Lawrence, 2010] have shown that latent variables can be approximately marginalized in the GP-LVM allowing a variational lower bound on the likelihood to be computed. The appropriate size of the latent space can be computed using automatic relevance determination (ARD) priors [Neal, 1996]. Damianou et al. [2011] extended this approach by placing a GP prior over the latent space, resulting in a Bayesian dynamical GP-LVM. Here we extend that approach to allow us to approximately marginalize any number of hidden layers. We demonstrate how a deep hierarchy of Gaussian processes can be obtained by marginalising out the latent variables in the structure, obtaining an approximation to the fully Bayesian training procedure and a variational approximation to the true posterior of the latent variables given the outputs. The resulting model is very flexible and should open up a range of applications for deep structures [3].

[2] They can also be treated as observed, e.g. in the upper most layer of the hierarchy where we might include the data label.

[3] A preliminary version of this paper has been presented in [Damianou and Lawrence, 2012].

2 The Model

We first consider standard approaches to modeling with GPs. We then extend these ideas to deep GPs by considering Gaussian process priors over the inputs to the GP model. We can apply this idea recursively to obtain a deep GP model.

2.1 Standard GP Modelling

In the traditional probabilistic inference framework, we are given a set of training input-output pairs, stored in matrices $\mathbf{X} \in \mathcal{R}^{N\times Q}$ and $\mathbf{Y} \in \mathcal{R}^{N\times D}$ respectively, and seek to estimate the unobserved, *latent* function $f = f(\mathbf{x})$, responsible for generating $\mathbf{Y}$ given $\mathbf{X}$. In this setting, Gaussian processes (GPs) [Rasmussen and Williams, 2006] can be employed as nonparametric prior distributions over the latent function f. More formally, we assume that each datapoint $\mathbf{y}_n$ is generated from the corresponding $f(\mathbf{x}_n)$ by adding independent Gaussian noise, i.e.

$$\mathbf{y}_n = f(\mathbf{x}_n) + \epsilon_n, \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \sigma_\epsilon^2 \mathbf{I}), \tag{1}$$

and f is drawn from a Gaussian process, i.e. $f(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, k(x, x'))$. This (zero-mean) Gaussian process prior only depends on the covariance function k operating on the inputs $\mathbf{X}$. As we wish to obtain a flexible model, we only make very general assumptions about the form of the generative mapping f and this is reflected in the choice of the covariance function which defines the properties of this mapping. For example, an exponentiated quadratic covariance function, $k(\mathbf{x}_i, \mathbf{x}_j) = (\sigma_{se})^2 \exp\left(-\frac{(\mathbf{x}_i - \mathbf{x}_j)^2}{2l^2}\right)$, forces the latent functions to be infinitely smooth. We denote any covariance function hyperparameters (such as (σ_{se}, l) of the aforementioned covariance function) by $\boldsymbol{\theta}$. The collection of latent function instantiations, denoted by $\mathbf{F} = \{\mathbf{f}_n\}_n^N$, is normally distributed, allowing us to compute analytically the marginal likelihood [4]

$$\begin{aligned} p(\mathbf{Y}|\mathbf{X}) &= \int \prod_{n=1}^{N} p(\mathbf{y}_n|\mathbf{f}_n) p(\mathbf{f}_n|\mathbf{x}_n) \mathrm{d}\mathbf{F} \\ &= \mathcal{N}(\mathbf{Y}|\mathbf{0}, \mathbf{K}_{NN} + \sigma_\epsilon^2 \mathbf{I}), \mathbf{K}_{NN} = k(\mathbf{X}, \mathbf{X}). \end{aligned} \tag{2}$$

Gaussian processes have also been used with success in unsupervised learning scenarios, where the input data $\mathbf{X}$ are not directly observed. The Gaussian process latent variable model (GP-LVM) [Lawrence, 2005, 2004] provides an elegant solution to this problem by treating the unobserved inputs $\mathbf{X}$ as latent variables, while employing a product of D independent GPs as prior for the latent mapping. The assumed generative procedure takes the form: $y_{nd} = f_d(\mathbf{x}_n) + \epsilon_{nd}$, where $\boldsymbol{\epsilon}$ is again Gaussian with variance σ_ϵ^2 and $\mathbf{F} = \{\mathbf{f}_d\}_{d=1}^D$ with $f_{nd} = f_d(\mathbf{x}_n)$. Given a finite data set, the Gaussian process priors take the form

$$p(\mathbf{F}|\mathbf{X}) = \prod_{d=1}^{D} \mathcal{N}(\mathbf{f}_d|\mathbf{0}, \mathbf{K}_{NN}) \tag{3}$$

which is a Gaussian and, thus, allows for general non-linear mappings to be marginalised out analytically to obtain the likelihood $p(\mathbf{Y}|\mathbf{X}) = \prod_{d=1}^{D} \mathcal{N}(\mathbf{y}_d|\mathbf{0}, \mathbf{K}_{NN} + \sigma_\epsilon^2 \mathbf{I})$, analogously to equation (2).

2.2 Deep Gaussian Processes

Our deep Gaussian process architecture corresponds to a graphical model with three kinds of nodes, illustrated in figure 1(a): the leaf nodes $\mathbf{Y} \in \mathcal{R}^{N\times D}$ which are observed, the intermediate latent spaces $\mathbf{X}_h \in \mathcal{R}^{N\times Q_h}, h = 1, ..., H-1$, where H is the number of hidden layers, and the parent latent node $\mathbf{Z} = \mathbf{X}_H \in \mathcal{R}^{N\times Q_Z}$. The parent node can be unobserved and potentially constrained with a prior of our choice (e.g. a dynamical prior), or could constitute the given inputs for a supervised learning task. For simplicity, here we focus on the unsupervised learning scenario. In this deep architecture, all intermediate nodes $\mathbf{X}_h$ act as inputs for the layer below (including the leaves) and as outputs for the layer above. For simplicity, consider a structure with only two hidden units, as the one depicted in figure 1(b). The generative process takes the form:

$$\begin{aligned} y_{nd} &= f_d^Y(\mathbf{x}_n) + \epsilon_{nd}^Y, \quad d = 1, ..., D, \quad \mathbf{x}_n \in \mathcal{R}^Q \\ x_{nq} &= f_q^X(\mathbf{z}_n) + \epsilon_{nq}^X, \quad q = 1, ..., Q, \quad \mathbf{z}_n \in \mathcal{R}^{Q_Z} \end{aligned} \tag{4}$$

and the intermediate node is involved in two Gaussian processes, f^Y and f^X, playing the role of an input and an output respectively: $f^Y \sim \mathcal{GP}(\mathbf{0}, k^Y(\mathbf{X}, \mathbf{X}))$ and $f^X \sim \mathcal{GP}(\mathbf{0}, k^X(\mathbf{Z}, \mathbf{Z}))$. This structure can be naturally extended vertically (i.e. deeper hierarchies) or horizontally (i.e. segmentation of each layer into different partitions of the output space), as we will see later in the paper. However, it is already obvious how each layer adds a significant number of model parameters ($\mathbf{X}_h$) as well as a regularization challenge, since the size of each latent layer is crucial but has to be a priori defined. For this reason, unlike Lawrence and Moore [2007], we seek to variationally marginalise out the whole latent space. Not only this will allow us to obtain an automatic Occam's razor due to the Bayesian training, but also we will end up with a significantly lower number of model parameters, since the variational procedure only adds variational parameters. The first step to this approach is to define automatic relevance determination (ARD) covariance functions for the GPs:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \sigma_{ard}^2 e^{-\frac{1}{2}\sum_{q=1}^{Q} w_q (x_{i,q} - x_{j,q})^2}. \tag{5}$$

This covariance function assumes a different weight w_q for each latent dimension and this can be exploited in a Bayesian training framework in order to "switch off" irrelevant dimensions by driving their corresponding weight to

[4] All probabilities involving f should also have $\boldsymbol{\theta}$ in the conditioning set, but here we omit it for clarity.

zero, thus helping towards automatically finding the structure of complex models. However, the nonlinearities introduced by this covariance function make the Bayesian treatment of this model challenging. Nevertheless, following recent non-standard variational inference methods we can define analytically an approximate Bayesian training procedure, as will be explained in the next section.

2.3 Bayesian Training

A Bayesian training procedure requires optimisation of the model evidence:

$$\log p(\mathbf{Y}) = \log \int_{\mathbf{X},\mathbf{Z}} p(\mathbf{Y}|\mathbf{X})p(\mathbf{X}|\mathbf{Z})p(\mathbf{Z}). \quad (6)$$

When prior information is available regarding the observed data (e.g. their dynamical nature is known a priori), the prior distribution on the parent latent node can be selected so as to constrain the whole latent space through propagation of the prior density through the cascade. Here we take the general case where $p(\mathbf{Z}) = \mathcal{N}(\mathbf{Z}|\mathbf{0}, \mathbf{I})$. However, the integral of equation (6) is intractable due to the nonlinear way in which $\mathbf{X}$ and $\mathbf{Z}$ are treated through the GP priors f^Y and f^X. As a first step, we apply Jensen's inequality to find a variational lower bound $\mathcal{F}_v \leq \log p(\mathbf{Y})$, with

$$\mathcal{F}_v = \int_{\mathbf{X},\mathbf{Z},\mathbf{F}^Y,\mathbf{F}^X} \mathcal{Q} \log \frac{p(\mathbf{Y}, \mathbf{F}^Y, \mathbf{F}^X, \mathbf{X}, \mathbf{Z})}{\mathcal{Q}}, \quad (7)$$

where we introduced a variational distribution $\mathcal{Q}$, the form of which will be defined later on. By noticing that the joint distribution appearing above can be expanded in the form

$$p(\mathbf{Y},\mathbf{F}^Y, \mathbf{F}^X, \mathbf{X}, \mathbf{Z}) = p(\mathbf{Y}|\mathbf{F}^Y)p(\mathbf{F}^Y|\mathbf{X})p(\mathbf{X}|\mathbf{F}^X)p(\mathbf{F}^X|\mathbf{Z})p(\mathbf{Z}), \quad (8)$$

we see that the integral of equation (7) is still intractable because $\mathbf{X}$ and $\mathbf{Z}$ still appear nonlinearly in the $p(\mathbf{F}^Y|\mathbf{X})$ and $p(\mathbf{F}^X|\mathbf{Z})$ terms respectively. A key result of [Titsias and Lawrence, 2010] is that expanding the probability space of the GP prior $p(\mathbf{F}|\mathbf{X})$ with extra variables allows for priors on the latent space to be propagated through the nonlinear mapping f. More precisely, we augment the probability space of equation (3) with K auxiliary pseudo-inputs $\tilde{\mathbf{X}} \in \mathcal{R}^{K\times Q}$ and $\tilde{\mathbf{Z}} \in \mathcal{R}^{K\times Q_Z}$ that correspond to a collection of function values $\mathbf{U}^Y \in \mathcal{R}^{K\times D}$ and $\mathbf{U}^X \in \mathcal{R}^{K\times Q}$ respectively [5]. Following this approach, we obtain the augmented probability space: $p(\mathbf{Y}, \mathbf{F}^Y, \mathbf{F}^X, \mathbf{X}, \mathbf{Z}, \mathbf{U}^Y, \mathbf{U}^X, \tilde{\mathbf{X}}, \tilde{\mathbf{Z}}) =$

$$p(\mathbf{Y}|\mathbf{F}^Y)p(\mathbf{F}^Y|\mathbf{U}^Y, \mathbf{X})p(\mathbf{U}^Y|\tilde{\mathbf{X}}) \cdot p(\mathbf{X}|\mathbf{F}^X)p(\mathbf{F}^X|\mathbf{U}^X, \mathbf{Z})p(\mathbf{U}^X|\tilde{\mathbf{X}})p(\mathbf{Z}) \quad (9)$$

The pseudo-inputs $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{Z}}$ are known as *inducing points*, and will be dropped from our expressions from now on, for clarity. Note that $\mathbf{F}^Y$ and $\mathbf{U}^Y$ are draws from the same GP so that $p(\mathbf{U}^Y)$ and $p(\mathbf{F}^Y|\mathbf{U}^Y, \mathbf{X})$ are also Gaussian distributions (and similarly for $p(\mathbf{U}^X), p(\mathbf{F}^X|\mathbf{U}^X, \mathbf{Z})$).

We are now able to define a variational distribution $\mathcal{Q}$ which, when combined with the new expressions for the augmented GP priors, results in a tractable variational bound. Specifically, we have:

$$\mathcal{Q} = p(\mathbf{F}^Y|\mathbf{U}^Y, \mathbf{X})q(\mathbf{U}^Y)q(\mathbf{X}) \cdot p(\mathbf{F}^X|\mathbf{U}^X, \mathbf{Z})q(\mathbf{U}^X)q(\mathbf{Z}). \quad (10)$$

We select $q(\mathbf{U}^Y)$ and $q(\mathbf{U}^X)$ to be free-form variational distributions, while $q(\mathbf{X})$ and $q(\mathbf{Z})$ are chosen to be Gaussian, factorised with respect to dimensions:

$$q(\mathbf{X}) = \prod_{q=1}^{Q} \mathcal{N}(\boldsymbol{\mu}_q^X, \mathbf{S}_q^X),\ q(\mathbf{Z}) = \prod_{q=1}^{Q_Z} \mathcal{N}(\boldsymbol{\mu}_q^Z, \mathbf{S}_q^Z). \quad (11)$$

By substituting equation (10) back to (7) while also replacing the original joint distribution with its augmented version in equation (9), we see that the "difficult" terms $p(\mathbf{F}^Y|\mathbf{U}^Y, \mathbf{X})$ and $p(\mathbf{F}^X|\mathbf{U}^X, \mathbf{Z})$ cancel out in the fraction, leaving a quantity that can be computed analytically:

$$\mathcal{F}_v = \int \mathcal{Q} \log \frac{p(\mathbf{Y}|\mathbf{F}^Y)p(\mathbf{U}^Y)p(\mathbf{X}|\mathbf{F}^X)p(\mathbf{U}^X)p(\mathbf{Z})}{\mathcal{Q}'}, \quad (12)$$

where $\mathcal{Q}' = q(\mathbf{U}^Y)q(\mathbf{X})q(\mathbf{U}^X)q(\mathbf{Z})$ and the above integration is with respect to $\{\mathbf{X}, \mathbf{Z}, \mathbf{F}^Y, \mathbf{F}^X, \mathbf{U}^Y, \mathbf{U}^X\}$. More specifically, we can break the logarithm in equation (12) by grouping the variables of the fraction in such a way that the bound can be written as:

$$\mathcal{F}_v = \mathbf{g}_Y + \mathbf{r}_X + \mathcal{H}_{q(\mathbf{X})} - \mathrm{KL}\left(q(\mathbf{Z}) \parallel p(\mathbf{Z})\right) \quad (13)$$

where $\mathcal{H}$ represents the entropy with respect to a distribution, KL denotes the Kullback – Leibler divergence and, using $\langle\cdot\rangle$ to denote expectations,

$$\mathbf{g}_Y = g(\mathbf{Y}, \mathbf{F}^Y, \mathbf{U}^Y, \mathbf{X}) = \left\langle \log p(\mathbf{Y}|\mathbf{F}^Y) + \log \tfrac{p(\mathbf{U}^Y)}{q(\mathbf{U}^Y)} \right\rangle_{p(\mathbf{F}^Y|\mathbf{U}^Y,\mathbf{X})q(\mathbf{U}^Y)q(\mathbf{X})}$$

$$\mathbf{r}_X = r(\mathbf{X}, \mathbf{F}^X, \mathbf{U}^X, \mathbf{Z}) = \left\langle \log p(\mathbf{X}|\mathbf{F}^X) + \log \tfrac{p(\mathbf{U}^X)}{q(\mathbf{U}^X)} \right\rangle_{p(\mathbf{F}^X|\mathbf{U}^X,\mathbf{Z})q(\mathbf{U}^X)q(\mathbf{X})q(\mathbf{Z})} \quad (14)$$

Both terms $\mathbf{g}_Y$ and $\mathbf{r}_X$ involve known Gaussian densities and are, thus, tractable. The $\mathbf{g}_Y$ term is only associated with the leaves and, thus, is the same as the bound found for the Bayesian GP-LVM [Titsias and Lawrence, 2010]. Since it only involves expectations with respect to Gaussian distributions, the GP output variables are only involved in

[5] The number of inducing points, K, does not need to be the same for every GP of the overall deep structure.

a quantity of the form $\mathbf{Y}\mathbf{Y}^{\mathrm{T}}$. Further, as can be seen from the above equations, the function $r(\cdot)$ is similar to $g(\cdot)$ but it requires expectations with respect to densities of all of the variables involved (i.e. with respect to all function inputs). Therefore, $\mathbf{r}_X$ will involve $\mathbf{X}$ (the outputs of the top layer) in a term $\left\langle \mathbf{X}\mathbf{X}^{\mathrm{T}} \right\rangle_{q(\mathbf{X})} = \sum_{q=1}^{Q} \left[\boldsymbol{\mu}_q^X \left(\boldsymbol{\mu}_q^X \right)^{\mathrm{T}} + \mathbf{S}_q^X \right]$.

3 Extending the hierarchy

Although the main calculations were demonstrated in a simple hierarchy, it is easy to extend the model vertically, i.e. by adding more hidden layers, or horizontally, i.e. by considering conditional independencies of the latent variables belonging to the same layer. The first case only requires adding more $\mathbf{r}_X$ functions to the variational bound, i.e. instead of a single $\mathbf{r}_X$ term we will now have the sum: $\sum_{h=1}^{H-1} \mathbf{r}_{X_h}$, where $\mathbf{r}_{X_h} = r(\mathbf{X}_h, \mathbf{F}^{X_h}, \mathbf{U}^{X_h}, \mathbf{X}_{h+1})$, $\mathbf{X}_H = \mathbf{Z}$.

Now consider the horizontal expansion scenario and assume that we wish to break the single latent space $\mathbf{X}_h$, of layer h, to M_h conditionally independent subsets. As long as the variational distribution $q(\mathbf{X}_h)$ of equation (11) is chosen to be factorised in a consistent way, this is feasible by just breaking the original $\mathbf{r}_{X_h}$ term of equation (14) into the sum $\sum_{m=1}^{M_h} \mathbf{r}_{X_h}^{(m)}$. This follows just from the fact that, due to the independence assumption, it holds that $\log p(\mathbf{X}_h|\mathbf{X}_{h+1}) = \sum_{m=1}^{M_h} \log p(\mathbf{X}_h^{(m)}|\mathbf{X}_{h+1})$. Notice that the same principle can also be applied to the leaves by breaking the $\mathbf{g}_Y$ term of the bound. This scenario arises when, for example we are presented with multiple different output spaces which, however, we believe they have some commonality. For example, when the observed data are coming from a video and an audio recording of the same event. Given the above, the variational bound for the most general version of the model takes the form:

$$\mathcal{F}_v = \sum_{m=1}^{M_Y} \mathbf{g}_Y^{(m)} + \sum_{h=1}^{H-1} \sum_{m=1}^{M_h} \mathbf{r}_{X_h}^{(m)} + \sum_{h=1}^{H-1} \mathcal{H}_{q(\mathbf{X}_h)} - \mathrm{KL}\left(q(\mathbf{Z}) \parallel p(\mathbf{Z})\right). \quad (15)$$

Figure 1(c) shows the association of this objective function's terms with each layer of the hierarchy. Recall that each $\mathbf{r}_{X_h}^{(m)}$ and $\mathbf{g}_Y^{(m)}$ term is associated with a different GP and, thus, is coming with its own set of automatic relevance determination (ARD) weights (described in equation (5)).

3.1 Deep multiple-output Gaussian processes

The particular way of extending the hierarchies horizontally, as presented above, can be seen as a means of performing unsupervised multiple-output GP learning. This only requires assigning a different $\mathbf{g}_Y$ term (and, thus, associated ARD weights) to each vector $\mathbf{y}_d$, where d indexes the output dimensions. After training our model, we hope that the columns of $\mathbf{Y}$ that encode similar information will be assigned relevance weight vectors that are also similar. This idea can be extended to all levels of the hierarchy, thus obtaining a fully factorised deep GP model.

This special case of our model makes the connection between our model's structure and neural network architectures more obvious: the ARD parameters play a role similar to the weights of neural networks, while the latent variables play the role of neurons which learn hierarchies of features.

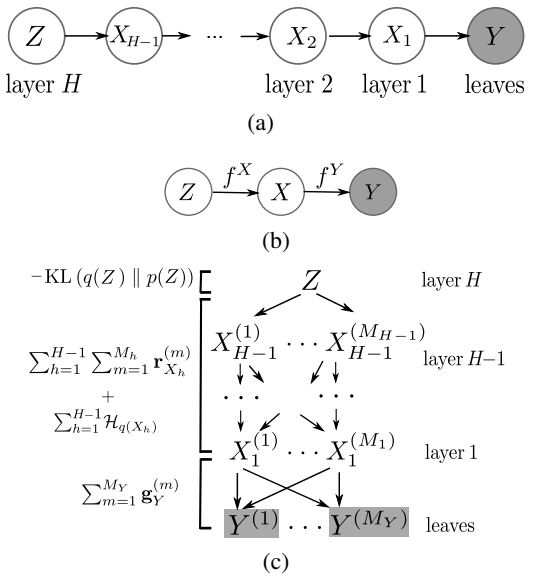


Figure 1: Different representations of the Deep GP model: (a) shows the general architecture with a cascade of H hidden layers, (b) depicts a simplification of a two hidden layer hierarchy also demonstrating the corresponding GP mappings and (c) illustrates the most general case where the leaves and all intermediate nodes are allowed to form conditionally independent groups. The terms of the objective (15) corresponding to each layer are included on the left.

3.2 Parameters and complexity

In all graphical variants shown in figure 1, every arrow represents a generative procedure with a GP prior, corresponding to a set of parameters $\{\tilde{\mathbf{X}}, \boldsymbol{\theta}, \sigma_\epsilon\}$. Each layer of latent variables corresponds to a variational distribution $q(\mathbf{X})$ which is associated with a set of variational means and covariances, as shown in equation (11). The parent node can have the same form as equation (11) or can be constrained with a more informative prior which would couple the points of $q(\mathbf{Z})$. For example, a dynamical prior would introduce $Q \times N^2$ parameters which can, nevertheless, be reparametrized using less variables [Damianou et al., 2011]. However, as is evident from equations (10) and (12), the inducing points and the parameters of $q(\mathbf{X})$ and $q(\mathbf{Z})$ are *variational* rather than model parameters, something which significantly helps in regularizing the problem.

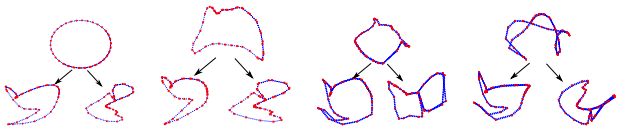
Therefore, adding more layers to the hierarchy does not introduce many more model parameters. Moreover, as in common sparse methods for Gaussian processes [Titsias, 2009], the complexity of each generative GP mapping is reduced from the typical $O(N^3)$ to $O(NM^2)$.

4 Demonstration

In this section we demonstrate the deep GP model in toy and real-world data sets. For all experiments, the model is initialised by performing dimensionality reduction in the observations to obtain the first hidden layer and then repeating this process greedily for the next layers. To obtain the stacked initial spaces we experimented with PCA and the Bayesian GP-LVM, but the end result did not vary significantly. Note that the usual process in deep learning is to seek a dimensional expansion, particularly in the lower layers. In deep GP models, such an expansion *does* occur between the latent layers because there is an infinite basis layer associated with the GP between each latent layer.

4.1 Toy Data

We first test our model on toy data, created by sampling from a three-level stack of GPs. Figure 2 (a) depicts the true hierarchy: from the top latent layer two intermediate latent signals are generated. These, in turn, together generate 10-dimensional observations (not depicted) through sampling of another GP. These observations are then used to train the following models: a deep GP, a simple stacked Isomap [Tenenbaum et al., 2000] and a stacked PCA method, the results of which are shown in figures 2 (b, c, d) respectively. From these models, only the deep GP marginalises the latent spaces and, in contrast to the other two, it is not given any information about the dimensionality of each true signal in the hierarchy; instead, this is learnt automatically through ARD. As can be seen in figure 2, the deep GP finds the correct dimensionality for each hidden layer, but it also discovers latent signals which are closer to the real ones. This result is encouraging, as it indicates that the model can recover the ground truth when samples from it are taken, and gives confidence in the variational learning procedure.



(a) (b) (c) (d)

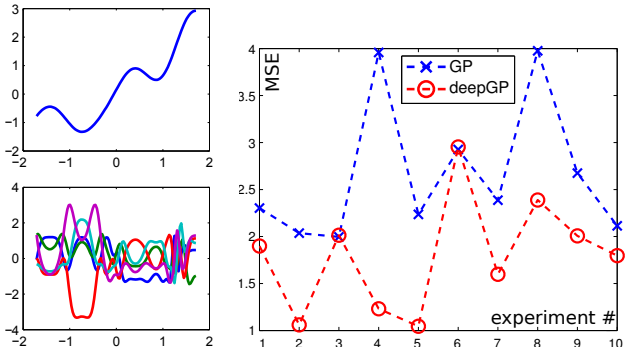
Figure 2: Attempts to reconstruct the real data (fig. (a)) with our model (b), stacked Isomap (c) and stacked PCA (d). Our model can also find the correct dimensionalities automatically.

We next tested our model on a toy regression problem. A deep regression problem is similar to the unsupervised learning problem we have described, but in the uppermost layer we make observations of some set of inputs. For this simple example we created a toy data set by stacking two Gaussian processes as follows: the first Gaussian process employed a covariance function which was the sum of a linear and an quadratic exponential kernel and received as input an equally spaced vector of 120 points. We generated 1-dimensional samples from the first GP and used them as input for the second GP, which employed a quadratic exponential kernel. Finally, we generated 10-dimensional samples with the second GP, thus overall simulating a warped process. The final data set was created by simply ignoring the intermediate layer (the samples from the first GP) and presenting to the tested methods only the continuous equally spaced input given to the first GP and the output of the second GP. To make the data set more challenging, we randomly selected only 25 datapoints for the training set and left the rest for the test set.

Figure 3 nicely illustrates the effects of sampling through two GP models, nonstationarity and long range correlations across the input space become prevalent. A data set of this form would be challenging for traditional approaches because of these long range correlations. Another way of thinking of data like this is as a nonlinear warping of the input space to the GP. Because this type of deep GP only contains one hidden layer, it is identical to the model developed by [Damianou et al., 2011] (where the input given at the top layer of their model was a time vector, but their code is trivially generalized). The additional contribution in this paper will be to provide a more complex deep hierarchy, but still learn the underlying representation correctly. To this end we applied a standard GP (1 layer less than the actual process that generated the data) and a deep GP with two hidden layers (1 layer more than the actual generating process). We repeated our experiment 10 times, each time obtaining different samples from the simulated warped process and different random training splits. Our results show that the deep GP predicted better the unseen data, as can be seen in figure 3(b). The results, therefore, suggest that our deep model can at the same time be flexible enough to model difficult data as well as robust, when modelling data that is less complex than that representable by the hierarchy. We assign these characteristics to the Bayesian learning approach that deals with capacity control automatically.

4.2 Modeling human motion

For our first demonstration on real data we recreate a motion capture data experiment from Lawrence and Moore [2007]. They used data from the CMU MOCAP database representing two subjects walking towards each other and performing a 'high-five'. The data contains 78 frames of motion and each character has 62 dimensions, leading to 124 dimensions in total (i.e. more dimensions than data). To account for the correlated motions of the subjects we

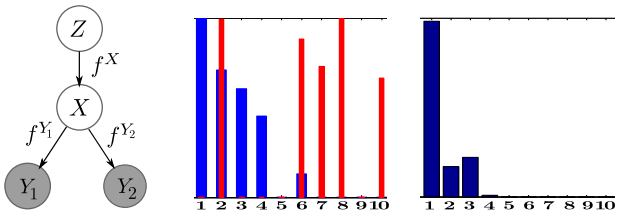


(a) (b)

Figure 3: (a) shows the toy data created for the regression experiment. The top plot shows the (hidden) warping function and bottom plot shows the final (observed) output. (b) shows the results obtained over each experiment repetition.

applied our method with a two-level hierarchy where the two observation sets were taken to be conditionally independent given their parent latent layer. In the layer closest to the data we associated each GP-LVM with a different set of ARD parameters, allowing the layer above to be used in different ways for each character. In this approach we are inspired by the shared GP-LVM structure of Damianou et al. [2012] which is designed to model loosely correlated data sets within the same model. The end result was that we obtained three optimised sets of ARD parameters: one for each modality of the bottom layer (fig. 4(b)), and one for the top node (fig. 4(c)). Our model discovered a common subspace in the intermediate layer, since for dimensions 2 and 6 both ARD sets have a non-zero value. This is expected, as the two subjects perform very similar motions with opposite directions. The ARD weights are also a means of automatically selecting the dimensionality of each layer and subspace. This kind of modelling is impossible for a MAP method like [Lawrence and Moore, 2007] which requires the exact latent structure to be given a priori. The full latent space learned by the aforementioned MAP method is plotted in figure 5 (d,e,f), where fig. (d) corresponds to the top latent space and each of the other two encodes information for each of the two interacting subjects. Our method is not constrained to two dimensional spaces, so for comparison we plot two-dimensional projections of the dominant dimensions of each subspace in figure 5 (a,b,c). The similarity of the latent spaces is obvious. In contrast to Lawrence and Moore [2007], we did not have to constrain the latent space with dynamics in order to obtain results of good quality.

Further, we can sample from these spaces to see what kind of information they encode. Indeed, we observed that the top layer generates outputs which correspond to different variations of the whole sequence, while when sampling from the first layer we obtain outputs which only differ in a small subset of the output dimensions, e.g. those corresponding to the subject's hand.



(a) (b) (c)

Figure 4: Figure (a) shows the deep GP model employed. Figure (b) shows the ARD weights for f^{Y_1} (blue/wider bins) and f^{Y_2} (red/thinner bins) and figure (c) those for f^X.

(a) (b) (c) (d) (e) (f)

Figure 5: Left (a,b,c): projections of the latent spaces discovered by our model, Right (d,e,f): the full latent space learned for the model of Lawrence and Moore [2007].

4.3 Deep learning of digit images

Our final experiment demonstrates the ability of our model to learn latent features of increasing abstraction and we demonstrate the usefulness of an analytic bound on the model evidence as a means of evaluating the quality of the model fit for different choices of the overall depth of the hierarchy. Many deep learning approaches are applied to large digit data sets such as MNIST. Our specific intention is to explore the utility of deep hierarchies when the digit data set is *small*. We subsampled a data set consisting of 50 examples for each of the digits $\{0, 1, 6\}$ taken from the USPS handwritten digit database. Each digit is represented as an image in 16×16 pixels. We experimented with deep GP models of depth ranging from 1 (equivalent to Bayesian GP-LVM) to 5 hidden layers and evaluated each model by measuring the nearest neighbour error in the latent features discovered in each hierarchy. We found that the lower bound on the model evidence increased with the number of layers as did the quality of the model in terms of nearest neighbour errors [6]. Indeed, the single-layer model made 5 mistakes even though it automatically decided to use 10 latent dimensions and the quality of the trained models was increasing with the number of hidden layers. Finally, only one point had a nearest neighbour of a different class in the $4-$dimensional top level's feature space of a model with depth 5. A $2D$ projection of this space is plotted in fig.7. The ARD weights for this model are depicted in fig. 6.

[6] As parameters increase linearly in the deep GP with latent units, we also considered the Bayesian Information Criterion, but we found that it had no effect on the ranking of model quality.

Our final goal is to demonstrate that, as we rise in the hierarchy, features of increasing abstraction are accounted for. To this end, we generated outputs by sampling from each hidden layer. The samples are shown in figure 8. There, it can be seen that the lower levels encode local features whereas the higher ones encode more abstract information.

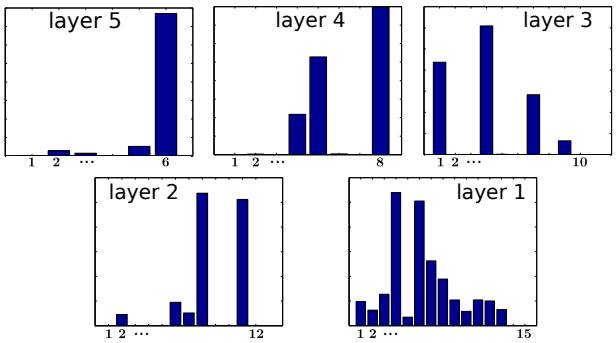


Figure 6: The ARD weights of a deep GP with 5 hidden layers as learned for the digits experiment.

Figure 7: The nearest neighbour class separation test on a deep GP model with depth 5.

5 Discussion and future work

We have introduced a framework for efficient Bayesian training of hierarchical Gaussian process mappings. Our approach approximately marginalises out the latent space, thus allowing for automatic structure discovery in the hierarchy. The method was able to successfully learn a hierarchy of features which describe natural human motion and the pixels of handwritten digits. Our variational lower bound selected a deep hierarchical representation for handwritten digits even though the data in our experiment was relatively scarce (150 data points). We gave persuasive evidence that deep GP models are powerful enough to encode abstract information even for smaller data sets. Further exploration could include testing the model on other inference tasks, such as class conditional density estimation to further validate the ideas. Our method can also be used to improve existing deep algorithms, something which we plan to further investigate by incorporating ideas from past approaches. Indeed, previous efforts to combine GPs with deep structures were successful at unsupervised pre-training [Erhan et al., 2010] or guiding [Snoek et al., 2012] of traditional deep models.

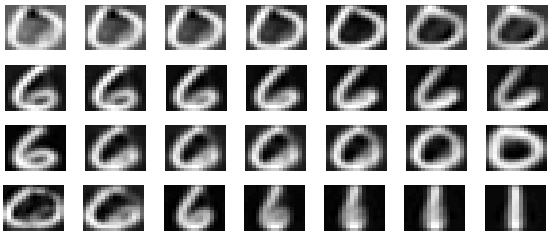


Figure 8: The first two rows (top-down) show outputs obtained when sampling from layers 1 and 2 respectively and encode very local features, e.g. explaining if a "0" has a closed circle or how big the circle of a "6" is. We found many more local features when we sampled from different dimensions. Conversely, when we sampled from the two dominant dimensions of the parent node (two rows in the bottom) we got much more varying outputs, i.e. the higher levels indeed encode much more abstract information.

Although the experiments presented here considered only up to 5 layers in the hierarchy, the methodology is directly applicable to deeper architectures, with which we intend to experiment in the future. The marginalisation of the latent space allows for such an expansion with simultaneous regularisation. The variational lower bound allows us to make a principled choice between models trained using different initializations and with different numbers of layers.

The deep hierarchy we have proposed can also be used with inputs governing the top layer of the hierarchy, leading to a powerful model for regression based on Gaussian processes, but which is not itself a Gaussian process. In the future, we wish to test this model for applications in multi-task learning (where intermediate layers could learn representations shared across the tasks) and in modelling nonstationary data or data involving jumps. These are both areas where a single layer GP struggles.

A remaining challenge is to extend our methodologies to very large data sets. A very promising approach would be to apply *stochastic variational inference* [Hoffman et al., 2012]. In a recent workshop publication Hensman and Lawrence [2012] have shown that the standard variational GP and Bayesian GP-LVM can be made to fit within this formalism. The next step for deep GPs will be to incorporate these large scale variational learning algorithms.

Acknowledgements

Research was supported by the University of Sheffield Moody endowment fund and the Greek State Scholarships Foundation (IKY).

References

Y. Bengio. Learning Deep Architectures for AI. *Found. Trends Mach. Learn.*, 2(1):1–127, Jan. 2009. ISSN 1935-8237. doi: 10.1561/2200000006.

Y. Bengio, A. C. Courville, and P. Vincent. Unsupervised feature learning and deep learning: A review and new perspectives. *CoRR*, abs/1206.5538, 2012.

A. C. Damianou and N. D. Lawrence. Deep Gaussian processes. *NIPS workshop on Deep Learning and Unsupervised Feature Learning (arXiv:1211.0358v1)*, 2012.

A. C. Damianou, M. Titsias, and N. D. Lawrence. Variational Gaussian process dynamical systems. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 2510–2518. 2011.

A. C. Damianou, C. H. Ek, M. K. Titsias, and N. D. Lawrence. Manifold relevance determination. In J. Langford and J. Pineau, editors, *Proceedings of the International Conference in Machine Learning*, volume 29, San Francisco, CA, 2012. Morgan Kauffman.

N. Durrande, D. Ginsbourger, and O. Roustant. Additive kernels for Gaussian process modeling. *ArXiv e-prints 1103.4023*, 2011.

D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio. Why does unsupervised pre-training help deep learning? *J. Mach. Learn. Res.*, 11:625–660, Mar. 2010. ISSN 1532-4435.

M. Gönen and E. Alpaydin. Multiple kernel learning algorithms. *J. Mach. Learn. Res.*, 12:2211–2268, Jul 2011. ISSN 1532-4435.

J. Hensman and N. Lawrence. Gaussian processes for big data through stochastic variational inference. *NIPS workshop on Big Learning*, 2012.

G. Hinton. A Practical Guide to Training Restricted Boltzmann Machines. Technical report, 2010.

G. E. Hinton. Training products of experts by maximizing contrastive likelihood. Technical report, Tech. Rep., Gatsby Computational Neuroscience Unit, 1999.

G. E. Hinton and S. Osindero. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:2006, 2006.

G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 303 (5786):504–507, 2006.

M. Hoffman, D. Blei, C. Wang, and J. Paisley. Stochastic variational inference. *ArXiv e-prints 1206.7051*, 2012.

N. D. Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In *In NIPS*, page 2004, 2004.

N. D. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, 11 2005.

N. D. Lawrence and A. J. Moore. Hierarchical Gaussian process latent variable models. In Z. Ghahramani, editor, *Proceedings of the International Conference in Machine Learning*, volume 24, pages 481–488. Omnipress, 2007. ISBN 1-59593-793-3.

R. M. Neal. *Bayesian Learning for Neural Networks*. Springer, 1996. Lecture Notes in Statistics 118.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. Cambridge, MA, 2006. ISBN 0-262-18253-X.

R. Salakhutdinov and I. Murray. On the quantitative analysis of deep belief networks. In *Proceedings of the International Conference on Machine Learning*, volume 25, 2008.

E. Snelson, C. E. Rasmussen, and Z. Ghahramani. Warped Gaussian processes. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

J. Snoek, R. P. Adams, and H. Larochelle. On nonparametric guidance for learning autoencoder representations. In *Fifteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2012.

J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000. doi: 10.1126/science.290.5500.2319.

J. B. Tenenbaum, C. Kemp, and P. Shafto. Theory-based bayesian models of inductive learning and reasoning. In *Trends in Cognitive Sciences*, pages 309–318, 2006.

M. Titsias. Variational learning of inducing variables in sparse Gaussian processes. *JMLR W&CP*, 5:567–574, 2009.

M. K. Titsias and N. D. Lawrence. Bayesian Gaussian process latent variable model. In Y. W. Teh and D. M. Titterington, editors, *Proceedings of the Thirteenth International Workshop on Artificial Intelligence and Statistics*, volume 9, pages 844–851, Chia Laguna Resort, Sardinia, Italy, 13-16 May 2010. JMLR W&CP 9.

A. G. Wilson, D. A. Knowles, and Z. Ghahramani. Gaussian process regression networks. In J. Langford and J. Pineau, editors, *Proceedings of the 29th International Conference on Machine Learning (ICML)*, Edinburgh, June 2012. Omnipress.