Avoiding Pathologies in Very Deep Networks

Anonymous Author 1
Unknown Institution 1

Anonymous Author 2
Unknown Institution 2

Anonymous Author 3
Unknown Institution 3

Anonymous Author 4
Unknown Institution 4

Abstract

Choosing appropriate architectures and initialization strategies is crucial to good performance of deep networks. To shed light on this problem, we analyze the analogous problem of constructing useful priors on compositions of functions. Specifically, we study deep Gaussian processes, a type of infinitely-wide, deep neural network. We show that for these architectures, the representational capacity of the network tends to capture fewer degrees of freedom as the number of layers increases, retaining only a single degree of freedom in the limit. We propose alternate priors on network architecture which do not suffer from these pathologies. We also derive novel covariance functions obtained by composing infinitely many feature transforms.

1 INTRODUCTION

Much recent work on deep networks has focused on weight initialization (Martens, 2010), regularization (Lee *et al.*, 2007) and network architecture (Poon and Domingos, 2011). The interactions between these different design decisions can be complex and difficult to characterize. We propose to approach the design of deep architectures by examining the problem of creating priors on functions with desirable properties. Although inference in fully Bayesian models is generally difficult, well-defined priors allow us to characterize models in a data-independent way. Once we identify classes of priors with useful properties, these models may suggest regularization, initialization, and architecture choices with similar properties.

Fundamentally, a multilayer neural network implements a composition of vector-valued functions (one per layer). Therefore understanding properties of such function compositions helps us understand these multilayer networks. In this paper, we examine a simple and flexible class of priors on compositions of functions, Deep Gaussian processes. (Damianou and Lawrence, 2013). Deep GPs are simply a prior on compositions of vector-valued functions, where each output of each layer is distributed independently according to a GP prior:

$$f^{1:L}(\mathbf{x}) = f^L(f^{L-1}(\dots f^3(f^2(f^1(\mathbf{x})))\dots)) \quad (1)$$

$$\text{where each } f_d^\ell \overset{\text{ind}}{\sim} \mathcal{GP}\big(0, k_i^\ell(\mathbf{x}, \mathbf{x}')\big) \quad (2)$$

$$\forall \ell \in \{1, \dots, L\} \forall d \in \{1, \dots, D\}$$

Although inference in these models is non-trivial, they can be derived as a special case of multi-layer perceptrons (MLPs), and so are a good candidate for generative models of functions with similar properties to existing neural networks.

By characterizing these models, this paper will show that representations based on repeated composition of independently-initialized functions exhibit a pathology where the representation becomes invariant to all but one direction of variation. However, we will demonstrate that a simple change in architecture, connecting the input to each layer, alleviates this problem.

2 RELATING DEEP NETS AND DEEP GAUSSIAN PROCESSES

Neural Nets with One Hidden Layer In the typical definition of an MLP, the hidden units of the first layer are defined as:

$$\Phi(\mathbf{x}) = h^{(1)}(\mathbf{x}) = \sigma\left(b^{(1)} + W^{(1)}\mathbf{x}\right) \quad (3)$$

where h are the hidden unit activations, b is a bias vector, W is a weight matrix and σ is a sigmoidal function applied element-wise. The output vector $f(\mathbf{x})$ is simply a weighted sum of these hidden unit activations:

$$f(\mathbf{x}) = V^{(1)}\sigma\left(b^{(1)} + W^{(1)}h(\mathbf{x})\right) = V^{(1)}h^{(1)}(\mathbf{x}) \quad (4)$$

where $V^{(1)}$ is another weight matrix.

There exists a correspondence between one-layer MLPs and GPs (Neal, 1995). GP priors can be viewed as a prior

Preliminary work. Under review by AISTATS 2014. Do not distribute.

on neural networks with infinitely many hidden units. More precisely, for any model of the form

$$f(\mathbf{x}) = \frac{1}{K}\alpha^\mathsf{T}\Phi(\mathbf{x}) = \frac{1}{K}\sum_{i=1}^{K}\alpha_i\phi_i(\mathbf{x}), \tag{5}$$

with fixed features $[\phi_1(\mathbf{x}), \ldots, \phi_K(\mathbf{x})]^\mathsf{T} = \Phi(\mathbf{x})$ and i.i.d. α's with zero mean and finite variance σ^2, the central limit theorem implies that as the number of features $K \to \infty$, any two function values $f(\mathbf{x}), f(\mathbf{x}')$ have a joint distribution approaching $\mathcal{N}\left(0, \frac{\sigma^2}{K}\sum_{i=1}^{K}\phi_i(\mathbf{x})\phi_i(\mathbf{x}')\right)$. A joint Gaussian distribution between any two function values is the definition of a Gaussian process.

The result is surprisingly general: It doesn't put any constraints on what the features are (other than having bounded activation), nor does it require that the feature weights α be Gaussian distributed.

We can also work backwards to derive a one-layer MLP from a GP: Mercer's theorem implies that any positive-definite kernel function corresponds to an inner product of features: $k(\mathbf{x}, \mathbf{x}') = \Phi(\mathbf{x})^\mathsf{T}\Phi(\mathbf{x}')$. Thus in the one-hidden-layer case, the correspondence between MLPs and GPs is simple: The features $\phi(\mathbf{x})$ of the GP correspond to the hidden units of the MLP.

2.1 Multiple Hidden Layers

In a neural net with multiple hidden layers, the correspondence is a little more complicated. In an MLP, the n^{th} layer units are given by the recurrence:

$$h^{(n)}(\mathbf{x}) = \sigma\left(b^{(n)} + W^{(n)}h^{(n-1)}(\mathbf{x})\right) \tag{6}$$

This architecture is shown in figure 1b. In this model, each hidden layer's output feeds directly into the next layer's input, weighted by the corresponding element of $W^{(n)}$.

However, in a deep GP, the D outputs $f^{(n)}(\mathbf{x})$ in between each layer are weighted sums of the hidden units of the layer below, and the next layer's hidden units depend only on these D outputs. Thus deep GPs have an extra set of layers that a MLP doesn't have, shown in figure 1c.

There are two ways to directly relate deep GPs to MLPs. First, we can note that, if the hidden units in a deep GP implied by Mercer's theorem $\phi^{(n)}(\mathbf{x})$ depend only on a linear projection of their inputs, as in the sigmoidal activation function $\phi(\mathbf{x}) = \sigma\left(b^{(n)} + W^{(n)}f^{(n-1)}(\mathbf{x})\right)$, then we can simply substitute $f^{(n-1)}(\mathbf{x}) = V^{(n-1)}\phi^{(n-1)}(\mathbf{x})$ to recover $\phi(\mathbf{x}) = \sigma\left(b^{(n)} + W^{(n)}V^{(n-1)}\phi^{(n-1)}(\mathbf{x})\right)$. Thus, we can ignore the intermediate outputs $f^{(n)}(\mathbf{x})$, and exactly recover an MLP with activation functions given by Mercer's theorem, but with rank-D weight matrices between layers!

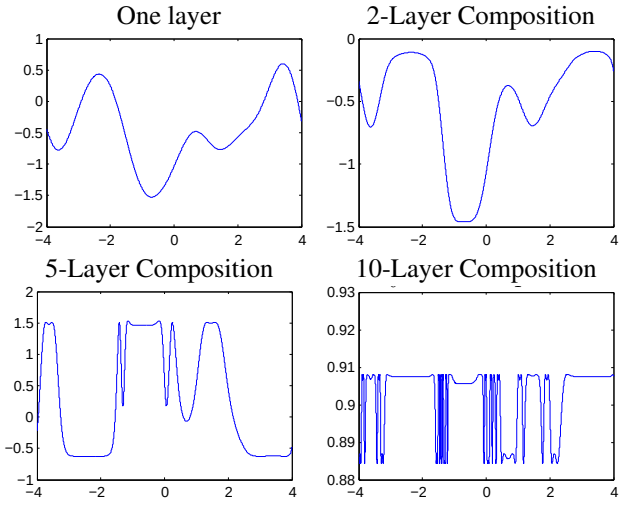


Figure 2: One-dimensional draws from a deep GP prior. After a few layers, the functions begin to be either nearly flat, or highly varying, everywhere. This is a consequence of the distribution on derivatives becoming heavy-tailed.

The second, more general way we can relate the two model classes is to integrate out all $V^{(n)}$, and view deep GP models as a neural network with a finite number of nonparametric, GP-distributed basis functions, where the D outputs of $f^{1:\ell}(\mathbf{x})$ represent the output of the hidden nodes at the ℓ^{th} layer. This second view lets us compare deep GP models to multilayer perceptrons more directly, examining the activations and shapes of the finite number of basis functions at each layer.

3 ONE-DIMENSIONAL ASYMPTOTICS

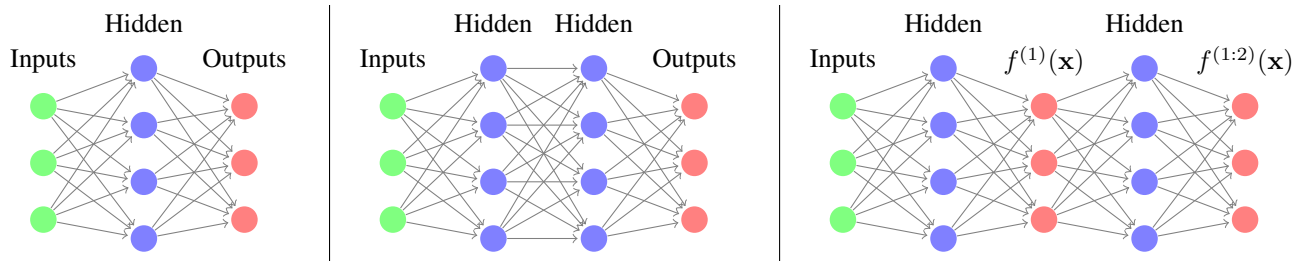
We can understand properties functions drawn from deep GPs and deep networks by looking at the distribution of the derivative of these functions. We first focus on the one-dimensional case.

In this section, we derive the limiting distribution of the derivative of an arbitrarily deep, one-dimensional GP.

The derivative of a GP with a squared-exp kernel

$$k(x, x') = \sigma_f^2 \exp\left(-(x-x')^2/2\ell^2\right) \tag{7}$$

is distributed as $\mathcal{N}\left(0, \sigma_f^2/\ell^2\right)$. By the chain rule, the derivative of a one-dimensional deep GP is simply a product of its (independent) derivatives. The distribution of the absolute value of this derivative is a product of half-normals, each



a) One-hidden-layer MLP b) Two-hidden layer MLP c) Two-layer GP: $\mathbf{y} = f^{(2)}(f^{(1)}(\mathbf{x}))$

Figure 1: Comparing architectures. In the deep GP models, there are two possible meanings for the hidden units. We can consider every other layer to be a linear combination of an infinite number of parametric hidden units. Alternatively, we can integrate out the hidden layers, and consider the deep GP to be a neural network with a finite number of hidden units, each with a different non-parametric activation function.

with mean $\sqrt{2\sigma_f^2/\pi\ell^2}$.

$$\frac{\partial f(x)}{\partial x} \overset{iid}{\sim} \mathcal{N}\left(0, \frac{\sigma_f^2}{\ell^2}\right) \tag{8}$$

$$\implies \left|\frac{\partial f(x)}{\partial x}\right| \overset{iid}{\sim} \text{half}\mathcal{N}\left(\sqrt{\frac{2\sigma_f^2}{\pi\ell^2}}, \frac{\sigma_f^2}{\ell^2}\left(1-\frac{2}{\pi}\right)\right) \tag{9}$$

Thus, if we choose kernel parameters such that $\sigma_f^2/\ell_{d_1}^2 = \pi/2$, then $\mathbb{E}\left[\left|\partial f(x)/\partial x\right|\right] = 1$, and so $\mathbb{E}\left[\left|\partial f^{1:L}(x)/\partial x\right|\right] = 1$, that is to say, the expected magnitude of the derivative remains constants no matter the depth. If σ_f^2/ℓ^2 is less than $\pi/2$, the expected derivative magnitude goes to zero, and if greater, the expected magnitude goes to infinity as a function of L.

The log of the magnitude of the derivatives has moments:

$$m_{\log} = \mathbb{E}\left[\log\left|\frac{\partial f(x)}{\partial x}\right|\right] = 2\log\left(\frac{\sigma_f}{\ell}\right) - \log 2 - \gamma \tag{10}$$

$$v_{\log} = \mathbb{V}\left[\log\left|\frac{\partial f(x)}{\partial x}\right|\right] = \frac{\pi^2}{4} + \frac{\log^2 2}{2} - \gamma^2 - \gamma\log 4 + 2\log\left(\frac{\sigma_f}{\ell}\right)\left[\gamma + \log 2 - \log\left(\frac{\sigma_f}{\ell}\right)\right] \tag{11}$$

where $\gamma \approxeq 0.5772$ is Euler's constant. Since the second moment is finite, by the central limit theorem, the limiting distribution of the size of the gradient approaches log-normal as L grows:

$$\log\left|\frac{\partial f^{1:L}(x)}{\partial x}\right| = \sum_{i=1}^{L}\log\left|\frac{\partial f^{i}(x)}{\partial x}\right|$$

$$\implies \log\left|\frac{\partial f^{1:L}(x)}{\partial x}\right| \overset{L\to\infty}{\sim} \mathcal{N}\left(Lm_{\log}, L^2 v_{\log}\right) \tag{12}$$

Even if the expected magnitude of the derivative remains constant, the variance of the log-normal distribution grows without bound as the depth increases. Because the log-normal distribution is heavy-tailed, and its domain is bonded below by zero, the derivative will become very small almost everywhere, with rare but very large jumps.

Figure 2 shows this behavior in a draw from a 1D deep GP prior, at varying depths. This figure also shows that once the derivative in one region of the input space becomes very large or very small, it is likely to remain that way in subsequent layers.

4 THE JACOBIAN OF A DEEP GP IS A PRODUCT OF INDEPENDENT NORMAL MATRICES

We now derive the distribution on Jacobians of multivariate functions drawn from a deep GP prior.

Lemma 4.1. *The partial derivatives of a function mapping $\mathbb{R}^D \to \mathbb{R}$ drawn from a GP prior with a product kernel are independently Gaussian distributed.*

Proof. Because differentiation is a linear operator, the derivatives of a function drawn from a GP prior are also jointly Gaussian distributed. The covariance between partial derivatives w.r.t. input dimensions d_1 and d_2 of vector $\mathbf{x}$ are given by Solak *et al.* (2003):

$$\text{cov}\left(\frac{\partial f(\mathbf{x})}{\partial x_{d_1}}, \frac{\partial f(\mathbf{x})}{\partial x_{d_2}}\right) = \left.\frac{\partial^2 k(\mathbf{x},\mathbf{x}')}{\partial x_{d_1}\partial x'_{d_2}}\right|_{\mathbf{x}=\mathbf{x}'} \tag{13}$$

If our kernel is a product over individual dimensions $k(\mathbf{x},\mathbf{x}') = \prod_d^D k_d(x_d, x'_d)$, as in the case of the squared-exp kernel, then the off-diagonal entries are zero, implying that all elements are independent. □

In the case of the multivariate squared-exp kernel, the co-

variance between derivatives has the form:

$$k_{SE}(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \prod_{d=1}^{D} \left(-\frac{1}{2} \frac{(x_d - x'_d)^2}{\ell_d^2} \right)$$

$$\implies \operatorname{cov}\left(\frac{\partial f(\mathbf{x})}{\partial x_{d_1}}, \frac{\partial f(\mathbf{x})}{\partial x_{d_2}} \right) = \begin{cases} \frac{\sigma_f^2}{\ell_{d_1}^2} & \text{if } d_1 = d_2 \\ 0 & \text{if } d_1 \neq d_2 \end{cases} \quad (14)$$

Lemma 4.2. *The Jacobian of a set of D functions $\mathbb{R}^D \to \mathbb{R}$ drawn independetly from a GP prior with a product kernel is a $D \times D$ matrix of independent Gaussian R.V.'s*

Proof. The Jacobian of the vector-valued function $f(\mathbf{x})$ is a matrix J with elements $J_{ij} = \frac{\partial f_i(\mathbf{x})}{\partial x_j}$. Because we've assumed that the GPs on each output dimension $f_d(\mathbf{x}) \sim \mathcal{GP}$ are independent, it follows that each row of J is independent. Lemma 4.1 shows that the elements of each row are independent Gaussian. Thus all entries in the Jacobian of a GP-distributed transform are independent Gaussian R.V.s. □

Theorem 4.3. *The Jacobian of a deep GP with a product kernel is a product of independent Gaussian matrices.*

Proof. When composing L different functions, we'll denote the *immediate* Jacobian of the function mapping from layer $\ell - 1$ to layer ℓ as $J^\ell(\mathbf{x})$, and the Jacobian of the entire composition of L functions by $J^{1:L}(\mathbf{x})$.

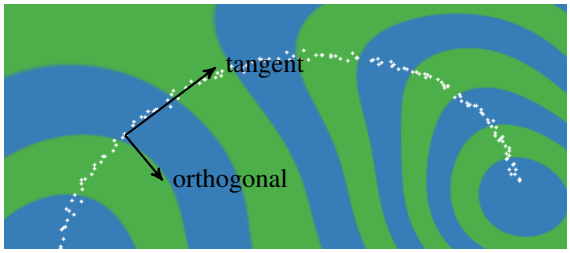
By the multivariate chain rule, the Jacobian of a composition of functions is simply the product of the Jacobian matrices of each function. Thus the Jacobian of the composed (deep) function $f^L(f^{L-1}(\dots f^3(f^2(f^1(\mathbf{x})))\dots))$ is $J^{1:L}(x) = J^L J^{L-1} \dots J^3 J^2 J^1$. By Lemma 4.2, each $J^\ell_{i,j} \overset{\text{iid}}{\sim} \mathcal{N}\left(0, \frac{\sigma_f^2}{\ell^2}\right)$, so the complete Jacobian is a product of independent Gaussian matrices. □

Theorem 4.3 allows us to analyze the representational properties of a deep Gaussian process by simply examining the properties of products of independent Gaussian matrices, a well-studied object.

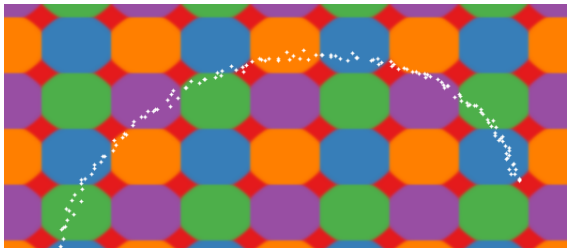
5 FORMALIZING A PATHOLOGY

Rifai *et al.* (2011b) argue that a good latent representation is invariant in directions orthogonal to the manifold on which the data lie. Conversely, a good latent representation must also change in directions tangent to the data manifold, in order to preserve relevant information. Figure 3 visualizes this idea. We follow Rifai *et al.* (2011a) in characterizing the representational properties of a function by the singular value spectrum of the Jacobian[1]. Figure 4 shows the spectrum for 5-dimensional deep GPs of different depths. As the net gets deeper, the largest singular value dominates, implying there is usually only one effective degree of freedom in representation being computed.

[1]Rifai *et al.* (2011a) examine the Jacobian at the training points, but the models we are examining are stationary, so it doesn't matter where we examine the function.



a) A noise-tolerant representation (blue & green) of a one-dimensional manifold (white)



A naïve representation (colors) of a one-dimensional manifold (white)

Figure 3: Comparing representations of data on a 1-D manifold. A representation is a function mapping the input space to some set of outputs. Here, colors show the output of the computed representation. Representation a) is invariant in directions orthogonal to the data manifold, making it robust to noise in those directions, and reducing the number of parameters needed to represent a datapoint. It also changes in directions tangent to the manifold, preserving information for later layers. Representation b) changes in all directions, preserving potentially useless information.

Figure 5 demonstrates a related pathology that arises when composing functions to produce a deep density model. The density in observed space eventually becomes locally concentrated onto one-dimensional manifolds, or *filaments*, implying that such models are insuitable to model manifolds of greater than one dimension.

To visualize this pathology in another way, figure 6 illustrates the value that at each point in the input space is mapped to after successive warpings. After 40 warpings, we can see that locally, there is usually only one direction that one can move in $\mathbf{x}$-space in order to change the value of the function.

To what extent are these pathologies present in nets being used today? In simulations, we found that for deep functions with a fixed latent dimension D, the singular value spectrum remained relatively flat for hundreds of layers as long as $D > 100$. Thus, these pathologies are unlikely to severely affect relatively shallow, wide networks.

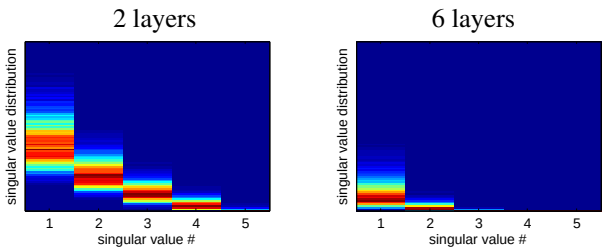


Figure 4: Normalized singular value spectrum of the Jacobian of a deep GP. As the net gets deeper, the largest singular value dominates. This implies that with high probability, there is only one effective degree of freedom in the representation being computed. As depth increases, the distribution on singular values also becomes heavy-tailed.

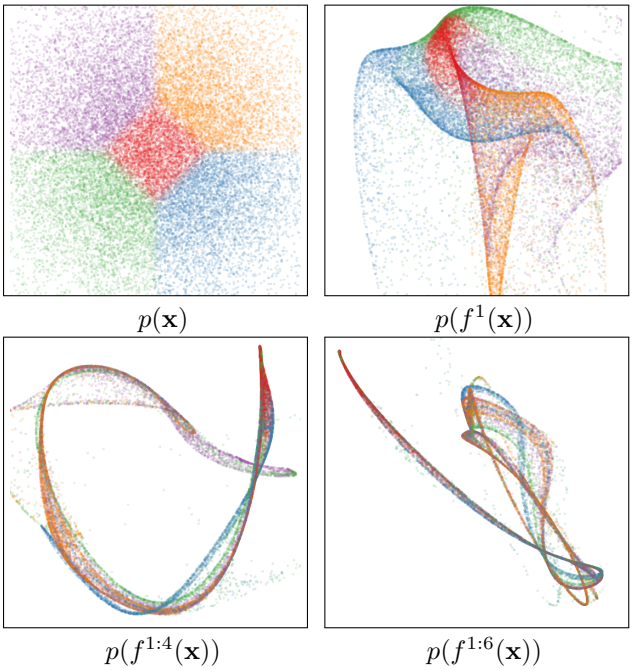


Figure 5: Draws from a deep GP. A distribution is warped by successive functions drawn from a GP prior. As the number of layers increases, the density concentrates along one-dimensional filaments.

6 FIXING THE PATHOLOGY

Following a suggestion from Neal (1995), we can fix the pathologies exhibited in figures 5 and 6 by simply making each layer depend not only on the output of the previous layer, but also on the original input $\mathbf{x}$. That is, $\forall L, f^{1:L}(\mathbf{x}) = f^L(f^{1:L-1}(\mathbf{x}), \mathbf{x})$. The Jacobian of a composite function which has had the original inputs appended $f_{\text{aug}}(\mathbf{x}) = [f(\mathbf{x}), \mathbf{x}]$, is $\begin{bmatrix} J_f \\ I_D \end{bmatrix}$ and the Jacobian of all one-layer transformations of these augmented functions are $D \times 2D$ matrices. Draws from the resulting prior are shown in figures 7, 8 and 10.

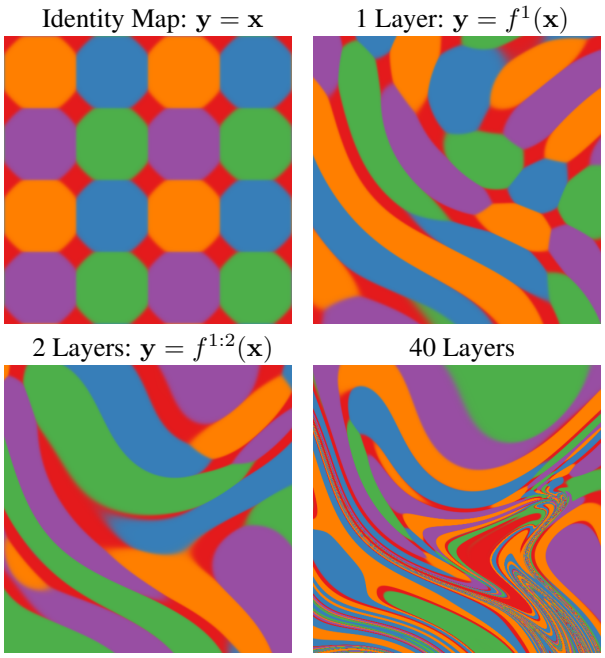


Figure 6: Feature Mapping of a deep GP. Colors correspond to the location $\mathbf{y} = f(\mathbf{x})$ that each point is mapped to after being warped by a deep GP. Just as the densities in figure 5 became locally one-dimensional, there is locally only one direction that one can move $\mathbf{x}$ in to change $\mathbf{y}$. The number of directions in which the color changes rapidly corresponds to the number of large singular values in the Jacobian.

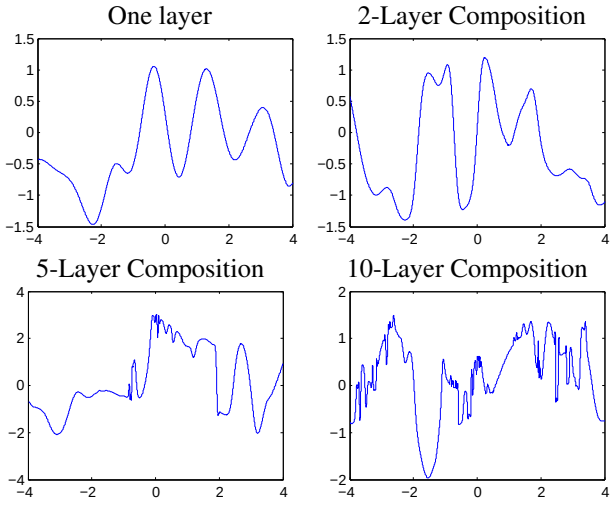
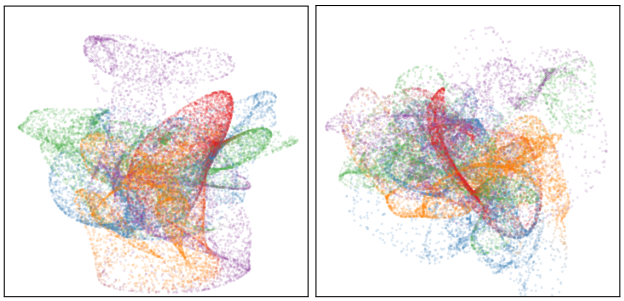


Figure 7: Draws from a 1D deep GP prior with each layer connected to the input. Even after many layers, the functions remain smooth in some regions, while varying rapidly in other regions.

Jacobians of Connected Deep Networks We can similarly examine the Jacobians of the new connected architecture. The Jacobian of the composed, connected



4 Layers 5 Layers

Figure 8: Left: Densities defined by a draw from a deep GP, with each layer connected to the input $\mathbf{x}$. As depth increases, the density becomes more complex without concentrating along filaments.

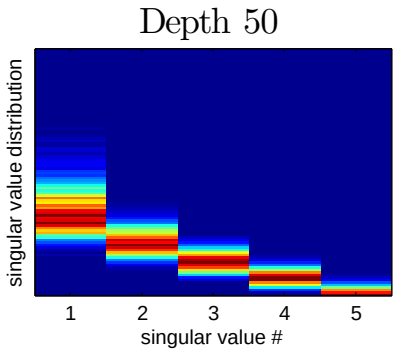


Figure 9: The singular value spectrum of a 5-dimensional deep GP prior 50 layers deep. The singular values remain roughly the same scale.

deep function is defined by the recurrence: $J^{1:L}(\mathbf{x}) = J^L \begin{bmatrix} J^{1:L-1} \\ I_D \end{bmatrix}$. Figure 9 shows that with this architecture, even 50-layer deep GPs have well-behaved singular value spectra.

7 ARBITRARILY DEEP KERNELS

Cho (2012) investigated kernels constructed by applying multiple layers of feature mappings. That is to say, if a kernel has the form $k_1(\mathbf{x}, \mathbf{x}') = \Phi(\mathbf{x})^\mathsf{T}\Phi(\mathbf{x}')$, then we can consider constructing a kernel based on the repeated feature mapping: $k_2(\mathbf{x}, \mathbf{x}') = k_2(\Phi(\mathbf{x}), \Phi(\mathbf{x}')) = \Phi(\Phi(\mathbf{x}))^\mathsf{T}\Phi(\Phi(\mathbf{x}'))$.

For the squared-exp kernel, this composition operation has

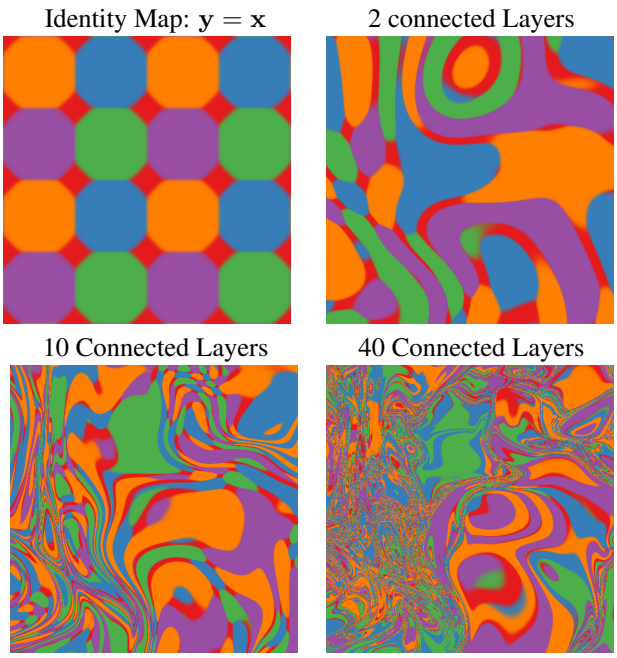


Figure 10: Feature Mapping of a deep GP with each layer connected to the input $\mathbf{x}$. Just as the densities in figure 8 became remained locally two-dimensional even after many transformations, in this mapping there are locally usually two directions that one can move in $\mathbf{x}$ to change $\mathbf{y}$.

a closed form:

$$
\begin{aligned}
&k_{n+1}(\mathbf{x}, \mathbf{x}') = \\
&= \exp\left(-\frac{1}{2}||\Phi_n(\mathbf{x}) - \Phi_n(\mathbf{x}')||_2^2\right) && (15)\\
&= \exp\left(-\frac{1}{2}\left[k_n(\mathbf{x}, \mathbf{x}) - 2k_n(\mathbf{x}, \mathbf{x}') + k_n(\mathbf{x}', \mathbf{x}')\right]\right) \\
&= \exp\left(k_n(\mathbf{x}, \mathbf{x}') - 1\right) \qquad (\text{if } k_n(\mathbf{x}, \mathbf{x}) = 1) && (16)
\end{aligned}
$$

Note that this result holds for any base kernel k_n, as long as $k_n(\mathbf{x}, \mathbf{x}) = 1$.

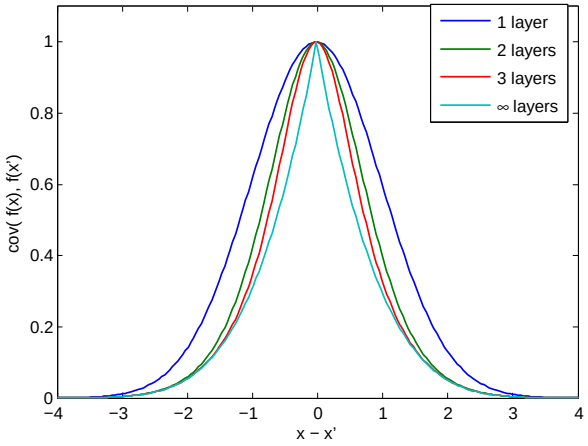
Infinitely Deep Kernels What happens when we apply this composition many times, starting with the squared-exp kernel? In the infinite limit, this recursion converges to $k(\mathbf{x}, \mathbf{x}') = 1$ for all pairs of inputs. One interpretation of why repeated feature transforms lead to this degenerate prior is that each layer can only lose information about the previous set of features. In the limit, the transformed features contain no information about the original input $\mathbf{x}$. Since the function doesn't depend on its input, it must be the same everywhere.

A Non-degenerate Construction Again, following a suggestion from Neal (1995), we connect the inputs $\mathbf{x}$ to each layer. To do so, we simply augment the feature vector

$\Phi_n(\mathbf{x})$ with the $\mathbf{x}$ at each layer:

$$k_{n+1}(\mathbf{x}, \mathbf{x}') = \\ = \exp\left(-\frac{1}{2}\left|\left|\begin{bmatrix}\Phi_n(\mathbf{x})\\ \mathbf{x}\end{bmatrix} - \begin{bmatrix}\Phi_n(\mathbf{x}')\\ \mathbf{x}'\end{bmatrix}\right|\right|_2^2\right) \\ = \exp\left(k_n(\mathbf{x}, \mathbf{x}') - 1 - \frac{1}{2}||\mathbf{x} - \mathbf{x}'||_2^2\right) \quad (17)$$

This kernel satisfies the recurrence $k - \log(k) = 1 + \frac{1}{2}||\mathbf{x} - \mathbf{x}'||_2^2$, a non-degenerate limit. The solution to this recurrence has no closed form, but it is continuous and differentiable everywhere except at $\mathbf{x} = \mathbf{x}'$. Samples from a GP with this prior are not differentiable, having a similar shape to the Ornstein-Ohlenbeck covariance: $\exp(-|x - x'|)$, but with lighter tails.



Connected transform kernel

Figure 11: A non-degenerate version of the infinitely deep feature transform kernel. By connecting the inputs $\mathbf{x}$ to each layer, the function can still depend on its input even after arbitrarily many layers of computation.

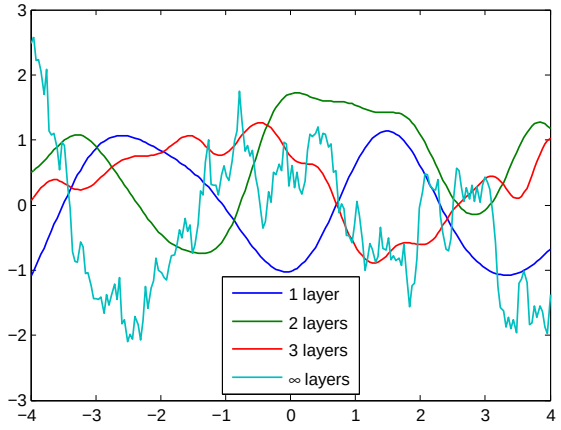


Figure 12: Draws from the deep connected kernel.

7.1 Can Deep Kernels Be Useful Models?

Bengio *et al.* (2006) showed that kernel machines, such as GPs, have limited generalization ability when they use a local kernel such as the squared-exp. However, many interesting non-local kernels can be constructed which allow non-trivial extrapolation, for example, periodic kernels. Periodic kernels can be viewed as a 2-layer-deep kernel, in which the first layer maps $x \to [\sin(x), \cos(x)]$, and the second layer maps through a set of radial-basis functions.

Kernels can capture many other types of generalization, such as translation and rotation invariance in images (Kondor, 2008). Salakhutdinov and Hinton (2008) even used a deep neural network to learn feature transforms for kernels, which learn invariances in an unsupervised manner. In contrast, the relatively uninteresting properties of the kernels derived in this section suggest that an arbitrary deep computation is not necessarily powerful unless combined with learning. For any specific problem, an arbitrary set of computations are unlikely to lead to useful generalization.

8 RELATED WORK

Deep GPs were first proposed by Lawrence and Moore (2007). Variational inference in deep GPs was developed by Damianou and Lawrence (2013), who also analyzed the effect of ARD in these models.

Other variants of deep Bayesian models have been proposed by, for example, Adams *et al.* (2010). Their Deep Bayesian Network architecture has no connections except between adjacent layers, and may also be expected to have similar pathologies as deep GPs as the number of layers increases. Deep Density Networks (Rippel and Adams, 2013) were constructed with invertibility in mind, with penalty terms encouraging the preservation of information about lower layers.

Bengio *et al.* (1994) and Pascanu *et al.* (2012) analyze the exploding-gradients problem in recurrent neural networks. Montavon *et al.* (2010) perform a layer-wise analysis of deep networks, and note that the performance of MLPs degrades as the number of layers with random weights increases.

9 CONCLUSIONS

In this paper, we have shown the following propositions:

- Deep GPs can be viewed as MLPs with a finite number of nonparametric hidden units.
- Deep GPs can be characterized using random matrix theory.

- Representations based on repeated composition of independently-initialized functions exhibit a pathology where the representation becomes invariant to all but one direction of variation.

- Connecting the input to each layer of a deep representation allows us to construct priors on deep functions without this pathology.

This analysis is simply a first step in constructing useful priors over deep functions. Future work could include relating automatic relevance determination to sparse regularization methods, or variational inference in deep GPs to dropout-related regularization.

Acknowledgements

We thank Carl Rasmussen, Andrew McHutchon, Neil Lawrence, Andreas Damianou, James Lloyd, Creighton Heaukulani, Dan Roy and Mark van der Wilk for helpful discussions.

References

Adams, R. P., Wallach, H. M., and Ghahramani, Z. (2010). Learning the structure of deep sparse graphical models. In *International Conference on Artificial Intelligence and Statistics*, pages 1–8.

Bengio, Y., Simard, P., and Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *Neural Networks, IEEE Transactions on*, **5**(2), 157–166.

Bengio, Y., Delalleau, O., and Le Roux, N. (2006). The curse of highly variable functions for local kernel machines. pages 107–114.

Cho, Y. (2012). *Kernel methods for deep learning*. Ph.D. thesis, University of California, San Diego.

Damianou, A. and Lawrence, N. (2013). Deep Gaussian processes. In *Artificial Intelligence and Statistics*, pages 207–215.

Kondor, I. R. (2008). *Group theoretical methods in machine learning*. Ph.D. thesis, Columbia University.

Lawrence, N. D. and Moore, A. J. (2007). Hierarchical Gaussian process latent variable models. In *Proceedings of the 24th international conference on Machine learning*, pages 481–488. ACM.

Lee, H., Ekanadham, C., and Ng, A. (2007). Sparse deep belief net model for visual area v2. In *Advances in neural information processing systems*, pages 873–880.

Martens, J. (2010). Deep learning via Hessian-free optimization. In *Proceedings of the 27th International Conference on Machine Learning*, pages 735–742.

Montavon, G., Braun, D., and Müller, K.-R. (2010). Layer-wise analysis of deep networks with Gaussian kernels. *Advances in Neural Information Processing Systems*, **23**, 1678–1686.

Neal, R. M. (1995). *Bayesian learning for neural networks*. Ph.D. thesis, University of Toronto.

Pascanu, R., Mikolov, T., and Bengio, Y. (2012). Understanding the exploding gradient problem. *arXiv preprint arXiv:1211.5063*.

Poon, H. and Domingos, P. (2011). Sum-product networks: A new deep architecture. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 689–690. IEEE.

Rifai, S., Vincent, P., Muller, X., Glorot, X., and Bengio, Y. (2011a). Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the 28th International Conference on Machine Learning*, pages 833–840.

Rifai, S., Mesnil, G., Vincent, P., Muller, X., Bengio, Y., Dauphin, Y., and Glorot, X. (2011b). Higher order contractive auto-encoder. In *Machine Learning and Knowledge Discovery in Databases*, pages 645–660. Springer.

Rippel, O. and Adams, R. P. (2013). High-dimensional probability estimation with deep density models. *arXiv preprint arXiv:1302.5125*.

Salakhutdinov, R. and Hinton, G. (2008). Using deep belief nets to learn covariance kernels for Gaussian processes. In *Advances in Neural Information Processing Systems*, volume 20.

Solak, E., Murray-Smith, R., Solak, E., Leithead, W., Rasmussen, C., and Leith, D. (2003). Derivative observations in Gaussian process models of dynamic systems.