Accelerating Deep Gaussian Processes Inference with Arc-Cosine Kernels

Kurt Cutajar
EURECOM, France
kurt.cutajar@eurecom.fr

Edwin V. Bonilla
University of New South Wales, Australia
e.bonilla@unsw.edu.au

Pietro Michiardi
EURECOM, France
pietro.michiardi@eurecom.fr

Maurizio Filippone
EURECOM, France
maurizio.filippone@eurecom.fr

Abstract

Deep Gaussian Processes (DGPs) are probabilistic deep models obtained by stacking multiple layers implemented through Gaussian Processes (GPs). Although attractive from a theoretical point of view, learning DGPs poses some significant computational challenges that arguably hinder their application to a wider variety of problems for which Deep Neural Networks (DNNs) are the preferred choice. We aim to bridge the gap between DGPs and DNNs by showing how random feature approximations to DGPs can leverage the key strengths of DNNs, while retaining a probabilistic formulation for accurate quantification of uncertainty. In particular, we show how DGPs with arc-cosine kernels can be approximated by DNNs with Rectified Linear Unit (ReLU) activation functions, leading to competitive performance and faster inference compared to state-of-the-art DGPs inference approaches.

1 Introduction

Deep Gaussian Processes (DGPs) [5] are deep probabilistic models obtained by stacking multiple layers implemented through Gaussian Processes (GPs) [13]. This construction to model composition of functions naturally resembles that of Deep Neural Networks (DNNs). The connection between DGPs and DNNs has been extensively investigated in the literature, and DGPs with a variety of kernel functions can be interpreted as DNNs with an infinite number of neurons at each layer and specific activation functions (see, e.g., [6, 11]). In contrast to DNNs, DGPs provide an elegant way to deal with the model-selection problem of determining a suitable number of neurons, as they are inherently nonparametric learning machines. Furthermore, they allow for a principled probabilistic framework to carry out learning of latent representations and hyper-parameters.

Although attractive from a theoretical point of view, learning DGPs poses some significant computational challenges that arguably hinder their application to a wider variety of problems. In contrast, DNNs have been extremely successful in areas such as computer vision because of their amenability to GPU and distributed computations, automatic differentiation tools, and mature developments of regularization techniques, such as low-rank weight representations and dropout [14]. We aim to bridge the gap between DGPs and DNNs by showing how DGPs can be learned at scale by borrowing the key strengths of DNNs, while retaining a probabilistic formulation for accurate quantification of uncertainty. We made a significant step in this direction in [4], by showing how random Fourier feature approximations for DGPs with Radial Basis Function (RBF) kernels lead to trigonometric DNNs with low-rank weight matrices. We conveniently implemented stochastic variational inference [8] in TensorFlow [1] to infer the parameters of such approximate DGPs.

In this paper, we aim to take this idea one step further by showing how DGPs with arc-cosine kernels can be approximated by DNNs with Rectified Linear Unit (ReLU) activations. Such activation functions are much cheaper to evaluate and differentiate than the trigonometric functions used in the approximation of DGPs with RBF kernels, leading to faster DGP inference. We experimentally validate this acceleration effect, while showing that the proposed approximation leads to competitive performance compared with the approximation using trigonometric activations.

2 Random features for Deep Gaussian Processes with Arc-cosine Kernels

Let $X = [\mathbf{x}_1, \ldots, \mathbf{x}_n]^\top$ be a set of input vectors associated with a set of labels $Y = [\mathbf{y}_1, \ldots, \mathbf{y}_n]^\top$, where $\mathbf{x}_i \in R^{D_{\text{in}}}$ and $\mathbf{y}_i \in R^{D_{\text{out}}}$. Let $F^{(0)} := X$ and consider a DGP where we define $F^{(l)}$ as the multivariate latent representation at layer l, which is a collection of zero-mean GPs with covariance $k(\cdot, \cdot; \boldsymbol{\theta})$. Each layer of DGPs can be approximated based on a finite set of basis functions $F^{(l+1)} = \Phi^{(l)} W^{(l)}$, leading to a DNN representation of DGPs [4]. The set of basis functions $\Phi^{(l)}$ and the distribution of $W^{(l)}$ depend on the choice of the kernel (covariance) function. For example, in the RBF case, Bochner's theorem suggests an expansion using trigonometric functions as basis functions [12] and $W_{ij}^{(l)} \sim \mathcal{N}(0, 1)$.

In [4], we explored the random feature approximation of RBF kernels using trigonometric functions. Here we propose arc-cosine kernels to accelerate learning of DGPs with the random features flavor. The arc-cosine kernel of order p is defined as:

$$k_{\text{arc}}^{(p)}(\mathbf{x}, \mathbf{x}') = \frac{1}{\pi} \|\mathbf{x}\|^p \|\mathbf{x}'\|^p J_p\left(\cos^{-1}\left(\frac{\mathbf{x}^\top \mathbf{x}'}{\|\mathbf{x}\|\|\mathbf{x}'\|}\right)\right) \quad \text{with} \tag{1}$$

$$J_p(\alpha) = (-1)^p (\sin\alpha)^{2p+1} \left(\frac{1}{\sin\alpha}\frac{\partial}{\partial\alpha}\right)^p \left(\frac{\pi - \alpha}{\sin\alpha}\right). \tag{2}$$

Let $\Theta(\cdot)$ be the Heaviside function. Following [3], an integral representation of this kernel is:

$$k_{\text{arc}}^{(p)}(\mathbf{x}, \mathbf{x}') = 2 \int \Theta(\boldsymbol{\omega}^\top \mathbf{x}) \, \left(\boldsymbol{\omega}^\top \mathbf{x}\right)^p \, \Theta(\boldsymbol{\omega}^\top \mathbf{x}') \, \left(\boldsymbol{\omega}^\top \mathbf{x}'\right)^p \, \mathcal{N}(\boldsymbol{\omega}|0, I) \, d\boldsymbol{\omega} \tag{3}$$

This integral formulation immediately suggests a random feature approximation for equation (1), noticing that the kernel can be seen as an expectation of the product of the same function applied to the inputs to the kernel. We can extend this by scaling the kernel with a σ^2 parameter, and features through $\Lambda = \text{diag}(l_1^2, \ldots, l_d^2)$ for automatic relevance determination [9]. In the case $p = 1$, the application of the random feature approximation of DGPs with arc-cosine kernels leads to DNNs with ReLU activations:

$$\Phi_{\text{arc}}^{(l)} = \sqrt{\frac{2(\sigma^2)^{(l)}}{N_{\text{RFF}}^{(l)}}} \max\left(0, F^{(l)}\Omega^{(l)}\right), \qquad \text{with} \quad \Omega_{\cdot j}^{(l)} \sim \mathcal{N}\left(\mathbf{0}, \left(\Lambda^{(l)}\right)^{-1}\right), \tag{4}$$

which are cheaper to evaluate and differentiate compared to trigonometric functions in the RBF case. Similarly to [4], we learn an approximate posterior over all weight matrices $\Omega^{(l)}$ and $W^{(l)}$ and optimize kernel parameters $(\sigma^2)^{(l)}$ and $\Lambda^{(l)}$ using stochastic variational inference.

3 Results

We follow the same experimental setup of [4], where we evaluate the performance of DGP approaches on regression and classification tasks. We compare the approximate DGP with arc-cosine kernel (DGP-ARC) with the approximate DGP with RBF kernel (DGP-RBF) [4], and the recently proposed DGP model approximated using expectation propagation (DGP-EP) [2]. All models exploit mini-batch-based stochastic gradient optimization. To ensure fairness in the comparison, we use the same configurations across all models, namely one three-dimensional hidden layer, number of inducing points/number of random features equal to 100 and same mini-batch size. An exception is made for the MNIST case, where greater dimensionality and more inducing points/random features are required (see [4]). We evaluate the error rate and mean negative log-likelihood (MNLL) on withheld

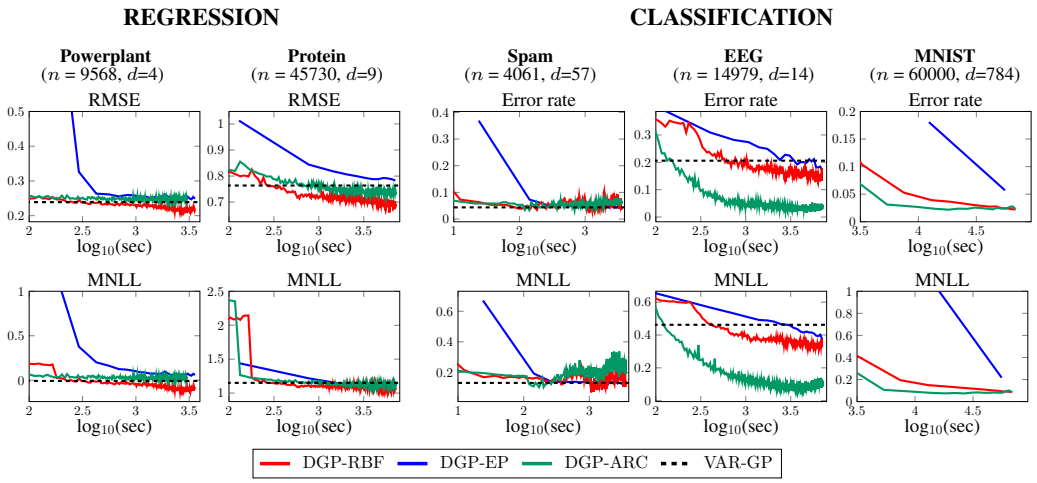

Figure 1: Progression of RMSE and MNLL on test data over time for competing DGP models. Dashed line indicates the accuracy obtained by the sparse variational GP [7] implemented in GPflow [10].

test data, and the results are averaged over 3 folds, except for the MNIST dataset where we used the usual split between training and test set.

Figure 1 shows that, across all datasets, DGP-ARC outperforms other approaches in terms of convergence speed, while leading to error values which are comparable to those obtained using DGP-RBF (or considerably better in the case of the EEG dataset). While not illustrated in the figures, we also assessed the performance of the arc-cosine kernel with degree two, but this case performed poorly compared to the arc-cosine kernel of degree one. Finally, we note that for the the arc-cosine kernel with degree zero, the activation functions are Heaviside functions that are unsuitable for our inference scheme given that they yield systematically zero gradients.

Summarizing, we regard these results as a further step in the direction of bridging the gap between DGPs and DNNs, hoping that this will foster a wider adoption of DGPs in applications.

Acknowledgements

MF gratefully acknowledges support from the AXA Research Fund.

References

[1] M. Abadi, A. Agarwal, P. Barham, et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015.

[2] T. D. Bui, D. Hernández-Lobato, J. M. Hernández-Lobato, Y. Li, and R. E. Turner. Deep Gaussian Processes for Regression using Approximate Expectation Propagation. In M. Balcan and K. Q. Weinberger, editors, *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1472–1481. JMLR.org, 2016.

[3] Y. Cho and L. K. Saul. Kernel methods for deep learning. In Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009. Proceedings of a meeting held 7-10 December 2009, Vancouver, British Columbia, Canada.*, pages 342–350. Curran Associates, Inc., 2009.

[4] K. Cutajar, E. V. Bonilla, P. Michiardi, and M. Filippone. Practical learning of deep Gaussian processes via random Fourier features, 2016. arXiv:1610.04386.

[5] A. C. Damianou and N. D. Lawrence. Deep Gaussian Processes. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2013, Scottsdale, AZ, USA, April 29 - May 1, 2013*, volume 31 of *JMLR Proceedings*, pages 207–215. JMLR.org, 2013.

[6] D. K. Duvenaud, O. Rippel, R. P. Adams, and Z. Ghahramani. Avoiding pathologies in very deep networks. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, AISTATS 2014, Reykjavik, Iceland, April 22-25, 2014*, volume 33 of *JMLR Workshop and Conference Proceedings*, pages 202–210. JMLR.org, 2014.

[7] J. Hensman, A. G. de G. Matthews, and Z. Ghahramani. Scalable variational Gaussian process classification. In G. Lebanon and S. V. N. Vishwanathan, editors, *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2015, San Diego, California, USA, May 9-12, 2015*, volume 38 of *JMLR Workshop and Conference Proceedings*, pages 351—-360. JMLR.org, 2015.

[8] D. P. Kingma and M. Welling. Auto-Encoding Variational Bayes. In *Proceedings of the Second International Conference on Learning Representations (ICLR 2014)*, Apr. 2014.

[9] D. J. C. Mackay. Bayesian methods for backpropagation networks. In E. Domany, J. L. van Hemmen, and K. Schulten, editors, *Models of Neural Networks III*, chapter 6, pages 211–254. Springer, 1994.

[10] A. G. d. G. Matthews, M. van der Wilk, T. Nickson, K. Fujii, A. Boukouvalas, P. León-Villagrá, Z. Ghahramani, and J. Hensman. GPflow: A Gaussian process library using TensorFlow. *arXiv preprint 1610.08733*, Oct. 2016.

[11] R. M. Neal. *Bayesian Learning for Neural Networks (Lecture Notes in Statistics)*. Springer, 1 edition, Aug. 1996.

[12] A. Rahimi and B. Recht. Random Features for Large-Scale Kernel Machines. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1177–1184. Curran Associates, Inc., 2008.

[13] C. E. Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[14] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.