Raiders of the Lost Architecture: A Kernel for Conditional Parameter Spaces

Kevin Swersky
University of Toronto
kswesrky@cs.utoronto.edu

David Duvenaud
University of Cambridge
dkd23@cam.ac.uk

Jasper Snoek
Harvard University
jsnoek@seas.harvard.edu

Frank Hutter
Freiburg University
fh@informatik.uni-freiburg.de

Michael A. Osborne
University of Oxford
mosb@robots.ox.ac.uk

Abstract

When performing model-based optimization, we must often search over structures with differing numbers of parameters. For instance, we may wish to search over neural network architectures with an unkown number of layers. To combine information between different architectures, we define a family of kernels for conditional parameter spaces.

1 Introduction

Recently, Bayesian optimization has been used to learn parameters for neural networks [1]. Because different neural net architectures have different numbers of parameters (or their parameters have different meanings), it is not clear how to combine information or extrapolate between the performance of different architectures. Historically, practitioners have simply built a separate model for each type of architecture [2]. However, if there is any relation between networks with different architectures, seperately modeling each architecture is wasteful.

Bayesian optimization methods rely on constructing a model of the function being optimized. We model this function using Gaussian process priors [3], where model assumptions are defined through the kernel. In this paper, we introduce a simple method for building a Gaussian process model over parameter spaces with varying numbers of parameters. We do this by defining an embedding of datapoints depending on which parameters are active, then perform standard regression in this new space.

2 A Kernel for Conditional Parameter Spaces

3 Experiments

Regression Bayesian optimization requires building a model of the function being optimized, and better models can be expected to lead to better outcomes. However, because of the many interating components of BO, optimizer performance might not correspond directly to the quality of the model. In this section, we isolate to what extent the conditional kernel is a better model than the alternatives.

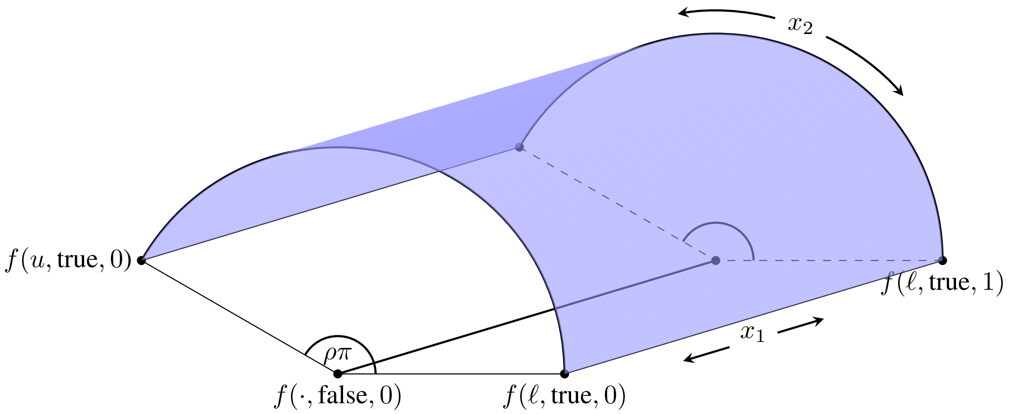
x_2
$f(u, \text{true}, 0)$
$f(\ell, \text{true}, 1)$
x_1
$\rho\pi$
$f(\cdot, \text{false}, 0)$
$f(\ell, \text{true}, 0)$

Figure 1: A demonstration of the embedding giving rise to the pseduo-metric in 2 dimensions. All points for which $\delta_i(x) =$ false are mapped onto a line varying only along x_1. Points for which $\delta_i(x) =$ true are mapped to the surface of a semicylinder, depending on both x_1 and x_2. This embedding gives a constant distance between pairs of points which have differing values of δ but the same values of x_1. The parameter ρ determines how much distance there is along the arc.

Table 1: Normalized Mean Squared Error on Neural Network data

Method	NN	NN log	NN half	NN log half
Separate Linear	**0.968**	0.886	**1.039**	2.120
Separate GP	**0.925**	0.641	**0.860**	0.848
Poor Man's embedding Linear	**0.905**	0.763	0.996	0.851
Poor Man's embedding GP	**0.907**	0.518	**1.178**	**0.752**
Separate Hierarchical GP	**0.801**	0.627	**0.956**	0.950
Hierarchical GP	**0.801**	**0.441**	**0.993**	**0.674**

Separate Linear and Seperate GP split the data into 6 different datasets, one for each level, and build a separate model for each level. The gp-hierarchical model gets all the data together because it can handle it.

However, The gp-hierarchical model changes two things at once compared to the separate-gp-ard model: besides embedding the missing data in a different spot, it also embeds the fully-observed data on semi-circles, and has a different parameterization. So, it could be the case that even when the data are fully observed, embedding the data on a semi-circle and using a different parameterization might cause better or worse performance than a standard squared-exp. To find out if this is the case, we compare separate-gp-ard and separate-hierarchical to find out if these two models have different performance even in the standard fully-observed case.

4 Conclusion

References

[1] Jasper Snoek, Hugo Larochelle, and Ryan Prescott Adams. Practical bayesian optimization of machine learning algorithms. In *Neural Information Processing Systems*, 2012.

[2] James Bergstra, Rémi Bardenet, Yoshua Bengio, Balázs Kégl, et al. Algorithms for hyper-parameter optimization. In *25th Annual Conference on Neural Information Processing Systems (NIPS 2011)*, 2011.

[3] C.E. Rasmussen and CKI Williams. Gaussian Processes for Machine Learning. *The MIT Press, Cambridge, MA, USA*, 2006.