AML-CW-2
Multi-output Gaussian Process Regression

Youngbin Lee
Department of Industrial Engineering
young@unist.ac.kr

Abstract

Gaussian process (GP) regression is a nonparametric, Bayesian approach to regression where the prediction is probablistic. Because the model output is a probablistic distribution, one can compute empirical confidence intervals. In this work, GP regression will be formulated for a multi-output (i.e., Vector-valued) model. Then, models with various settings (e.g., Hyperparameters, Kernels) will be implemented to predict robot arm torques with SARCOS dataset.

1 Introduction

In this project, GP regression will make predictions on robot arm torques from SARCOS dataset which consists of 21 input features (e.g., joint positions, velocities, and accelerations) and 7 output features (e.g., torques). And because output dimension is more than one, this is a multi-output (or vector-valued) regression model. A multi-output model assumes output values are related to each other and we further assume that all input features affect every output features. In other words, if we say dataset X_i is a set of input data for predicting i^{th} output feature from $y \in R^D$, $X_1 = ... = X_i = ... = X_D$.

2 Formulation

2.1 Gaussian process regression

The idea of Bayesian linear regression is to directly make prediction for a given input, without building any function between input and output (e.g., $f(x) = w^T x$) and thus finding an optimal parameter w^*. This can be done by applying the marginalization of w to the prediction posterior (or predictive distribution). The parameter posterior $p(w|y, X)$ depends on the likelihood $p(y|w, X)$ and the prior $p(w|X)$. If an independent and identically distributed (i.i.d.) Gaussian noise model is assumed for f, then the prior $p(w)$ is a conjugate for Gaussian likelihood and posterior is also a Gaussian distribution.

Prediction posterior of Bayesian linear regression:

$$p(y'|x', y, X) = \int p(y'|x', w)p(w|y, X)dw \tag{1}$$

If this idea expands to non-parametric regression, the model assumption (e.g., $f(x) = w^T x$) is removed and function f can be used as a variable. In this case, the marginalization of f (instead of w) is applied to the prediction posterior. Now, the function posterior $p(f|y, X)$ depends on the prior $p(f|X)$ and this is where the Gaussian process kicks in. This is because we need a Gaussian distribution on the space of functions which is an infinite space. And this is why the model is called GP regression.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

Prediction posterior of GP regression:

$$p(y'|x', y, X) = \int p(y'|x', f)p(f|y, X)df \tag{2}$$

GP regression can now be divided into linear model and non-linear model. In non-linear model, the idea is to map input data x to a feature space F using a non-linear map ϕ and build a linear regressor in F. In this case, kernel functions like Gaussian kernel can be used to efficiently calculate inner-product form of mapped inputs (e.g., $\phi(x)^T\phi(x)$) from the prediction posterior $p(y'|x', y, X)$. On the other hand, linear model uses an identity map ϕ and thus uses a linear kernel such as a standard dot product kernel.

Linear regressor with feature map ϕ :

$$f(x) = w^T\phi(x) \tag{3}$$

For linear and non-linear GP regression model, predictive distribution is formulated as follow. Here, M^* denotes the mean prediction and V^* indicates the predictive covariance.

Predictive mean:

$$M^* = K(X', X)_{MN}\{K(X, X)_{NN} + \sigma^2 I_{NN}\}^{-1} vecY_N \tag{4}$$

Predictive covariance:

$$V^* = K(X', X')_{MM} - K(X', X)_{MN}\{K(X, X)_{NN} + \sigma^2 I_{NN}\}^{-1} K(X, X')_{NM} \tag{5}$$

Notation K indicates a covariance function (or kernel function) and subscript denotes shape of a matrix. For example, $K(X, X')_{NM}$ is a covariance matrix between test dataset and train dataset with a shape of (N, M). And N, M are the numbers of train samples and test samples, respectively.

2.2 Multi-output Gaussian process regression

Gaussian processes for multi-output learning can be formulated with Linear Model of Coregionalization (LMC). Considering a set of D outputs, the kernel matrix corresponding to a dataset X now takes the form below, where C_{DD} is known as a coregionalization matrix and $\bigotimes$ is a Kronecker product. In this project, C_{DD} is an identity matrix because we assume that the kernels for each output feature are independent one another.

Kernel matrix under LMC:

$$C_{DD} \bigotimes K(X, X)_{NN} \tag{6}$$

With a set of D outputs and corresponding coregionalization matrix C, the predictive distribution is reformulated as follow.

Predictive mean:

$$M^* = \{C_{DD} \bigotimes K(X', X)_{MN}\}\{C_{DD} \bigotimes K(X, X)_{NN} + \Sigma_{DD} I_{NN}\}^{-1} vecY_{ND} \tag{7}$$

Predictive covariance:

$$V^* = \{C_{DD} \bigotimes K(X', X')_{MM}\} - \{C_{DD} \bigotimes K(X', X)_{MN}\}$$
$$\{C_{DD} \bigotimes K(X, X)_{NN} + \Sigma_{DD} I_{NN}\}^{-1}\{C_{DD} K(X, X')_{NM}\}$$

2.3 Hyperparameter optimization

In this project, a zero-mean i.i.d. Gaussian noise model is used for f and a zero-mean GP prior is put on f. That is, from the prediction posterior of GP regression, the following equation holds.

Prediction posterior of GP regression:

$$
\begin{aligned}
&p(f|y,X) \propto p(y|X,f)p(f|X) \\
&where \\
&p(y|X,f) = N(f, \sigma^2 I) \\
&p(f|X) = N(0, K(X,X))
\end{aligned}
$$

For kernel functions, a standard dot product kernel and an isotropic Gaussian kernel are used for linear model and non-linear model, respectively. In addition, a Sigmoid kernel is used for non-linear model to make comparisons.

Dot product kernel:

$$k(x,x') = x^T x' \tag{8}$$

Gaussian kernel:

$$k(x,x') = exp(-\frac{||x-x'||^2}{\sigma_k^2}) \tag{9}$$

Sigmoid kernel:

$$k(x,x') = tanh(\alpha x^T x') \tag{10}$$

From equations 9 and 10, hyperparameters are σ_k^2 and α, respectively. And these hyperparameters can be optimized by maximizing the marginal likelihood as the following.

$$argmax_{\sigma_n^2,\sigma_k^2} p(y|\Phi) = \int p(y|\Phi, w)p(w)dw \tag{11}$$

where Φ is a mapped dataset, σ_n^2 is a noise variance, and σ_k^2 is a kernel variance.

This can be reformulated as minimizing negative log marginal likelihood (NLL) as the following.

$$argmin_{\sigma_n^2,\sigma_k^2} NLL = -\frac{ND}{2}ln(2\pi) - \frac{1}{2}ln|K| - \frac{1}{2}vecY^T K^{-1} vecY \tag{12}$$

where $K = C_{DD} \bigotimes K(X,X)_{NN} + \Sigma_{DD} \bigotimes I_{NN}$

Taking partial derivatives of NLL with respect to each hyperparameter, a gradient descent can be applied to update hyperparameters. In experiment, however, hyperparameters are optimized by a grid search with a separate validation dataset. For the search range, refer to the figure 1

Model	Kernel	Hyperparameter	Range
Linear	Dot product	σ_n^2	[0.05, 0.1, 1, 5, 10]
Non-linear	Gaussian	σ_n^2	[0.05, 0.1, 1, 5, 10]
		σ_k^2	[0.05, 1, 5]
Non-linear	Sigmoid	σ_n^2	[0.05, 0.1, 1, 5, 10]
		α	[0.05, 1, 5]

Figure 1: Hyperparameter Search Range

3 Experiment

3.1 Subset of Regressors (SOR) approximation

In experiment, a SOR approximation is conducted due to the computational cost of a full GP regression. Thus, a basis set $\{b_1, ..., b_n\}$ with $n << N$ is randomly selected from original train dataset and we used 10% of data (i.e., $n = N * 0.1$). Then, the predictive mean of SOR becomes the following.

Predictive mean:

$$M^* = \{C_{DD} \bigotimes K(X', X_B)_{Mn}\}$$
$$[C_{DD} \bigotimes \{K(X_B, X)_{nN} K(X, X_B)_{Nn}\} + \{\Sigma_{DD} K(X_B, X_B)_{nn}\}]^{-1}$$
$$\{C_{DD} \bigotimes K(X_B, X)_{nN} vecY_{ND}\}$$

where X_B is a basis set with n samples that are sampled from the original train set X. And thus, for example, $K(X', X_B)_{Mn}$ is a kernel matrix between test dataset and basis set with a shape of (M, n).

3.2 Result and Discussion

Hyperparameter Optimization (HPO) In figure 2, 3, 4, runtime and loss are presented with respect to each hyperparameter combination. The runtime is measured in microsecond and the loss is calculated as a mean squared error (MSE) on a validation dataset. The lowset loss is shown in bold to figure out optimized case of hyperparameters.

Model	Kernel	σ_n^2	Runtime	MSE loss
Linear	Dot product	0.05	373195	13070863
		0.1	673963	11031468
		1	310300	**60396**
		5	553296	288277964
		10	895313	186614

Figure 2: HPO: Linear model with Dot product kernel

Model	Kernel	σ_n^2	σ_k^2	Runtime	MSE loss
Non-linear	Gaussian	0.05	0.05	468432	370
		0.1	0.05	492986	370
		1	0.05	431557	370
		5	0.05	177430	370
		10	0.05	759776	370
		0.05	1	293153	334
		0.1	1	440249	333
		1	1	553447	339
		5	1	485533	344
		10	1	879994	347
		0.05	5	318610	**64**
		0.1	5	196211	65
		1	5	821603	73
		5	5	797869	93
		10	5	492172	112

Figure 3: HPO: Non-linear model with Gaussian kernel

From figure 2, the linear model with a dot product kernel showed high prediction loss on all range of σ_n^2. Hyperparameter σ_n^2 is best at the value of 1, as it showed the shortest runtime and lowest loss.

Model	Kernel	σ_n^2	α	Runtime	MSE loss
		0.05	0.05	861084	8
		0.1	0.05	954705	**7**
		1	0.05	227890	**7**
		5	0.05	271570	10
		10	0.05	775042	116
		0.05	1	964304	26
		0.1	1	149503	36
Non-linear	Sigmoid	1	1	240528	32
		5	1	919076	32
		10	1	872059	37
		0.05	5	17034	34
		0.1	5	819889	32
		1	5	965240	41
		5	5	19321	41
		10	5	168143	34

Figure 4: HPO: Non-linear model with Sigmoid kernel

The Figure 3 shows that σ_k^2 with the value of 5 performs best among other values, regardless of the value of σ_n^2. Thus, 5 is clearly the optimzed value for kernel variance. However, runtime seems a bit random (i.e., There is no clear trend with hyperparameter values).

We found that non-linear model with Sigmoid kernel worked best overall among others, shown in Figure 4. The best combination of hyperparameters are $\sigma_n^2 = 1$ and $\alpha = 0.05$ with its MSE loss of 7, which is the lowest out of other combinations. This result may have been from the role of Sigmoid kernel. In other words, the nonlinearity of hyperbolic tangent function worked better with the dataset used in experiment than other kernels.

As a result of hyperparameter search among three different models, it is found that a non-linear GP regression with Sigmoid kernel works best with σ_n^2=1 and $\alpha = 0.05$. And the optimal MSE loss on valid data is 7. In figure 5, we refer this model to 'GP_optimized' and compare its performance on test dataset with another regression model, a Ridge regression. Ridge regression is a regularized linear regression model and a regularization parameter $\alpha = 1$ is used by default from scikit-learn.

Model	Kernel	Hyperparameter	Runtime	MSE loss
GP_optimized	Sigmoid	$\sigma_n^2 = 1$ $\alpha = 0.05$	242429	**7**
Ridge	Gaussian	$\alpha = 1$	43172	10

Figure 5: Comparison with another regression model

It is shown that GP model outperformed Ridge model with a slightly lower loss. This is because Ridge model does not capture a non-linear relationship between inputs and outputs whereas GP model added a nonlinearity by using feature map that is calculated with Sigmoid kernel function. However, GP model takes much more time than Ridge regression as its computation contains calculating kernel matrices and its inverse.

4 Conclusion

We formulated GP regression for multi-dimensional outputs with Linear Model of Coregionalization. We then implemented and compared multi-output GP regression models with different kernels. A subset of regressor approximation is used instead of a full GP model with a high time complexity. As a result of hyperparameter optimization on a validation dataset, it is found that the non-linear GP regression with Sigmoid kernel worked best among others.

References

[1] Rakitsch, B., Lippert, C., Borgwardt, K., Stegle, O. (2013). It is all in the noise: Efficient multi-task Gaussian process inference with structured residuals. Advances in neural information processing systems, 26.

[2] Alvarez, M. A., Rosasco, L., Lawrence, N. D. (2012). Kernels for vector-valued functions: A review. Foundations and Trends® in Machine Learning, 4(3), 195-266.

[3] Rasmussen, C. E. (2006). ChristopherKI. Williams, Gaussian Processes for Machine Learning.