Gaussian Process Regression Network derivation methods for lipidomics and metabolomics analysis

David Stewart

Abstract

Motivation: GPR (Gaussian Process Regression) is an underutilized method for network derivation from -omics datasets. Given its ability to draw significant predictions with relatively few samples and many features, GPR should be investigated as an alternative to popular network derivation techniques.
Results: The tool presented in this paper is designed to identify relationships between features in a dataset based on the strength of the predictions that can be made from a GPR based model. For each pair of features, the first is modeled as a function of the second. Once all pairs have been modeled, we present three techniques for thresholding to determine which pairs should be considered significantly related; Top score, prior known pairs, and dynamic threshold based on prior knowledge. The tool can output a binary matrix of significant pairs, or a weighted matrix for either the significant pairs as determined by thresholding, or for every pair in the sheet.

1 Introduction

Establishing biological networks creates a perspective of biological systems that is useful for studying the behavior of diseases. Investigation of these networks should be carried out in parallel to that of individual molecular studies (Nguyen-Tran *et al.* , 2023). In order to derive biological networks, we require a quantification of the relationships between individual features within that network (Ma'ayan, 2011).

Gaussian process regression is an effective tool for the analysis of biological networks, specifically when applied to -omics datasets. The nature of an -omics dataset, specifically its many features with relatively few samples, make it challenging to analyze via machine learning due to the risk of over or under fitting. GPR lends itself to this application by being non parametric and not requiring a prior distribution, thus making it less likely than other models to assume a certain shape for the data. In addition, a properly selected kernel will create a covariance measure that allows accurate predictions even in areas where source data is sparse (Williams *et al.* *,* 2006).

2 Implementation

The tool is implemented in python. Data is input in a csv format, with column names in the first row, with all following rows containing a concentration measure for a single sample.

Data are Z-score normalized across samples. The tool iterates pairwise through all feature pairs x and y, modeling y as a function of x. Scikit-learn GaussianProcessRegressor is used to model these functions. Using leave-one-out cross validation (LOOCV), the tool creates models using all but one sample for a given feature. Leave-one-out uses all but one data point as its training set, with the holdout being used to evaluate the model. This is demonstrated in figure 1. This process can be repeated for every sample in the data set, or for a specific number of randomly selected samples. In the case that random selection is needed, the randint() method from the random python library is used to generate random indices.

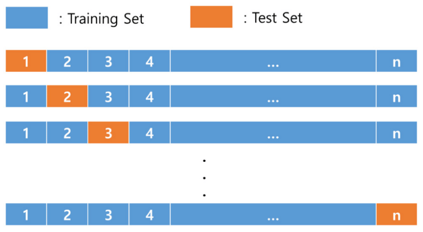

Fig 1: Leave one out cross validation, adapted from Cha *et al.* 2020

The tool uses the sum of three kernels as the kernel for the GaussianProcessRegressor instance; Dot Product, Radial Basis Function (RBF), and WhiteKernel.

$$k(x_i, x_j) = \sigma_0^2 + x_i \cdot x_j$$

$$k(x_i, x_j) = \exp\left(-\frac{d(x_i, x_j)^2}{2l^2}\right)$$

$$k(x_1, x_2) = noise\_level \text{ if } x_i == x_j \text{ else } 0$$

A list of predictions and expected values is created, with each element corresponding to a model, where the element is the holdout from the training set. With a list of predictions and target values for a pair of features, a mean squared error (MSE) is calculated via the following formula.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

This gives a score related to the ability of GPR to accurately model one feature as a function of the other. This is repeated for all possible pairs of features, giving a matrix of mse values representing a weighted graph where all nodes are connected to all other nodes. In order to remove all edges that are not significant, we present three methods for thresholding; 1) Top score, 2) prior known pairs, and 3) dynamic threshold based on prior knowledge.

1) Top score thresholding is implemented by ignoring all but the most significant outgoing edge for each node. This can be extended to include the k best scoring outgoing edges for each node.
2) Prior known pair thresholding is implemented by considering only pairs that are expected to interact. These prior known pairs should be included as input in the form of a binary matrix, where only pairs marked with a 1 will be included in the output.
3) Dynamic thresholding is achieved using prior known pairs as described in the section above. However in this case, the MSE of known pairs is used to create a dynamic threshold for determining significance of all pairs. Figure 2 shows the ratio between known pairs included and total pairs included, as the threshold is increased.

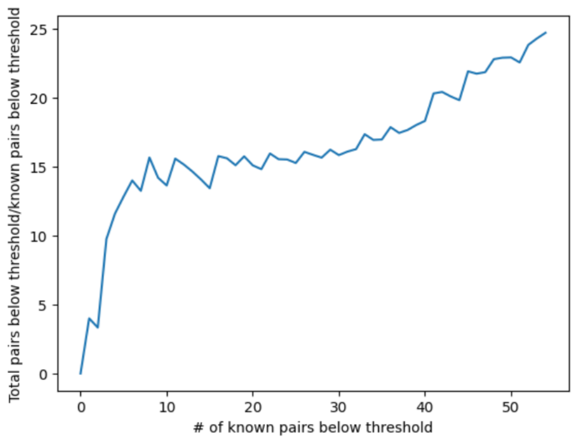

Fig 2: Ratio between known pairs and all pairs included during dynamic thresholding

In figure 2, a local minimum indicates a threshold that maximizes known pairs included while minimizing total pairs included.

Once thresholding is completed, what remains is a network containing all edges deemed significant, stored as a matrix. This matrix can be output as is, or converted to a binary adjacency matrix.

3 Results

In order to select the appropriate kernel, the average of all MSE values from a given run were considered. Runs using various combinations of dot product, radial basis function (RBF), and WhiteKernel kernels are compared using histograms in figure 3.

Most of note are the mean and median MSE values from building models using each kernel. These were compared to similar histograms built on only models of prior known pairs.

Although the sum of dot product and WhiteKernel kernels seems promising in figure 3, the increase in error when considering known pairs disqualifies it as the appropriate kernel, as it is mostly. In fact when looking at the difference between all pairs and known pairs, only two kernels show significant improvement; RBF * dot product + WhiteKernel, and RBF + dot product + WhiteKernel. The significant improvement in MSE from all pairs to known pairs indicates the models are picking up the relationships that exist in the known pairs effectively. Ultimately, RBF + dot product + WhiteKernel was selected as the most appropriate kernel for its slightly better performance and faster execution times.

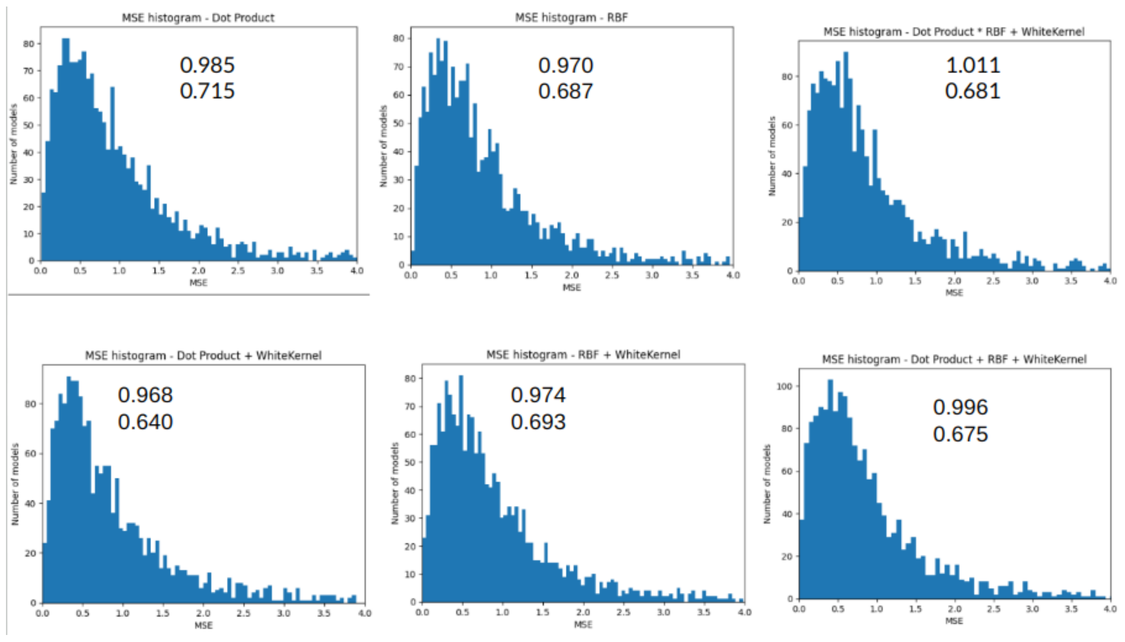

Fig 3: MSE distribution of all pairs

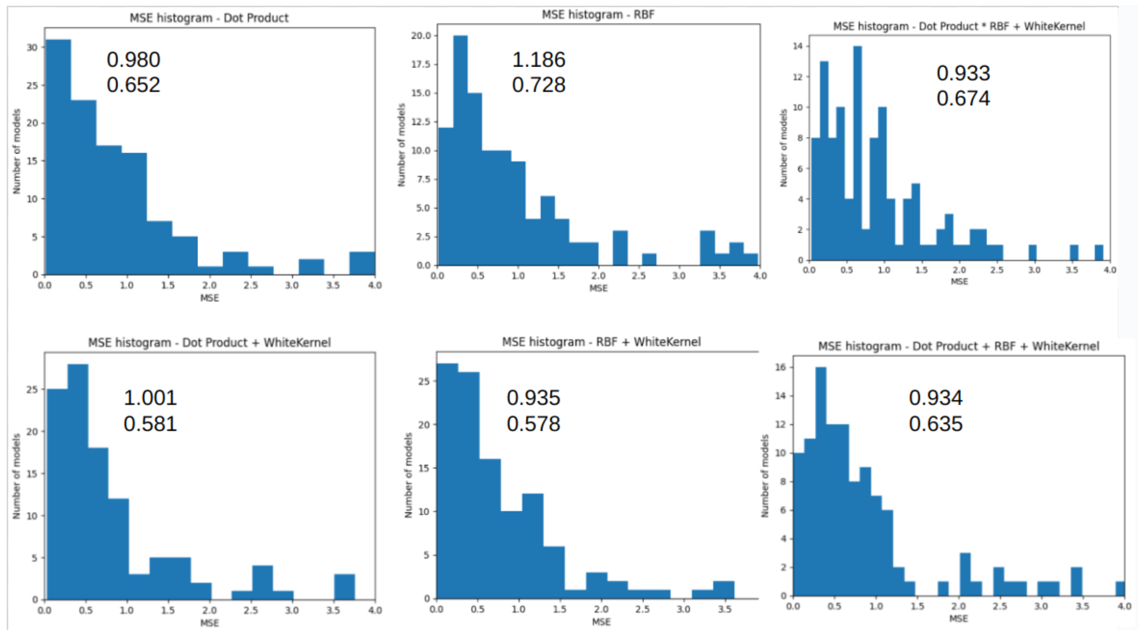

Fig 4: MSE distribution of known pairs

4 Discussion

Each thresholding method proposed has a unique set of advantages and disadvantages, and would likely be the best method for certain applications. The first method, top score thresholding, is useful for creating a minimum viable network. This network is the least populated possible while still containing at least one outgoing edge for each node. However, as shown in figure 5, it can also create a disconnected network.

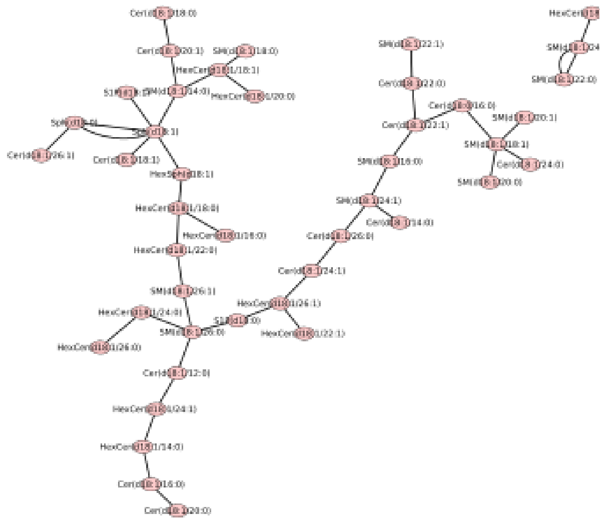

Fig 5: Network created with top score thresholding

Possible improvements to this method could include allowing a user to select a specific number of highest scoring edges to include from each node. In addition, when networks are disconnected, the highest scoring edges that connect two subgraphs could be added until the entire network is connected. It should also be explored whether bidirectional edges should be allowed - disallowing these could prevent a disconnected network, and in this case the node whose edge has been removed will have its second best scoring edge included instead. In addition, exploring adding the highest incoming edge for each node could yield different results.

The prior known pair thresholding is mainly useful for supplementing the findings from an already existing network. It can be used to synthesize edge weights and directionality. With any machine learning inferences, there is a risk that connections that do not exist or that are not significant will be misidentified. This approach avoids this pitfall by only including information on pairs that have been independently verified.

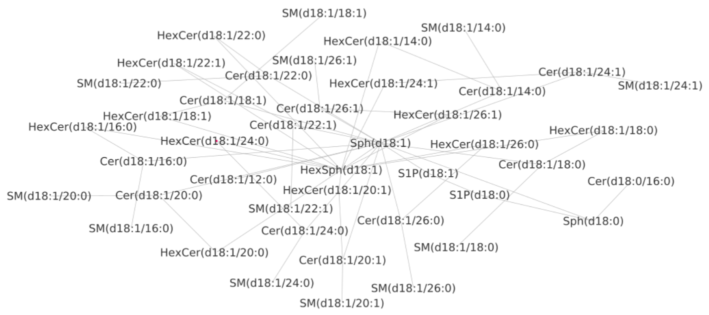

Fig 6: prior knowledge network

Dynamic thresholding is the approach in this paper that likely has the most potential for improvement. Using each MSE of a known pair as potential threshold, the ratio of known pairs below threshold to total pairs below threshold will show how effective a threshold is at minimizing the pairs included while maximizing the known pairs included. In figure 7 below, two possible points are identified by the red arrows as potential thresholds.

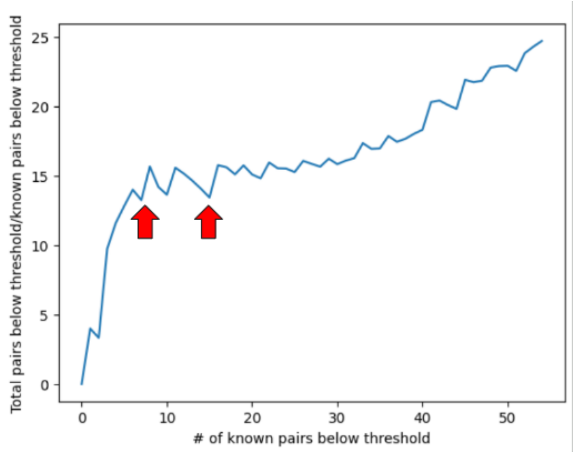

Fig 7: possible threshold values from dynamic thresholding

Using the leftmost identified threshold (the MSE of the 9th best performing known pair), the network in figure 8 was created. This network contains 141 total edges between 43 nodes.

This thresholding method may be improved by adding additional criteria for selection. For example, the average top score, or worst top score could be used as a minimum cutoff. The top score network could be included as part of the known pairs network to allow more options for threshold.

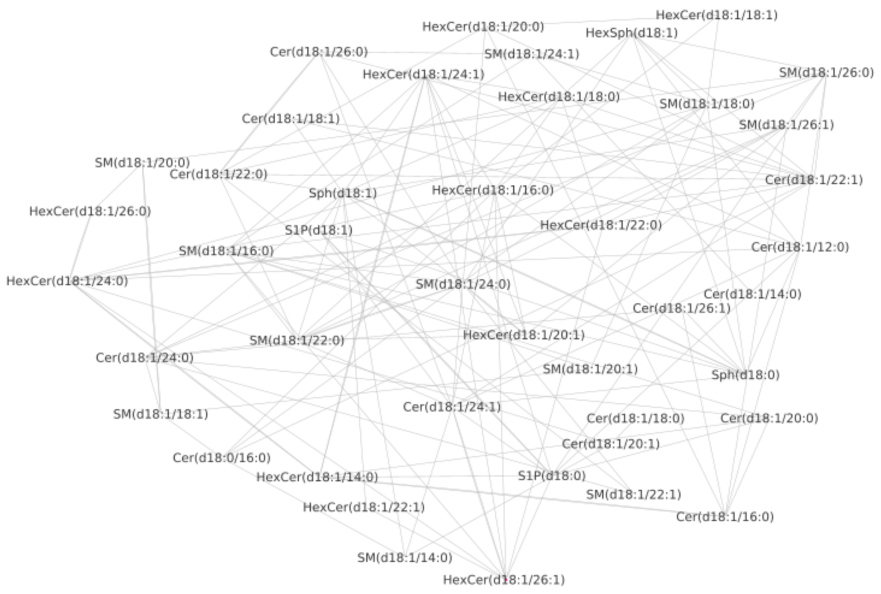

Fig 8: Network created with dynamic thresholding

5 Conclusion

GPR has been shown to be a reliable model for predicting the interactions between two variables. This can be applied to the derivation of biological networks from -omics data sheets. Kernel selection plays a significant role in performance of GPR output. Many options exist for thresholding the output from GPR, and should be explored in more detail. Users should consider their specific use case when selecting a thresholding method

References

Benedetti, E., Pučić-Baković, M., Keser, T. et al. A strategy to incorporate prior knowledge into correlation network cutoff selection. Nat Commun 11, 5153 (2020)

Cha, Gi-Wook & Moon, Hyeun & Kim, Young-Min & Hong, Won-Hwa & Hwang, Jung-Ha & Park, Won-Jun & Kim, Young-Chan. (2020). Development of a Prediction Model for Demolition Waste Generation Using a Random Forest Algorithm Based on Small DataSets. International journal of environmental research and public health. 17. 10.3390/ijerph17196997.

Ma'ayan, A. Introduction to network analysis in systems biology.
Sci Signal 2011;4(190):tr5.

Nguyen-Tran, *et al*. Network Development and Comparison in Lipidomics and Metabolomics, Metabolomics: Recent Advances and Future Applications with Springer Nature (2023)

Williams, C. K. I., Rasmussen, C. E. (2006). Gaussian Processes for Machine Learning. United Kingdom: MIT Press.