Gaussian Process Regression

Introduction, Comparison and Analysis

Tzu-Heng Lin, 2014011054, W42
Department of Electronic Engineering, Tsinghua University, Beijing, China
lzhbrian@gmail.com

ABSTRACT

[1] Gaussian Process Regression(GPR) is a powerful, non-parametric tool developed based on the Bayesian theory and the Statistical learning theory. Choosing the right *Mean Functions*, *Kernel Functions* as well as the *Likelihood Functions* and the *Inference Methods* have been critical to the performance of the model. However, these works are often hard and require much expertise & experience.

In this paper, we first give an introduction on the overall process of GPR. Subsequently, we give a precise explanation on some of the recent works which emphasize on the automatic construction of the *Kernel Function*. In addition, we implement sufficient number of experiments to systematically analyze the performance of GPR with different *Mean Functions*, *Likelihood Functions* and the *Inference Methods*. Our experiments are conducted on two interesting datasets.

We seek to provide an comprehensive practical overview in the field of Gaussian Process Regression.

1. INTRODUCTION

Machine learning has been a heated research topic these days. With this amazing tool, we are now capable of predicting the price of the stock price based on history, doing the classifying by just inputing the pixels of images.

Supervised learning is one of the most important sections for machine learning. And Regression is probably the core of supervised learning. By firstly inputing in the computer some of the training data, the machine would be able to learn the characteristics of the dataset and make predictions.

Gaussian Process Regression(GPR) is a supervised learning regression method which are getting increasingly welcome in both the research field and the industry. In a GPR, we take advantage of the flexibility and simplicity of a Gaussian Process and implement it into a regression problem.

There are still countless unsolved problems in the field of GPR. In this paper, we would comprehensively introduce the concept of GPR, including most of the notable works. Specifically, we focus on some marvellous kernel choosing works. We also implemented several experiments to quantitatively analyze the performance of different *Mean Functions*, *Likelihood Functions* and the *Inference Methods* We seek to provide a practical overview in the field of GPR.

The structure of this pape is as follows: In section 2, we provide a precise overview of a Gaussian Process Regression. In section 3, we briefly listed some of the notable progress on GPR. Section 4 describes a great recent work on the methods for auto-construction of the kernels. In section 5, we conduct two experiments and use them to compare the performance of different methods and algorithms. Conclusions are drawn in section 6.

[1]Tzu-Heng Lin is currently an undergraduate student in the Department of Electronic Engineering, Tsinghua University. His research interests include Big Data Mining, Machine Learning, etc. For more information about me, please see my personal website. Please feel free to contact me at any time via email

2. GAUSSIAN PROCESS REGRESSION

2.1 Regression

Regression Regression is probably one of the most fundamental problems in a wide range of fields including *Statistics*, *Signal Processing* and *Machine Learning*, etc. A regression problem is usually formulated as follows: Given a training set $D = \{(\mathbf{x}_i, y_i)|i = 1, 2, ..., n\}$, we assume that $\mathbf{x}_i, y_i$ have the following relationship:

$$y_i = f(\mathbf{x}_i) + e \tag{1}$$

where e is the error noise. By finding such $f(\cdot)$, we can predict what a corresponding y^* is in some test case $\mathbf{x}^*$. Note that $\mathbf{x}$ can either be a vector or a scalar.

Generalized Linear Model A widely used regression model is called Generalized Linear Model(GLM)[7], in which a regression function can be expressed as a linear combination:

$$f(\mathbf{x}) = \sum_{i=1}^{M} w_i \phi_i(\mathbf{x}) \tag{2}$$

where $\phi_i(x)$ is called the basis function. In a regular GLM analysis, we have to firstly determine what our basis functions we are going to use, and subsequently can we use the training dataset to derive the parameters in the basis functions and the coefficients in the regression function.

Mean Square Error We use a measurement called the Mean Square Error(MSE) to evaluate the performance of the regression function. It is defined as follows:

$$MSE = \frac{1}{m} \sum_{i=1}^{m} (f(\mathbf{x}^*) - y_i^*)^2 \tag{3}$$

where $f(x)$ represents the regression function. A smaller MSE represents a better regression function on a test set.

2.2 Gaussian Process Regression

Gaussian Process A Gaussian Process(GP) is any distribution over functions such that any finite set of function values $f(x_1), f(x_2), ..., f(x_N)$ have a joint Gaussian distribution[11]. It can usually be represented as

$$N\{E[f(x)], Cov[f(x), f(x^{'})]\} \tag{4}$$

where $E[f(x)]$ refers to its *Mean Function*, and $Cov[f(x), f(x^{'})]$ refers to its *Covariance Function*.

Gaussian Process Regression Gauss Process Regression(GPR)[11] is a popular regression method these years. The key of this method is to model the regression function $\{f(\mathbf{x})|\mathbf{x} \in S\}$ as a GP

$$f(x) \leftarrow N\{m(\mathbf{x}), K(\mathbf{x}, \mathbf{x}^{'})\} \tag{5}$$

where $m(\mathbf{x})$ is the *Mean function* and $K(\mathbf{x}, \mathbf{x}^{'})$ is the *Kernel Function*.

In a GPR, we don't have to derive the exact form of the regression function, we just need to determine the form of the above two functions. As introduced in GP, the *Kernel Function* $K(\mathbf{x}, \mathbf{x}^{'})$ is actually the covariance between $f(\mathbf{x})$ and $f(\mathbf{x}^{'})$, and if it is a zero-mean GP, the covariance turns into correlation. So a *Kernel Function* represents the relationship between $f(\mathbf{x})$ and $f(\mathbf{x}^{'})$.

By calculating the posterior probability of the desired $f(\mathbf{x}^*)$, *i.e.* $p(f(\mathbf{x}^*)|\mathbf{x}^*, D)$, we can derive the mean value along with the standard deviation of this estimation. On the other hand, we must noted that calculating the posterior probability will become an intractable work when we have a high-dimensional dataset. It is a need that we introduce some inference method to estimate this work.

To summarize, choosing a suitable *Mean Functions*, *Kernel Functions* as well as the *Likelihood Functions* and the *Inference Methods* is the key of a GPR model. Rasmussen's *Gaussian Processes for Machine Learning*[11] has implemented some marvellous Matlab/Octave code of GPR, it is available on his website[2] known as ***GPML***.

3. RELATED WORK

Kernel Function

Inference Methods Some popular works on inference methods include MCMC[5], Expectation Propagation[8], Variational Bayes[10, 9] and Laplace Approximation[13], etc. Due to the limited time and space, we won't go very deep into these inference methods.

4. KERNEL CHOICE

Choosing an appropriate *Kernel Function* in a regression problem has always been a nightmare for researchers and analysts. In this section, from the practical meaning of a *Kernel Function*, we introduce an automatic way to construct a desired composite form of *Kernel*. We also introduce a special *Additive Kernel* used specifically for an Additive Gaussian Process, which is a case we will be solving in our experiment (Section 5). This section is based on the work of Duvenaud and Lloyd et. al [3, 6, 2, 4].

[2]Available at http://www.gaussianprocess.org/gpml/code/

4.1 Kernels express structures

Base Kernels As we all know, different kernels can represent different kinds of data. In Figure 1, we show 4 different kinds of base kernels, each represents a specific kind of data characteristics (Table 1). Each function of the kernel is shown in Equation 6

$$\begin{cases} k_{LIN}(x, x^{'}) = \sigma_b^2 + \sigma_v^2(x - l)(x^{'} - l) \\ k_{SE}(x, x^{'}) = \sigma^2 exp(-\frac{(x - x^{'})^2}{2l^2}) \\ k_{PER}(x, x^{'}) = \sigma^2 exp(-\frac{2sin^2(\pi(x - x^{'})/p)}{l^2}) \\ k_{RQ}(x, x^{'}) = \sigma^2(1 + \frac{(x - x^{'})^2}{2\alpha l^2})^{-\alpha} \end{cases} \tag{6}$$

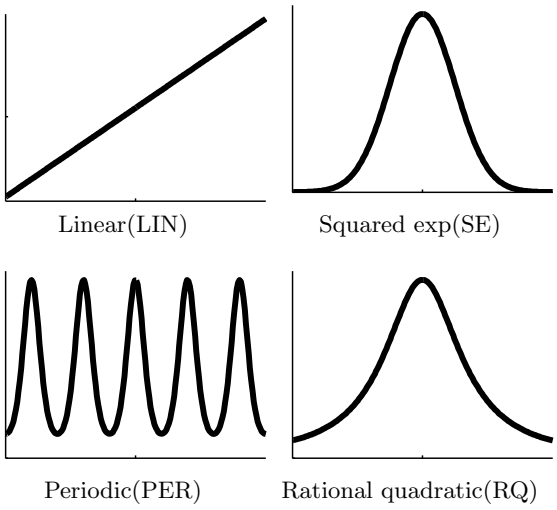

Figure 1: Base Kernels

Kernel	Data Structures
Linear(LIN)	linear functions
Squared exponential(SE)	local variation
Periodic(PER)	repeating structure
Rational quadratic(RQ)	multi-scale variation

Table 1: Different kinds of kernels and its represented data structures.

Compositional Kernels When the data structure we are dealing with is not contained in any of the above base kernel, we have to made one to fit the targeted data characteristics. A possible and probable approach of making a customized kernel is to do some addition and multiplication to the base kernels.

$$k_{k_a + k_b}(\mathbf{x}, \mathbf{x}^{'}) = k_a(\mathbf{x}, \mathbf{x}^{'}) + k_b(\mathbf{x}, \mathbf{x}^{'}) \tag{7}$$

$$k_{k_a \times k_b}(\mathbf{x}, \mathbf{x}^{'}) = k_a(\mathbf{x}, \mathbf{x}^{'}) \times k_b(\mathbf{x}, \mathbf{x}^{'}) \tag{8}$$

By addition and multiplication, we are able to construct some very complicated data structure. Below, we show some examples of structures by compositional kernels (Figure 2) and the data structures they are capable of representing (Table 2).

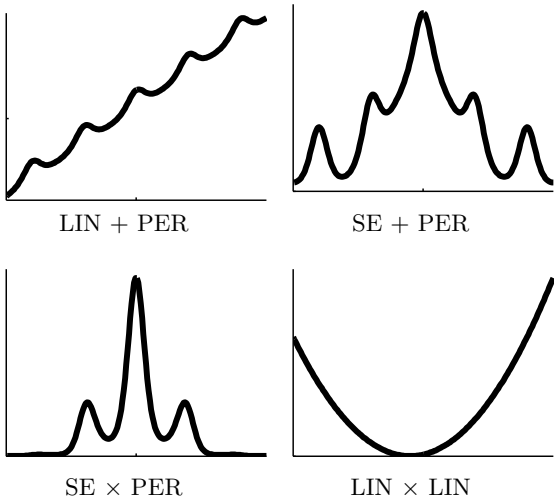

Figure 2: Some examples of the compositional kernels

Kernel	Data Structures
LIN+ PER	periodic plus trend
SE + PER	periodic plus noise
SE × PER	locally periodic
LIN × LIN	quadratic functions

Table 2: Some examples of the compositional kernels

4.2 Automatic Construction

As we can see in the above analysis, choice of the form of a *Kernel Function* can be critical in a GPR. However, this process was used to be a privileged work for experts since it requires a whole lot of experience and expertise. In [3, 6, 2], Duvenaud and Lloyd et.al describe a tractable method to let a computer be capable of doing this work. This procedure is summarized as follows:

Kernel Family We consider the *base kernels* as shown in Figure 1, any algebraic expression using these kernels as a combination of + and × could be our target. And any of our target with the concatenation of the parameters forms a kernel family.

Scoring a Kernel Family We must find a way to evaluate a kernel family. At here, we use the marginal likelihood[1] as our criterion, and use the Bayesian Information Criterion(BIC)[12] to approximate the integration over kernel parameters.

Search over structures Last, by using a greedy search: At each stage, expand the current kernel by all possible *operators* and choose the highest scoring one. The number of working stage is defined by user. Possible operators are listed as follows:

1. Any expression S can be replaced by $S + B$
2. Any expression S can be replaced by $S \times B$
3. Any base kernel B can be replaced by $B^{'}$

where B and $B^{'}$ represent any base kernel family. Note that, in the work [6], Lloyd et. al also take account of the changepoint operator CP, the changewindow operator CW, and some empirical operators. They also add some base kernels to improve the algorithm. Due to limited time and space, we only introduce the most important part here.

The implemented Matlab and Python code by Duvenaud and Lloyd et.al can be found on github[3].

Algorithm 1 Automatic kernel construction algorithm

Require: Required search depth N
  Initialize kernel S
  for $t = 1 \rightarrow N$ **do**
    $CurScore \leftarrow 0$
    for *Possible operators* $S^{'}$ **do**
      if Score(S) > CurScore **then**
        $S \leftarrow S^{'}$
      end if
    end for
  end for
  return *kernel* S

4.3 Additive Gaussian Processes

Additive Gaussian Process [4] In some

Additive Kernels We now give the precise definition of additive kernels. As shown in [4], we define the 1st, 2nd, nth order additive kernel as:

$$k_{add_1}(\mathbf{x}, \mathbf{x}^{'}) = \sigma_1^2 \sum_{i=1}^{D} k_i(\mathbf{x}_i, \mathbf{x}_i^{'}) \tag{9}$$

$$k_{add_2}(\mathbf{x}, \mathbf{x}^{'}) = \sigma_2^2 \sum_{i=1}^{D} \sum_{j=i+1}^{D} k_i(\mathbf{x}_i, \mathbf{x}_i^{'}) k_j(\mathbf{x}_j, \mathbf{x}_j^{'}) \tag{10}$$

$$k_{add_n}(\mathbf{x}, \mathbf{x}^{'}) = \sigma_n^2 \sum_{1 \leqslant i_1 < i_2 < \ldots < i_n \leqslant D} (\prod_{d=1}^{n} k_{i_d}(\mathbf{x}_{i_d}, \mathbf{x}_{i_d}^{'})) \tag{11}$$

$$\tag{12}$$

where $k_i(\mathbf{x}_i, \mathbf{x}_i^{'})$ is the *base kernel* we assigned at first for each dimension $i \in \{1, 2, ..., D\}$. A full additive kernel is a sum of all orders' additive kernels, as in equation 15:

$$K_{add_{full}}(\mathbf{x}, \mathbf{x}^{'}) = \sum_{n=1}^{D} k_{add_n}(\mathbf{x}, \mathbf{x}^{'}) \tag{13}$$

In the real practice, we could choose specific orders of the additive kernels:

$$K_{add_{prac}}(\mathbf{x}, \mathbf{x}^{'}) = \sum_{n} k_{add_n}(\mathbf{x}, \mathbf{x}^{'}) \tag{14}$$

We also noted that if every base kernel is a one-dimensional squared-exponential(SE) kernal, the Dth order term of the additive kernel would be:

$$\begin{aligned} K_{add_D}(\mathbf{x}, \mathbf{x}^{'}) &= \sigma_D^2 \prod_{d=1}^{D} k_d(\mathbf{x}_d, \mathbf{x}_d^{'}) \\ &= \sigma_D^2 \prod_{d=1}^{D} exp(-\frac{(\mathbf{x}_d - \mathbf{x}_d^{'})^2}{2l_d^2}) \\ &= \sigma_D^2 exp(-\sum_{d=1}^{D} \frac{(\mathbf{x}_d - \mathbf{x}_d^{'})^2}{2l_d^2}) \end{aligned} \tag{15}$$

which is just the multivariate squared-exponential kernel.

[3]Available at github.com/jamesrobertlloyd/gpss-research and github.com/jamesrobertlloyd/gp-structure-search.

5. EXPERIMENTS

5.1 First Experiment

Our first experiment is conducted on a one-dimensional data as shown in Figure 4. In this experiment, we have a 543-point training set ranging from $x = -32$ to $x = 14$, and we want to predict some 67-point test set ranging from $x = 14$ to $x = 21$.

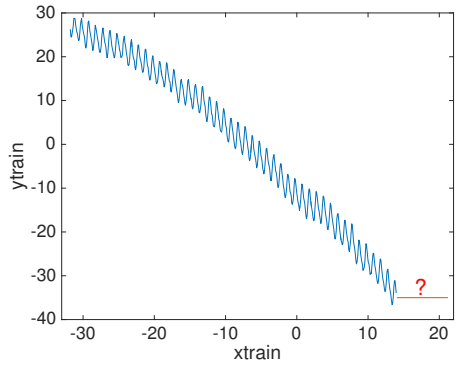

Figure 3: A one-dimensional dataset

5.2 Second Experiment

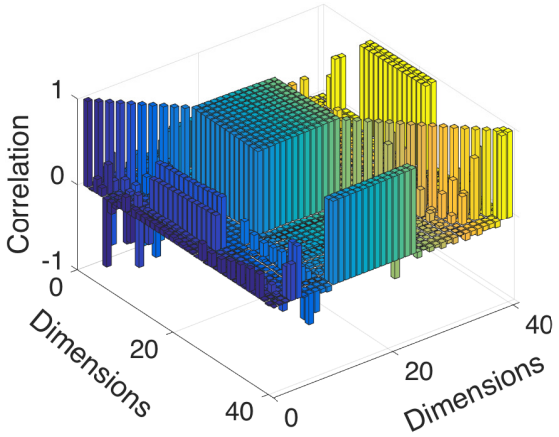

Figure 4: A 40-dimensional dataset

6. CONCLUSION

In this paper, we discuss about the Gaussian Process Regression.

We systematically compare three methods of partition function estimation which are crucial works in training a Restricted Bolztmann Machine or a Deep Belief Network.

As future work, we would like to join more methods to the comparison and if could, propose some improvement to the algorithms available.

7. ACKNOWLEDGEMENT

I would like to thank Yuanxin Zhang, XueChao Wang, for the discussion with me on the algorithms. Without them, I wouldn't have the possibility to accomplish this work in such a short time. This paper is a project of Stochastic Process Course in Tsinghua University, taught by Prof. Zhijian Ou.

Source Code and Dataset Source Code to perform all experiments, along with the dataset in this paper can be found in my github repository[4].

[4]Available at http://github.com/lzhbrian/gpr

8. REFERENCES

[1] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Occam's razor. *Information processing letters*, 24(6):377–380, 1987.

[2] D. Duvenaud. *Automatic model construction with Gaussian processes*. PhD thesis, University of Cambridge, 2014.

[3] D. K. Duvenaud, J. R. Lloyd, R. B. Grosse, J. B. Tenenbaum, and Z. Ghahramani. Structure discovery in nonparametric regression through compositional kernel search. In *ICML (3)*, pages 1166–1174, 2013.

[4] D. K. Duvenaud, H. Nickisch, and C. E. Rasmussen. Additive gaussian processes. In *Advances in neural information processing systems*, pages 226–234, 2011.

[5] D. Gamerman. Sampling from the posterior distribution in generalized linear mixed models. *Statistics and Computing*, 7(1):57–68, 1997.

[6] J. R. Lloyd, D. Duvenaud, R. Grosse, J. B. Tenenbaum, and Z. Ghahramani. Automatic construction and natural-language description of nonparametric regression models. *arXiv preprint arXiv:1402.4304*, 2014.

[7] P. McCullagh. Generalized linear models. *European Journal of Operational Research*, 16(3):285–292, 1984.

[8] T. P. Minka. Expectation propagation for approximate bayesian inference. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, pages 362–369. Morgan Kaufmann Publishers Inc., 2001.

[9] H. Nickisch and M. W. Seeger. Convex variational bayesian inference for large scale generalized linear models. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 761–768. ACM, 2009.

[10] J. Palmer, D. Wipf, K. Kreutz-Delgado, and B. Rao. Variational em algorithms for non-gaussian latent variable models. *Advances in neural information processing systems*, 18:1059, 2006.

[11] C. E. Rasmussen. Gaussian processes for machine learning. 2006.

[12] G. Schwarz et al. Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464, 1978.

[13] L. Tierney and J. B. Kadane. Accurate approximations for posterior moments and marginal densities. *Journal of the american statistical association*, 81(393):82–86, 1986.