A Precise Hard-Cut EM Algorithm for Mixtures of Gaussian Processes

Ziyi Chen[1], Jinwen Ma[1,*], and Yatong Zhou[1,2]

[1] Department of Information Science, School of Mathematical Sciences, Peking University, Beijing, 100871, China
jwma@math.pku.edu.cn
[2] School of Information Engineering, Hebei University of Technology, Tianjin, 300401, China

Abstract. The mixture of Gaussian processes (MGP) is a powerful framework for machine learning. However, its parameter learning or estimation is still a very challenging problem. In this paper, a precise hard-cut EM algorithm is proposed for learning the parameters of the MGP without any approximation in the derivation. It is demonstrated by the experimental results that our proposed hard-cut EM algorithm for MGP is feasible and even outperforms two available hard-cut EM algorithms.

Keywords: Mixture of Gaussian process, Parameter learning, EM algorithm.

1 Introduction

Gaussian process (GP) is a powerful learning model for both regression and classification. Nevertheless, it cannot describe multimodality dataset and needs a large number of computations. To tackle these issues, Tresp [2] proposed the Mixture of GPs (MGP) in 2000, which was directly derived from the mixture of experts.

From then on, various versions of MGP models have been suggested. Most of them could be classified into the conditional model 'x→z→y' [2-6] and the generative model 'z→x→y' [1],[7-10] where x, y and z denote the input, output and the latent component indicator, respectively. In comparison with the first model, the second one has two advantages: (1). The missing features can be easily inferred from the outputs; (2). The influence of inputs on the outputs is more clear [8].

For the parameter learning or estimation of MGP, there are three kinds of learning algorithms: MCMC, variational Bayesian inference, and EM algorithm. As for MCMC approaches, Gibbs samples of the indicators, parameters or hyper-parameters were obtained in turn from their posteriors [8,9]. However, Nguyen & Bonilla [3] pointed out that the time complexity of the MCMC method is very high. As for the variational Bayesian algorithms, the main strategy is to approximate the posterior of parameters by a factorized and simplified form [5,6], but such an approximation may lead to a rather deviation from the true objective function.

* Corresponding author.

D.-S. Huang et al. (Eds.): ICIC 2014, LNAI 8589, pp. 68–75, 2014.

In general, EM algorithm is a popular and efficient choice for the parameter learning of mixture models. However, the posteriors of latent variables and Q function in the cases of MGP are rather complicated as the outputs are dependent. In order to alleviate this difficulty, some EM algorithms with the help of certain approximation mechanisms have already been proposed successively.

Tresp [2] firstly proposed the EM algorithm for MGP, in which the M-step integrated the posterior probability of each sample belonging to a GP component into the learning of each component. Along this direction, Stachniss et al. [4] developed a similar EM algorithm for the sparse MGP. However, the learning in the M-step was heuristic in [2] and [4] without maximizing Q function and the time complexity in [2] was still high as that in [1]. On the other hand, Yuan & Neubauer [1], Nguyen & Bonilla [3], Miguel et al. [5] and Sun & Xu [10] proposed some variational EM algorithms for MGP in which the posterior probabilities were approximated with certain factorized forms. Recently, Yang & Ma [7] also proposed the EM algorithm for MGP with the help of leave-one-out cross-validation (LOOCV) probability decomposition of the total likelihood.

Although the approximations or simplifications have been made for the learning in the M-step, these soft EM algorithms for MGP are still time-consuming. In order to reduce the time complexity, Nguyen & Bonilla [3] recently proposed a variational hard-cut EM algorithm for MGP under certain sparseness constraints, which actually partitions all the samples into these components according to the MAP criterion in the E-step. In fact, this hard-cut EM algorithm was more efficient than the soft EM algorithm since we could get the hyper-parameters of each GP independently in the M-step rather than maximize a complicated Q function. Moreover, in the same way, the soft EM algorithm for MGP with the LOOCV probability decomposition [7] can be easily turned into a hard-cut version of the EM algorithm for MGP, which is here referred to as the LOOCV hard-cut EM algorithm for convenience.

In this paper, we follow the generative MGP model and propose a precise hard-cut EM algorithm for MGP without any approximation used for the likelihood function or Q function. Actually, we further refine the MGP model to exclude extra priors from the main chain 'z→x→y' and according to this refined model, the hard-cut EM algorithm becomes more accurate than some popular algorithms since the posterior probability of each sample belonging to each component used in the algorithm is strictly derived, and the heuristic approximations used in [3] and [7] are avoided. It is demonstrated by the experimental results that our proposed hard-cut EM algorithm is feasible and even outperforms the two available hard-cut EM algorithms.

2 MGP Model

2.1 GP Model

For regression and prediction task, a GP is mathematically defined by

$$y = [y_1 \quad y_2 \quad \cdots \quad y_N]^T \sim N[m(X), K(X,X) + \sigma^2 I], \tag{1}$$

where $\{(x_t, y_t)\}_{t=1}^{N}$ denotes the set of given sample points or dataset, σ^2 is the intensity of noise, I is an $N \times N$ identity matrix, $m(X) = [m(x_1) \quad m(x_2) \quad \cdots \quad m(x_N)]^T$ and $K(X,X) = [K(x_i, x_j)]_{N \times N}$ are the means and kernel matrix, respectively. For simplicity, we set $m(X) = 0$. The most commonly used kernel is the SE kernel [11], being given by $K(x_i, x_j) = l^2 \exp(-0.5 f^2 \left\| x_i - x_j \right\|^2)$.

In order to learn the hyper-parameters $\theta = \{l, f, \sigma\}$, we use the commonly adopted approach------maximize likelihood estimation (MLE).

After the parameter learning process, the prediction of the output at a test input x^* is

$$\hat{y}^* = K(x^*, X)[K(X,X) + \sigma^2 I]^{-1} y \tag{2}$$

where $y = [y_1, y_2, \cdots, y_N]^T$ is the vector of training outputs, $K(X,X) = [K(x_i, x_j)]_{N \times N}$, and $K(x^*, X) = [K(x^*, x_1), K(x^*, x_2), \cdots, K(x^*, x_N)]$ denotes the kernel relationship vector of the training inputs to the test input.

2.2 Generative MGP Model

We adopt a full generative model ('z→x→y') due to its resistance to missing features and clear relationship between inputs and outputs [8].

At first, the latent indicators $\{z_t\}_{t=1}^{N}$ are generated by the Multinomial distribution:

$$\Pr(z_t = c) = \pi_c; c = 1 \sim C \ i.i.d \ for \ t = 1 \sim N \tag{3}$$

Given indicators, each input fulfills a Gaussian distribution:

$$p(x_t \mid z_t = c) \sim N(\mu_c, S_c); c = 1 \sim C \ \ i.i.d \ \ for \ \ t = 1 \sim N \tag{4}$$

After specifying $\{z_t, x_t\}_{t=1}^{N}$, the outputs of each component fulfill a GP, that is

$$p\left(\begin{bmatrix} y_{c,1} & y_{c,2} & \cdots & y_{c,N(c)} \end{bmatrix} \middle| \begin{bmatrix} x_{c,1} & x_{c,2} & \cdots & x_{c,N(c)} \end{bmatrix}\right) \sim N\left(\vec{0}, \left[K(x_{c,i}, x_{c,j} \mid \theta_c)\right]_{N(c) \times N(c)} + \sigma_c^2 I_{N(c)}\right) \ i.i.d. for c = 1 \sim C \tag{5}$$

where for the c-th component, $\{x_{c,i}, y_{c,i}\}_{i=1}^{N(c)}$ are the samples, $\theta_c = \{l_c, f_c, \sigma_c\}$ are the GP hyper-parameters and $K(x_{c,i}, x_{c,j} \mid \theta_c) = l_c^2 \exp(-0.5 f_c^2 \left\| x_i - x_j \right\|^2)$ is the SE kernel function.

The whole generative model can be completely defined by Eqs (3), (4) & (5). In fact, after the allocation of samples to these components, each GP can be learnt independently, as suggested in (4) and (5).

3 Precise Hard-Cut EM Algorithm

For this generative MGP model, we here adopt the EM algorithm framework to learn the whole parameters $\{\pi_c, \mu_c, S_c, l_c, f_c, \sigma_c\}_{c=1}^{C}$, taking $\{z_t\}_{t=1}^{N}$ as the latent variables.

Firstly, we derive the posterior probabilities of these indicators, i.e., z_t. Based on Eqs (3), (4) & (5), we get the following likelihood function:

$$p(z_t = c, x_t, y_t) = \pi_c N(x_t \mid \mu_c, S_c) N\left(y_t \middle| 0, l_c^2 + \sigma_c^2\right) \tag{6}$$

According to the Bayesian formula, it can be derived that

$$p(z_t = c \mid x_t, y_t) = \pi_c N(x_t \mid \mu_c, S_c) N\left(y_t \middle| 0, l_c^2 + \sigma_c^2\right) / \left\{\sum_{c'=1}^{C} \pi_{c'} N(x_t \mid \mu_{c'}, S_{c'}) N\left(y_t \middle| 0, l_{c'}^2 + \sigma_{c'}^2\right)\right\} \tag{7}$$

As the computational complexity of Q function is $O(C^N)$, it is more reasonable to construct a hard-cut version of the EM algorithm to reduce the computational burden. According to this idea, we propose a hard-cut EM algorithm as follows.

- Step 1 Initialization of indicators: cluster $\{(x_t, y_t)\}_{t=1}^{N}$ into C classes by the k-means clustering, and set $z_t \leftarrow$ *The indicator of the t – th sample to the cluster*
- Step 2 M-step: calculate π_c, μ_c and S_c in the way of the Gaussian mixture model:

$$\pi_c \leftarrow \frac{1}{N}\sum_{t=1}^{N} I(z_t = c), \quad \mu_c \leftarrow \frac{\sum_{t=1}^{N} I(z_t = c) x_t}{\sum_{t=1}^{N} I(z_t = c)}, \quad S_c \leftarrow \frac{\sum_{t=1}^{N} I(z_t = c)(x_t - \mu_c)(x_t - \mu_c)^T}{\sum_{t=1}^{N} I(z_t = c)} \tag{8}$$

  and obtain the GP parameters $\{l_c, f_c, \sigma_c\}_{c=1}^{C}$ by maximizing the likelihood (5).

- Step 3 E-step: classify each sample into the corresponding component according to the MAP criterion:

$$z_t \leftarrow \arg\max_{1 \le c \le C} p(z_t = c \mid x_t, y_t) = \arg\max_{1 \le c \le C} \pi_c N(x_t \mid \mu_c, S_c) N(y_t \mid 0, l_c^2 + \sigma_c^2) \tag{9}$$

- Step 4 If the indicators do not change any more, stop and output the parameters of MGP. Otherwise, return to Step 2.

After the convergence of the hard-cut EM algorithm, we have obtained the estimates of all the parameters of the MGP. For a test input x^*, we can classify it into the z -th component of MGP by the MAP criterion as follows:

$$z = \arg\max_{1 \le c \le C} p(z^* = c \mid x^*) = \arg\max_{1 \le c \le C} \pi_c N(x^* \mid \mu_c, S_c) \tag{10}$$

Based on such a classification, we can predict the output of the test input via the corresponding GP using (2).

4 Experimental Results

4.1 On a Typical Synthetic Dataset of MGP

In order to test the validity and feasibility of our proposed hard-cut EM algorithm, we begin to generate a typical synthetic dataset of MGP with 4 GP components. Actually, for each GP, we typically generate 500 training samples as well as 100 test samples. For the evaluation of the algorithm, we will compute the absolute error between the learned and true parameters as well as the root mean squared error (RMSE) for the output prediction. The true parameters of the four components are given in Table 1.

We implement our proposed hard-cut EM algorithm on this synthetic dataset. It is found by the experiments that the classification error rates on the training and test datasets are 0.30% and 0.50%, respectively. The running time for both the learning and prediction tasks is just 81.8680s with an Intel(R) Core(TM) i5 CPU and 4.00GB of RAM using Matlab R2013a, which is acceptable since we have 2000 training samples and 400 test samples in total. The true and estimated values of the parameters as well as the absolute error between them are listed in Table 1. From Table 1, we can observe the absolute errors are generally very small, and it can be found from Fig.1 that the predictive curve fits the test points very well. Moreover, the RMSE of the output prediction is only 0.4901. In sum, our proposed algorithm for MGP is demonstrated valid and feasible on the synthetic datasets.

Table 1. The true value (TV), estimated value (EV) and absolute error (AE) of the parameters for each GP component (C) on the typical synthetic dataset of MGP

C	Value	π_c	μ_c	S_c	l_c^2	σ_c^2	f_c^2
1	TV	0.2500	0.0000	0.1000	0.9000	0.1000	2.0000
	EV	0.2500	-0.0069	0.1006	0.7978	0.1041	0.3461
	AE	0.0000	0.0069	0.0006	0.2022	0.0041	1.6539
2	TV	0.25	3.0000	0.2000	1.0000	0.2000	3.0000
	EV	0.2495	3.0151	0.1916	1.3512	0.2007	3.1279
	AE	0.0005	0.0151	0.0084	0.3512	0.0007	0.1279
3	TV	0.2500	6.0000	0.3000	1.1000	0.3000	4.0000
	EV	0.2500	6.0219	0.2982	2.7321	0.2966	2.6416
	AE	0.0000	0.0219	0.0018	1.6321	0.0034	1.3584
4	TV	0.2500	9.0000	0.4000	1.2000	0.4000	5.0000
	EV	0.2505	9.0060	0.4085	0.5385	0.3790	7.7323
	AE	0.0005	0.0060	0.0085	0.6615	0.0210	2.7323

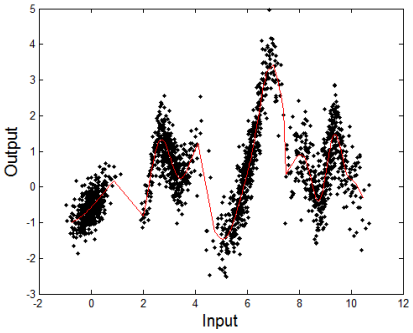

Fig. 1. The predictive curve (red solid line) and test sample points (black dots) of our proposed hard-cut EM algorithm on the typical synthetic dataset

4.2 Comparison with the LOOCV and Variational Hard-Cut EM Algorithms

We further compare our proposed hard-cut EM algorithm with the LOOCV and variational hard-cut EM algorithms on a toy dataset and a motorcycle dataset used in [7] and [8]. Actually, the toy dataset consists of four groups which are generated from 4 continuous functions. For each group, there are 50 training samples and 50 test samples. For the purpose of prediction, we certainly use the MGP with 4 components.

Motorcycle dataset is another popular one for the evaluation of the MGP methods [3],[7]. It consists of observations of accelerometer readings at 133 different times, belonging to three actual classes. Here, we use the 7-fold cross-validation, with the k-th fold being composed of $\{(x_t,y_t) : t=7s+k, s=0,1,\ldots,18\}$, where the inputs x_t are sorted in an ascending order. In this case, we use the MGP with 3 components.

We implement each of the three hard-cut EM algorithms five times on these two datasets under the same computational environment as above. The average prediction RMSEs and running times of the three algorithms on toy and motorcycle dataset are listed in Tables 2, respectively, while the corresponding predictive curves are also plotted in Figs. 2 & 3.

From Table 2, it can be found that our proposed hard-cut EM algorithm converges more accurately than the LOOCV and variational hard-cut EM algorithms. The reason may be that our proposed algorithm is more precise since it is strictly derived without

Table 2. The average prediction RMSEs and running times of the three hard-cut EM algorithms on toy and motorcycle dataset

The hard-cut EM algorithms	Toy dataset		Motorcycle dataset	
	Average RMSE	Average Time (s)	Average RMSE	Average Time (s)
Our proposed EM algorithm	16.5199	8.4513	19.1109	2.0401
The LOOCV EM algorithm	24.9874	43.4974	28.9551	10.5501
The variational EM algorithm	20.4238	57.3100	26.7883	62.0234

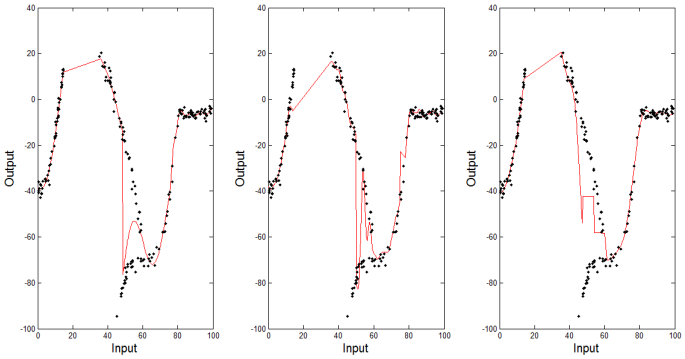

Fig. 2. The predictive curves (red solid line) and test points (black dots) of our proposed hard-cut EM algorithm (left), the LOOCV hard-cut EM algorithm (middle) and the variational hard-cut EM algorithm (right) on toy dataset

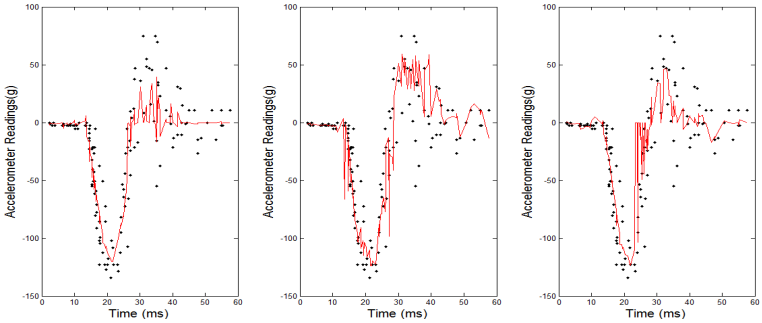

Fig. 3. The predictive curves (red solid line) and test points (black dots) of our proposed hard-cut EM algorithm (left), the LOOCV hard-cut EM algorithm (middle) and the variational hard-cut EM algorithm (right) on motorcycle dataset with 7 fold CV

any approximations like those used in the LOOCV decomposition and variational inference. Besides, our algorithm consumes much less time than the two EM algorithms due to its easy computation of the posterior probabilities. It can be also observed from Figs 2 & 3 that the predictive curves of our algorithm fit at least as well as the variational hard-cut EM algorithm, while these two hard-cut EM algorithms are smoother and fit better than those of the LOOCV hard-cut EM algorithm. On the whole, our proposed hard-cut EM algorithm clearly outperforms the LOOCV and variational hard-cut EM algorithm for prediction on both toy and motorcycle datasets.

5 Conclusion

We have investigated the learning problem of mixture of Gaussian processes (MGP) and proposed a precise hard-cut EM algorithm for it. In order to get this algorithm, the generative MGP model is redefined and the posterior probabilities of the latent indicators are strictly derived. In the algorithm design, the samples are partitioned in a

hard-cut way according to the MAP criterion on their posterior probabilities obtained in E-step, while each GP component is learned independently in M-step. It is demonstrated by the experimental results that our proposed hard-cut EM algorithm is effective and efficient, and even outperforms the LOOCV and variational hard-cut EM algorithms.

Acknowledgements. This work was supported by the Natural Science Foundation of China for Grant 61171138. The authors would like to thank Dr. Yang Yan for her valuable discussions on the analysis and comparison of the LOOCV hard-cut EM algorithm for MGP.

References

1. Yuan, C., Neubauer, C.: Variational mixture of Gaussian process experts. In: Advances in Neural Information Processing Systems, vol. 21, pp. 1897–1904 (2009)
2. Tresp, V.: Mixtures of Gaussian processes. In: Advances in Neural Information Processing Systems, vol. 13, pp. 654–660 (2000)
3. Nguyen, T., Bonilla, E.: Fast Allocation of Gaussian Process Experts. In: Proceedings of The 31st International Conference on Machine Learning, pp. 145–153 (2014)
4. Stachniss, C., Plagemann, C., Lilienthal, A.J., et al.: Gas Distribution Modeling using Sparse Gaussian Process Mixture Models. In: Proc. of Robotics: Science and Systems, pp. 310–317 (2008)
5. Lázaro-Gredilla, M., Van Vaerenbergh, S., Lawrence, N.D.: Overlapping Mixtures of Gaussian Processes for the data association problem. Pattern Recognition 45, 1386–1395 (2012)
6. Ross, J., Dy, J.: Nonparametric Mixture of Gaussian Processes with Constraints. In: Proceedings of the 30th International Conference on Machine Learning, pp. 1346–1354 (2013)
7. Yang, Y., Ma, J.: An efficient EM approach to parameter learning of the mixture of Gaussian processes. In: Liu, D., Zhang, H., Polycarpou, M., Alippi, C., He, H. (eds.) ISNN 2011, Part II. LNCS, vol. 6676, pp. 165–174. Springer, Heidelberg (2011)
8. Meeds, E., Osindero, S.: An alternative infinite mixture of Gaussian process experts. In: Advances in Neural Information Processing Systems, vol. 18, pp. 883–890 (2006)
9. Sun, S.: Infinite mixtures of multivariate Gaussian processes. In: Proceedings of the International Conference on Machine Learning and Cybernetics, pp. 1–6 (2013)
10. Sun, S., Xu, X.: Variational inference for infinite mixtures of Gaussian processes with applications to traffic flow prediction. IEEE Trans. on Intelligent Transportation Systems 12(2), 466–475 (2011)
11. Rasmussen, C.E., Williams, C.K.I.: Gaussian processes for machine learning. In: Adaptive Computation and Machine Learning. The MIT Press, Cambridge (2006)