Sequential Training of Semi-Supervised Classification Based on Sparse Gaussian Process Regression

Rongqing Huang and Shiliang Sun

Department of Computer Science and Technology, East China Normal University
500 Dongchuan Road, Shanghai 200241, P. R. China
E-MAIL: rqhuang09@gmail.com, slsun@cs.ecnu.edu.cn

Abstract:

Gaussian process regression (GPR) is a very important Bayesian approach in machine learning applications. It has been extensively used in semi-supervised learning tasks. In this paper, we propose a sequential training method for solving semi-supervised binary classification problem. It assigns targets to test inputs sequentially making use of sparse Gaussian process regression models. The proposed approach deals with only one part of the whole data set at a time. Firstly, the IVM produces a sparse approximation to a Gaussian process (GP) by combining assumed density filtering (ADF) with a heuristic for choosing points based on minimizing posterior entropy, and then a sparse GPR classifier is learnt on part of the whole data set. Secondly, the representative points selected in the first step including part of remainder examples are used to train another sparse GPR classifier. Repeat the two steps sequentially until all unlabeled examples are deal with. The proposed approach is simple and easy to implement. The hyperparameters are estimated easily by maximizing the marginal likelihood without resorting to expensive cross-validation technique. The evaluations of the proposed method on several real world data sets reveal promising results.

Keywords:

Gaussian process (GP); information vector machine (IVM); sparse Gaussian process regression; sequential training; assumed density filtering (ADF).

1. Introduction

In real world problems, semi-supervised learning usually involves lots of unlabeled examples. As to many existing Gaussian process-based semi-supervised learning algorithms [1], the computation cost of training a GPR model is three power of the number of examples, which is mainly due to the requirement to invert a matrix. The high computational complexity makes the standard GP-based semi-supervised algorithms difficult to be directly applied to large-scale problems. On the other hand, if a semi-supervised learning algorithm lacks the ability to deal with large-scale problems, it is hard to demonstrate it's effectiveness in practical applications. To overcome the drawbacks of many existing semi-supervised learning algorithms of high computational complexity and difficult application in large-scale data sets, we propose the sequential training method of semi-supervised classification. Although the idea of sequential training is not new [2], we are not aware of its any application to Gaussian process based semi-supervised learning so far. The proposed method proceeds as follows. Firstly, we use the IVM technique to train a sparse GPR classifier [3] on part of labeled and unlabeled examples, the outputs are the targets of unlabeled examples and the representative points. Secondly, we go on using the representative points selected in the first step and some new examples to train another sparse semi-supervised GPR classifier. We repeat the above two steps sequentially until all unlabeled examples are assigned targets. The sequential training method of semi-supervised learning applicable to large-scale data sets is obtained in the end. Besides the advantage of dealing with large-scale data sets, the proposed method is also appropriate to the case of online learning that the training examples are increasing.

Lawrence et al extended the standard IVM technique combining with a null category noise model (NCNM) to build a sparse GP classifier for semi-supervised classification. They introduced a virtual category 0 and assumed that the decision boundary avoids dense unlabeled data region. However, the marginal likelihood associated with the noise model is not log-concave. Therefore, the Gaussian approximation to the noise model can have negative variance [4].

Patel et al firstly extended the iterative support vector regression of cluster [5] to semi-supervised learning and then proposed the iterative Gaussian process regression algorithm. Finally, they proposed the sparse Gaussian process regression algorithm by combining the iterative Gaussian process regression with the IVM algorithm, which is referred to as SSuGPs [3]. The proposed method is inspired by IVM and SSuGPs algorithms, which is to deal with large-scale problems.

The rest of this paper is organized as follows. Section 2 reviews Gaussian processes and the IVM algorithm. Section 3 thoroughly introduces the sequential training method of semi-supervised classification. Section 4 reports our experimental results on eight real word data sets, including comparisons with other two related methods. Finally, Section 5 concludes this paper and gives future research direction.

2. Gaussian Processes and IVM

In this section, we will review the general Gaussian process methodology and how IVM uses the assumed density filtering approximation technique to approximate the true posterior probability to produce a sparse Gaussian process model. We also give a brief introduction of ADF.

2.1. Gaussian processes

A Gaussian process is a natural generalization of the Gaussian distribution whose mean and covariance is a vector and matrix. It defines a distribution over functions [1]. It is completely specified by its mean function and covariance function. For example,

$$f \sim \mathcal{GP}(m(x), k(x, x')), \tag{1}$$

means that the function f is distributed as a GP with mean function $m(x)$ and covariance function $k(x, x')$.

The learning of GPs involves learning of latent functions f of the input variables. Assuming that we are given a data set of N instances, the input data, $\mathrm{X} = [\mathrm{x}_1, \ldots, \mathrm{x}_N]^T$, and the targets $\mathrm{y} = [y_1, \ldots, y_N]^T$ are independent from the input data. A set of random variables, $\mathrm{f} = [f(\mathrm{x}_1), \ldots, f(\mathrm{x}_N)]^T$, are function values on the input data. The prior distribution over the latent variables is given by a GP,

$$p(\mathrm{f}) = \mathcal{N}(0, K),$$

with a covariance matrix K, which is evaluated at the input data X by a kernel function. The parameters associated with the kernel function are called kernel hyperparameters.

The joint likelihood can be written as

$$p(\mathrm{y}, \mathrm{f}) = p(\mathrm{f}) \prod_{n=1}^{N} p(y_n|f_n), \tag{2}$$

where $p(y_n|f_n)$ gives the relationship between the latent variable and the output target and is referred to as the noise model. For Gaussian process regression, the noise model is also a Gaussian process, that is

$$p(y_n|f_n) = \mathcal{N}(y_n|f_n, \sigma_n^2).$$

Therefore, it is easy to compute the marginal likelihood for Gaussian process regression. However, for non-Gaussian noise models the required marginalization is intractable and we must turn to approximations [7, 8, 9]. In this paper, we adopt the assumed density filtering to approximate the true process posterior.

One key advantage of the Gaussian process framework is that the hyperparameters are automatically optimized by maximizing the marginal likelihood. In the next subsection, we will review how IVM uses the ADF to compute the marginal likelihood to produce a sparse Gaussian process model.

2.2. IVM with ADF

The inspiration for the IVM is the support vector machine, of which the solution is sparse. The IVM algorithm seeks to find a sparse representation for the data set [7]. IVM uses the ADF algorithm to select only d most informative points in an online way based on the information theoretic selection criterion. Therefore, the resulting model is forced sparse and the computational complexity is reduced from $O(N^3)$ to $O(Nd^2)$. The selected points are referred to as representative points or active set.

Assumed density filtering (ADF) has its origins in online learning. It deals with only one data point at a time, computing the modified posterior and replacing it with an approximation to keep the algorithm tractable [4]. The joint likelihood (2) can be formulated as

$$p(\mathrm{y}, \mathrm{f}) = \prod_{n=0}^{N} t_n(\mathrm{f}),$$

where $t_0(\mathrm{f})$ is the prior of f, and $t_n(\mathrm{f}) = p(y_n|f_n)$. ADF makes use of this factorized form to approximate the true posterior, $p(\mathrm{f}|\mathrm{y})$, as $q(\mathrm{f})$. Initially, $q_0(\mathrm{f}) = t_0(\mathrm{f}) = \mathcal{N}(0, K)$. ADF proceeds in a sequential way by absorbing one data point at a time. Therefore, once a more point x_{n_i} is

included, it leads to an updated posterior distribution of this term,

$$\hat{p}_i(\mathrm{f}) \propto q_{i-1}(\mathrm{f}) t_{n_i}(\mathrm{f}).$$

Then, the new approximation of the true posterior, $q_i(\mathrm{f})$, can be obtained by minimizing the Kullback Leibler (KL) divergence between the two distributions.

$$\mathrm{KL}(\hat{p}_i||q_i) = -\int \hat{p}_i(\mathrm{f}) \log \frac{q_i(\mathrm{f})}{\hat{p}_i(\mathrm{f})} d\mathrm{f}. \tag{3}$$

Generally, the updated posterior after including the i-th point can be written as

$$\hat{p}_i(\mathrm{f}) = \frac{q_{i-1}(\mathrm{f}) t_{n_i}(\mathrm{f})}{Z_i}, \tag{4}$$

where the normalization constant is

$$Z_i = \int t_{n_i}(\mathrm{f}) q_{i-1}(\mathrm{f}) d\mathrm{f}. \tag{5}$$

The minimization of KL divergence (3) is achieved by 'moment matching' equations of the form

$$q_i(\mathrm{f}) = \mathcal{N}(\mathrm{f}|\mu_i, \Sigma_i).$$

In the process of data point selection, the point that gives the largest reduction in posterior process entropy is selected. The entropy change associated with including the n_ith point at the ith inclusion is given by

$$\Delta H_i = -\frac{1}{2} \log |\Sigma_{i,n_i}| + \frac{1}{2} \log |\Sigma_{i-1}|. \tag{6}$$

More details about the IVM and ADF algorithms can be found in [4].

3. The proposed method

In semi-supervised classification, we are given a training data set of L labeled examples, $D = \{(\mathrm{x}_i, y_i)|i = 1, \ldots, L\}$, $y_i \in \{-1, 1\}$, and a test data set of U unlabeled examples, $D' = \{\mathrm{x}_i^*|i = 1, \ldots, U\}$, $N = L + U$. Our aim is to use the training data D along with the test data D' to determine unknown labels y_i^*. We now give a brief overview of the sparse Gaussian process based approach for semi-supervised classification and then present the proposed sequential training method.

3.1. SSuGPs

SSuGPs is a GP-based algorithm for solving semi-supervised binary classification problem using sparse Gaussian process regression models [3]. It is closely related to semi-supervised learning based on support vector regression (SVR) and maximum margin clustering [5, 6].

In clustering problems, using large margin methods to design an algorithm is gaining wide popularity. Zhang et al proposed an iterative algorithm for maximum margin clustering based on support vector regression [5]. The goal of SVR is to find a function $f(\mathrm{x}^*) = \mathrm{w}^T \phi(\mathrm{x}^*) + b$ which best fits the data. Here, ϕ is a mapping function induced by a kernel function k. The primal problem of the iterative algorithm for maximum margin clustering can be written as

$$\begin{array}{ll} \min_{\mathrm{w},b,\xi_i^*} & \frac{1}{2}\|\mathrm{w}\|^2 + C\sum_i \xi_i^* \\ s.t. & |y_i^* - (\mathrm{w}^T\phi(\mathrm{x}^*) + b)| = \xi_i^* \ \forall i \\ & y_i{}^* \in \{-1, 1\} \forall i, \ -l \le e^T \mathrm{y}^* \le l, \end{array} \tag{7}$$

where $-l \le e^T \mathrm{y}^* \le l$ is a cluster balance constraint to avoid meaningless solutions, $l \ge 0$ is a constant controlling the class imbalance and e is the vector of ones.

For Gaussian process regression, the relationship between the latent variables and the targets can be written as $\mathrm{y} = \mathrm{f} + \varphi$, where φ is the Gaussian noise, $\varphi \sim \mathcal{N}(\varphi|\mathrm{b}, \sigma_{\mathrm{N}}^2 \mathrm{I})$. Let $\tilde{\mathrm{f}} = \left[\mathrm{f}^T \ \mathrm{f}^{*T}\right]^T$ and

$$\mathrm{K} = \begin{bmatrix} K(X,X) & K(X,X^*) \\ K(X^*,X) & K(X^*,X^*) \end{bmatrix}$$

denote the covariance matrix obtained using both the labeled and unlabeled examples. Therefore, the problem formulation (7) can be easily extended to solve semi-supervised classification problems making use of Gaussian process regression. That is,

$$\begin{array}{ll} \min_{\tilde{\mathrm{f}},b,\mathrm{y}^*} & \frac{\sigma_N^2}{2} \tilde{\mathrm{f}}^T \mathrm{K}^{-1} \tilde{\mathrm{f}} + C_L \sum_{i=1}^{L} \xi_i^2 + C_U \sum_{i=1}^{U} \xi_i^{*2} \\ s.t. & y_i - (f_i + b) = \xi_i, \ i = 1, \ldots, L \\ & y_i^* - (f_i^* + b) = \xi_i^*, \ i = 1, \ldots, U \\ & y_i^* \in \{-1, 1\} \forall i, \ -l \le e^T \mathrm{y}^* \le l. \end{array} \tag{8}$$

Problem (8) can be solved in an iterative way. Firstly, use only the labeled examples to train a GP classifier to predict y^* and b. Secondly, compute the dual problem of Problem (8) to find $\tilde{\mathrm{f}}$. Thirdly, with a fixed $\tilde{\mathrm{f}}$, y^* and b can be obtained by solving the following problem:

$$\begin{array}{ll} \min_{\mathrm{y}^*,b} & \sum_{i=1}^{L} \xi_i^2 + \sum_{i=1}^{U} \xi_i^{*2} \\ s.t. & y_i - (f_i + b) = \xi_i, \ i = 1, \ldots, L \\ & y_i^* - (f_i^* + b) = \xi_i^*, \ i = 1, \ldots, U \\ & y_i^* \in \{-1, 1\} \forall i, \ -l \le e^T \mathrm{y}^* \le l. \end{array} \tag{9}$$

Repeat the above two steps until convergence or the maximum number of iterations is reached. The IVM algorithm has the capability of selecting the most informative points to force a sparse model. Therefore, making use of the IVM algorithm to select representative points before the second step of the iterative GPR algorithm leads to a sparse GPR algorithm, that is SSuGPs [3].

3.2. The sequential training method based on SSuGPs

Due to a high demand in memory for dealing with large-scale data sets, it makes the current sparse GP-based semi-supervised algorithms difficult to solve complex problems. On the other hand, even if the sparse algorithms can be applied to solve the large-scale problems, it would have poor performance due to the complex structure of the large data set. Therefore, we propose to solve large-scale problems using existing GP-based algorithm in a sequential way. In this paper, we focus on the SSuGPs algorithm. The proposed algorithm proceeds as follows. Let the number of representative points be d_{max}, and the number of examples being dealt with in every iteration is N_d. Firstly, use the first N_d examples to train a SSuGPs model, the outputs are d_{max} representative points and the targets of unlabeled examples in the dealt N_d examples. Secondly, train another SSuGPs model on the d_{max} representative points got in the previous step and $N_d - d_{max}$ examples from the remainder examples. Repeat the two steps until all unlabeled examples are assigned targets. For more information about SSuGPs, please refer to [3]. The proposed algorithm is referred to as Se_SSuGPs.

The sequential training method of semi-supervised classification based on SSuGPs can be illustrated as follows.

- **Begin** Initialize d_{max}, N_d, σ_N and kernel hyperparameters.
- **Repeat**
  - Train a SSuGPs classifer on the current N_d examples.
  - Keep the d_{max} representative points and add $N_d - d_{max}$ examples from the reminder examples to form the new N_d examples.
- **Until** all unlabeled examples are assigned targets.
- **End**

4. Experiments

4.1. Data description and configuration

The proposed algorithm is tested on eight real world data sets comparing with SSuGPs and NCNM [4]. The eight data sets are presented on the following table. Of the eight data sets, the data sets pima, waveform, ionosphere and letter are from the UCI repository: *http://archive.ics.uci.edu/ml/*, the data set USPS is from *http://www-i6.informatik.rwth-aachen.de/~keysers/usps.html* and the rest data sets are obtained from *http://ida.first.fraunhofer.de/projects/bench/benchmarks.htm*.

Table 1. The data sets used in experiments.

dataset	classes	attributes	training & test
ionosphere	2	34	351
pima	2	8	768
waveform	3	21	5000
letter	26	16	20000
Splice	2	60	1000+2175
Thyroid	2	5	140+75
Ringnorm	2	20	400+7000
USPS	10	256	7291+2007

Following the settings in [3], let k be the difference of proportions of examples belonging to the two classes, and we set l in problem (8) to be $l = 2kU$, where U denotes the number of unlabeled examples. For all the experiments, we use the Gaussian kernel function defined by $k(\mathbf{x}_i, \mathbf{x}_j) = \nu \exp\{\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\}$. C_L and C_U are set to be 1 corresponding to the standard Gaussian process regression.

In the next subsection, we report the experimental results of Se_SSuGPs, SSuGPs and NCNM tested on eight real world data sets.

4.2. Experimental results

Firstly, we test the proposed algorithm and two competing algorithms SSuGPs and NCNM on eight data sets. As to multiple classes problems, we divide the classes to one and the rest. On the next Table 2, the results on data set waveform is '0 vs rest', letter is '6 vs rest' and USPS is '3 vs 5'. The data sets on which the proposed algorithm performs best are marked bold.

To evaluate the performance of semi-supervised classification algorithms on different proportions of labeled examples, we set the proportion of labeled examples from 0.01 to

Table 2. The average error rate (%) of three algorithms on eight data sets.

data set	Se_SSuGPs	SSuGPs	NCNM
ionosphere	**14.20±1.61**	16.76±3.68	17.90±3.53
pima	**43.36±2.92**	53.86±1.49	44.42±3.42
waveform	**10.39±0.80**	11.53±1.17	16.72±6.32
letter	**0.89±0.52**	0.90±0.48	1.10±0.18
Splice	28.14±8.82	**21.53±0.55**	31.36±1.40
Thyroid	**7.11±0.77**	16.00±4.81	8.00±2.67
Ringnorm	**14.83±1.36**	25.97±3.56	17.24±1.82
USPS(3vs5)	7.36 ±0.66	8.22 ±1.41	**7.09 ±1.24**

0.05 and then test the three algorithms on corresponding data sets. Test results on the USPS data set of NCNM, SSuGPs and Se_SSuGPs are reported on the next Table 3, Table 4 and Table 5.

Table 3. Test set errors (%) on the USPS data set using the NCNM algorithm.

digit	0.01	0.025	0.05
0	7.63±1.96	6.22±0.81	2.00±0.37
1	3.16±0.76	0.76±0.04	3.04±3.37
2	8.75±1.81	8.89±2.30	2.25±0.35
3	3.18±0.30	7.12±2.43	2.94±0.42
4	8.37±3.35	7.88±3.36	5.62±2.92
5	6.57±1.99	6.59±2.23	2.92±0.35
6	7.97±2.94	6.31±3.48	1.70±0.46
7	13.93±2.54	6.09±2.59	2.23±0.41
8	6.76±1.77	7.42±1.80	3.37±0.28
9	11.85±4.45	7.63±2.04	3.97±0.58

From experimental results shown on Table 2, Table 3, Table 4 and Table 5, we can see that the proposed algorithm performs better than the other two semi-supervised classification algorithms. When the proportion of labeled examples increases, the corresponding algorithms perform better in general.

5. Conclusions

In this paper, a sequential training method for solving semi-supervised binary classification problem is proposed. It assigns targets to test inputs sequentially making use of sparse Gaussian process regression models and finally results in a sequential and sparse method. It is closely related to the informative vector machine (IVM) technique and the iterative Gaussian process regression algorithm. Furthermore, the hyperparameters are estimated easily by maximizing the marginal likelihood without resorting to expensive cross-validation technique. When evaluated on several real world data sets, the proposed algorithm gets the best performance comparing with two related methods. The promising results demonstrate the effectiveness of the proposed method.

As a preliminary study of GP-based semi-supervised algorithm for solving large-scale problems, Se_SSuGPs seems to be a promising beginning. In the future, more sparse GP-based algorithms should be learnt for semi-supervised classification.

Table 4. Test set errors (%) on the USPS data set using the SSuGPs algorithm.

digit	0.01	0.025	0.05
0	4.70±1.00	2.97±1.15	2.08±0.29
1	2.57±4.08	0.77±0.04	3.30±2.84
2	6.82±1.79	4.46±0.89	2.31±0.22
3	4.26±0.76	3.40±0.32	3.16±0.37
4	6.84±3.17	6.75±3.60	5.46±2.52
5	6.25±1.76	6.15±2.16	2.95±0.39
6	6.49±3.14	4.23±1.46	1.60±0.26
7	7.93±2.41	4.56±1.95	1.83±0.33
8	6.23±1.23	3.26±0.40	3.17±0.40
9	6.89±2.37	4.46±2.49	3.78±0.22

Table 5. Test set errors (%) on the USPS data set using the Se_SSuGPs algorithm.

digit	0.01	0.025	0.05
0	4.29±0.90	2.62±0.96	1.92±0.33
1	0.95±0.10	0.91±0.12	2.80±2.79
2	6.92±2.12	4.07±0.77	2.15±0.29
3	4.24±0.66	3.46±0.41	2.92±0.42
4	6.87±1.91	4.40±1.97	5.69±2.49
5	5.90±1.35	4.13±0.63	2.90±0.18
6	3.26±1.14	2.35±0.52	1.50±0.20
7	2.95±0.95	1.66±0.20	1.73±0.22
8	4.78±0.41	3.22±0.46	3.24±0.67
9	4.38±2.37	3.68±0.51	3.24±0.57

Acknowledgements

This work is supported in part by the National Natural Science Foundation of China under Project 61075005, and the Fundamental Research Funds for the Central Universities.

References

[1] C. Rasmussen and C. Williams. “Gaussian Processes for Machine Learning”, MIT Press, 2006.

[2] S. Sun and J. Shawe-Taylor. “Sparse Semi-supervised Learning using Conjugate Functions”, Journal of Machine Learning Research, vol. 11, pp. 2423-2455, 2010.

[3] A. Patel, S. Sundararajan, and S. Shevade. “Semi-supervised Classification using Sparse Gaussian Process Regression”, in *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pp. 1193-1198, 2009.

[4] N. Lawrence, J. Platt, and M. Jordan. “Extensions of the Informative Vector Machine”, in *Proceedings of the Sheffield Machine Learning Workshop*, pp. 56-87, 2005.

[5] K. Zhang, I. Tsang, and J. Kwok. “Maximum Margin Clustering Made Practical”, in *Proceedings of the 24th International Conference on Machine Learning*, pp. 1119-1126, 2007.

[6] L. Xu, J. Neufeld, B. Larson, and D. Schuurmans. “Maximum Margin Clustering”, in *Advances in Neural Information Processing Systems*, pp. 1537-1544, 2004.

[7] N. Lawrence, M. Seeger, and R. Herbrich. “Fast Sparse Gaussian Process Methods: the Informative Vector Machine”, in *Advances in Neural Information Processing Systems*, pp. 625-632, 2003.

[8] V. Sindhwani, W. Chu, and S. Keerthi. “Semi-supervised Gaussian Process Classifiers”, in *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pp. 1059-1064, 2007.

[9] N. Lawrence and M. Jordan. “Semi-supervised Learning via Gaussian processes”, in *Advances in Neural Information Processing Systems*, pp. 753-760, 2005.