Neurocomputing 191 (2016) 95–106

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Kernel least mean square with adaptive kernel size

Badong Chen [a,*], Junli Liang [b], Nanning Zheng [a], José C. Príncipe [a,c]

[a] Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University, 28 Xianning West Road, Xi'an 710049, China
[b] School of Electronics and Information, Northwestern Polytechnical University, Xi'an, China
[c] Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611 USA

ARTICLE INFO

Article history:
Received 13 February 2014
Received in revised form
1 October 2015
Accepted 6 January 2016
Communicated by: Steven Hoi
Available online 5 February 2016

Keywords:
Kernel methods
Kernel adaptive filtering
Kernel least mean square
Kernel selection

ABSTRACT

Kernel adaptive filters (KAF) are a class of powerful nonlinear filters developed in Reproducing Kernel Hilbert Space (RKHS). The Gaussian kernel is usually the default kernel in KAF algorithms, but selecting the proper kernel size (bandwidth) is still an open important issue especially for learning with small sample sizes. In previous research, the kernel size was set manually or estimated in advance by Silverman's rule based on the sample distribution. This study aims to develop an online technique for optimizing the kernel size of the kernel least mean square (KLMS) algorithm. A sequential optimization strategy is proposed, and a new algorithm is developed, in which the filter weights and the kernel size are both sequentially updated by stochastic gradient algorithms that minimize the mean square error (MSE). Theoretical results on convergence are also presented. The excellent performance of the new algorithm is confirmed by simulations on static function estimation, short term chaotic time series prediction and real world Internet traffic prediction.

1. Introduction

Kernel based methods are successfully used in machine learning and nonlinear signal processing due to their inherent advantages of convex optimization and universality in the space of L_2 functions. By mapping the input data into a feature space associated with a Mercer kernel, many efficient nonlinear algorithms can be developed, thanks to the kernel trick. Popular kernel methods include support vector machine (SVM) [1,2], kernel regularization network [3], kernel principal component analysis (KPCA) [4], and kernel Fisher discriminant analysis (KFDA) [5], etc. These nonlinear algorithms show significant performance improvement over their linear counterparts.

Online kernel learning [36–39,47–58] has also been extensively studied in the machine learning and statistical signal processing literature, and it provides efficient alternatives to approximate a desired nonlinearity incrementally. As the training data are sequentially presented to the learning system, online learning requires, in general, much less memory and computational cost. Recently, kernel based online algorithms for adaptive filtering have been developed and have become an emerging area of research [6]. Kernel adaptive filters (KAF) are derived in Reproducing Kernel Hilbert Spaces (RKHS) [7,8], by using the linear structure and inner product of this space to implement the well-established linear adaptive filtering algorithms that correspond to nonlinear filters in the original input space. Typical KAF algorithms include the kernel least mean square (KLMS) [9,10], kernel affine projection algorithms (KAPA) [11], kernel recursive least squares (KRLS) [12], and extended kernel recursive least squares (EX-KRLS) [13], etc. With a radially symmetric Gaussian kernel they create a growing radial-basis function (RBF) network to learn the network topology and adapt free parameters directly from the training data. Among these KAF algorithms, the KLMS is the simplest, and fastest to implement yet very effective.

There are two main open challenges in the KAF algorithms . The first is their growing structure with each sample, which results in increasing computational costs and memory requirements especially in continuous adaptation scenarios. In order to curb the network growth and to obtain a compact representation, a variety of sparsification techniques have been applied, where only the important input data are accepted as new centers. The presently available sparsification criteria include the novelty criterion [14], approximate linear dependency (ALD) criterion [12] surprise criterion [15], and so on. In a recent work [16], we have proposed a novel method, the quantized kernel least mean square (QKLMS) algorithm, to compress the input space and hence constrain the network size which is shown to be very effective in yielding a compact network with desirable accuracy.

Selecting a proper Mercer kernel is the second remaining problem that should be addressed when implementing kernel adaptive filtering algorithms, especially when the training data

* Corresponding author. Tel.: +86 29 8266 8802x8009; fax: +86 29 8266 8672.
E-mail addresses: chenbd@mail.xjtu.edu.cn (B. Chen), liangjunli@xaut.edu.cn (J. Liang), principe@cnel.ufl.edu (J.C. Príncipe).

http://dx.doi.org/10.1016/j.neucom.2016.01.004
0925-2312/

size is small. In this case, the kernel selection includes two parts: first, the kernel type is chosen, and second, its parameters are determined. Among various kernels, the Gaussian kernel is very popular and is usually a default choice in kernel adaptive filtering due to its universal approximating capability, desirable smoothness and numeric stability. The normalized Gaussian kernel is

$$\kappa(\boldsymbol{u},\boldsymbol{u}') = \exp\left(-\|\boldsymbol{u}-\boldsymbol{u}'\|^2/2\sigma^2\right) \tag{1}$$

where the free parameter σ ($\sigma>0$) is called the kernel size (also known as the kernel bandwidth or smoothing parameter). In fact, the Gaussian kernel is strictly positive definite and as such produces a RKHS that is dense [8] and as such linear algorithms in this RKHS are universal approximators of smooth L_2 functions. In principle this means that in the large sample size regime the asymptotic properties of the mean square approximation are independent of the kernel size σ [46]. This means that the kernel size in KAF only affects the dynamics of learning, because in the initial steps the sample size is always small, therefore both the accuracy for batch learning and the convergence properties for online learning are dependent upon the kernel size. This should be contrasted with the effect of the kernel size in classification where the kernel size controls both the accuracy and the generalization of the optimal solution [1,2]. Up to now, there are many methods for selecting a kernel size for the Gaussian kernel borrowed from the areas of statistics, nonparametric regression and kernel density estimation. The most popular methods for the selection of the kernel size are: cross-validation (CV) [17–21] which can always be used since the kernel size is a free parameter, penalizing functions [18], plug-in methods [18,22], Silverman's rule [23] and other rules of thumb [24]. The cross-validation, penalizing functions, and plug-in methods are computationally intensive and are not suitable for online kernel learning. The Silverman's rule is widely accepted in kernel density estimation although it is derived under a Gaussian assumption and is usually not appropriate for multimodal distributions. Besides the fixed kernel size, some adaptive or varying kernel size algorithms can also be found in the literature [25–28]. This topic is also closely related to the techniques of *multi-kernel learning* or *learning the kernel* in the machine learning literature [40–45]. There the goal is typically to learn a combination of kernels based on some optimization methods, but in KAF this approach is normally avoided due to the computational complexity [6].

All the above mentioned methods, however, are not suitable for determining an optimal kernel size in online kernel adaptive filtering, since they either are batch mode methods or originate from a different problem, such as the kernel density estimation. Given that in online learning the number of samples is large and not specified a priori, the final solution will be practically independent of the kernel size. The real issue is therefore how to speed up convergence to the neighborhood of the optimal solution, which will also provide smaller network sizes. In the present work, by treating the kernel size as an extra parameter for the optimization, a novel sequential optimization framework is proposed for the KLMS algorithm. The new optimization paradigm allows for an online adaptation algorithm. At each iteration cycle, the filter weights and the kernel size are both sequentially updated to minimize the mean square error (MSE). As the kernel size is updated sequentially, the proposed algorithm is computationally very simple. The new algorithm can also be incorporated in the quantization method so as to yield a compact model.

The rest of the paper is organized as follows. In Section 2, we briefly revisit the KLMS algorithm. In Section 3, we propose a sequential optimization strategy for the kernel size in KLMS, and then derive a simple stochastic gradient algorithm to adapt the kernel size. In Section 4, we give some theoretical results on the convergence. Specifically, we derive the energy conservation relation in RKHS, and on this basis we derive a sufficient condition for the mean square convergence, and arrive at a theoretical value of the steady-state excess mean-square error (EMSE). In Section 5, we present simulation examples on static function estimation, short term chaotic time series prediction and real world Internet traffic prediction to confirm the satisfactory performance of the KLMS with adaptive kernel size. Finally, in Section 6, we present the conclusion.

2. KLMS

Suppose the goal is to learn a continuous input–output mapping $f:\mathbb{U}\to\mathbb{Y}$ based on a sequence of input–output examples (training data) $\{\boldsymbol{u}(i),y(i)\}_{i=1}^{N}$, where $\mathbb{U}\subset\mathbb{R}^m$ is the input domain, $\mathbb{Y}\subset\mathbb{R}$ is the desired output space. The hypothesis space for learning is assumed to be a Reproducing Kernel Hilbert Space (RKHS) $\mathscr{H}_k$ associated with a Mercer kernel $\kappa(\boldsymbol{u},\boldsymbol{u}')$, a continuous, symmetric, and positive-definite function $\kappa:\mathbb{U}\times\mathbb{U}\to\mathbb{R}$ [7]. To find such a function f, one may solve the regularized least squares regression in $\mathscr{H}_k$:

$$\min_{f\in\mathscr{H}_k}\sum_{i=1}^{N}(y(i)-f(\boldsymbol{u}(i)))^2+\gamma\|f\|_{\mathscr{H}_k}^2 \tag{2}$$

where $\|.\|_{\mathscr{H}_k}$ denotes the norm in $\mathscr{H}_k$, $\gamma\geq 0$ is the regularization factor that controls the smoothness of the solution. As the inner product in RKHS satisfies the *reproducing property*, namely, $\langle f|\kappa(\boldsymbol{u},.)\rangle_{\mathscr{H}_k}=f(\boldsymbol{u})$, (2) can be rewritten as

$$\min_{f\in\mathscr{H}_k}\sum_{i=1}^{N}\left(y(i)-\langle f|\kappa(\boldsymbol{u}(i),.)\rangle_{\mathscr{H}_k}\right)^2+\gamma\|f\|_{\mathscr{H}_k}^2 \tag{3}$$

By the *representer theorem* [8], the solution of (2) can be expressed as a linear combination of kernels:

$$f(\boldsymbol{u})=\sum_{i=1}^{N}\alpha_i\kappa(\boldsymbol{u}(i),\boldsymbol{u}) \tag{4}$$

The coefficient vector can be calculated as $\boldsymbol{\alpha}=(\mathbf{K}+\gamma\mathbf{I})^{-1}\boldsymbol{y}$, where $\boldsymbol{y}=[y(1),\cdots,y(N)]^T$, and $\mathbf{K}\in\mathbb{R}^{N\times N}$ is the Gram matrix with elements $\mathbf{K}_{ij}=\kappa(\boldsymbol{u}(i),\boldsymbol{u}(j))$.

Solving the previous least squares problem usually requires significant memory and computational burden due to the necessity of calculating a large Gram matrix, whose dimension equals the number of input patterns. The KAF algorithms, however, provide efficient alternatives that build the solution incrementally, without explicitly computing the Gram matrix. Denote f_i the estimated mapping (hypothesis) at iteration i. The KLMS algorithm can be expressed as [6]

$$\begin{cases} f_0=0 \\ f_i=f_{i-1}+\eta\kappa(\boldsymbol{u}(i),.)e(i) \end{cases} \tag{5}$$

where η denotes the step size, $e(i)$ is the instantenous prediction error at iteration i, $e(i)=y(i)-f_{i-1}(\boldsymbol{u}(i))$, i.e. the instantenous error only depends upon the difference between the desired response at the current time and the evaluation of the current sample ($\boldsymbol{u}(i)$) with the previous system model (f_{i-1}). The learned mapping of KLMS, at iteration N, will be

$$f_N(\boldsymbol{u})=\eta\sum_{i=1}^{N}e(i)\kappa(\boldsymbol{u}(i),\boldsymbol{u}) \tag{6}$$

This is a very nice result because it states that the solution to the unknown nonlinear mapping is done incrementally one step at a time, with a growing RBF network, where the centers are the samples and the fitting parameter is automatically determined as the current error.