Knowledge-Based Systems 89 (2015) 173–180

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Kernel online learning algorithm with state feedbacks

Haijin Fan [a,b,*], Qing Song [a], Xulei Yang [b], Zhao Xu [b]

[a] School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798, Singapore
[b] Institute of High Performance Computing, A*STAR, 1 Fusionopolis Way, #16-16 Connexis North, Singapore 138632, Singapore

ARTICLE INFO

Article history:
Received 7 October 2014
Received in revised form 1 July 2015
Accepted 8 July 2015
Available online 16 July 2015

Keywords:
Online learning
Recurrent kernel
Adaptive training
Weight convergence

ABSTRACT

This paper presents a novel recurrent kernel algorithm for online learning. It introduces a propagation scheme to recycle the kernel state information. The novel structure keeps records of the training sample information and incorporates it in the learning task over time to preserve the characteristics of the training sequences. In order to ensure the convergence of the algorithm, an adaptive training method is proposed to tune the kernel weight and recurrent weight simultaneously followed by detailed analysis of the weight convergence. Numerical simulations are presented to show the effectiveness of the proposed algorithm.

1. Introduction

Kernel methods are nonparametric tools with universal approximation capacity. The fundamental idea of kernel methods is that a Mercer kernel is applied to transform the low-dimensional feature vector into a high-dimensional reproducing kernel Hilbert space (RKHS). The increase of feature dimensionality makes many nonlinear problems linearly solvable in the RKHS. The inner product of two high-dimensional feature vectors in RKHS can be calculated efficiently without knowing the exact transform function, which is known as the popular "kernel trick". Underlying the RKHS, kernel methods provide powerful tools with linearity, convexity, and universal approximation capabilities for machine learning, data mining and pattern recognition.

In the last decade, a lot of kernel online learning (KOL) algorithms have been proposed for various applications for the structure simplicity and computational power of kernel methods. These algorithms include the type of kernel least mean square (LMS) algorithm [1,2], the kernel affine projection typed algorithm [3,4], and the type of kernel recursive least square algorithm [5]. The kernel online learning algorithms are used in many applications [6–8]. They were also extended to the multiple kernel online learning using multiple-scale kernels [9–11]. Due to the linearity of the output in kernel methods, the popular used linear square least algorithms are able to be generalized into the RKHS for the KOL algorithm updating. However, most of the current KOL algorithms are type of *feed-forward* networks. The system output only depends on its current feature inputs, with no explicit relations to the previous inputs or outputs. The *recurrent networks* have been very popular for their powerful capabilities in nonlinear modeling. Some successful examples of the recurrent networks can be seen in recurrent neural networks (RNNs) [12,13], recurrent SVMs [14–16], and recurrent radial basis function (RBF) networks [17,18]. The modeling ability of recurrent neural networks were enhanced especially in time series processing and motivated a lot of applications of other recurrent networks. Recently, some recurrent kernel algorithms are proposed and they perform very well in various applications. A example is the linear recurrent kernel online learning algorithm where the one-step previous output was applied to estimated the current output in a linear way [19]. There are some fundamental differences between kernel methods and artificial neural networks. Kernel methods are applied in many domains by the use of kernel functions. The inner products of two feature vectors can be easily calculated by a kernel and this operation is often computationally cheaper than their explicit computation in the high dimensional space. While Artificial neural networks are analogy to the neurons of human beings. Theoretically, kernel functions meet the Mercer's theorem but neural networks activation functions do not have this property. Structurally, kernel methods have only one layer, but neural networks can have multiple layers and the center of the kernel function is based on the data points. However, the center of activation function in neural network are usually randomly selected and evolved accordingly.

* Corresponding author at: School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798, Singapore.
E-mail addresses: hfan1@e.ntu.edu.sg, fanhiking@gmail.com (H. Fan).

http://dx.doi.org/10.1016/j.knosys.2015.07.001
0950-7051/

This paper presents a recurrent kernel algorithm with a novel structure. The feedback is from the kernel state vector (the output of the hidden-layer) and it is recycled at each training step and incorporated in the estimation of the current output. In such a novel way, the current state of the hidden-layer as a feedback is preserved to estimate the next-step state constrained by a recurrent weight. This type of scheme helps to keep records of the prior state information of training samples, which enables the algorithm to continuously apply the training sample information over time. Due to this feedback, the algorithm can preserve training sample information over multiple steps and this is especially significant in the case of applications involving time-series and streaming data. In the recurrent algorithm, an additional recurrent weight is applied to constrain the feedback of current state information. A LMS type training algorithm is proposed to tune the optimal kernel weight and recurrent weight simultaneously in an online fashion where at each training step one training sample is presented to the learning system. In order to guarantee the whole weight convergence and make the learning system converge fast in terms of external disturbance, an adaptive training method is proposed to adjust the training process. The main contributions of this paper are (1) it firstly introduces a novel recurrent structure for kernel online learning algorithm which recycles the hidden-layer state information; (2) it presents an adaptive training method to tune the kernel weight and recurrent weight simultaneously which ensures the convergence of the two weight parameters, while in our previous work [19], the kernel weight and the recurrent parameter were trained separately. The large amount of data is a challenge in data processing in both clustering and classification. A variety of suitable approaches have been proposed for this issue such as the scalable sparse subspace clustering applying l_1 norm constraint [20] and the locality representing strategy in [21]. The sparsification rules for kernel online learning have been widely investigated [2–5,22]. To curb the increasing growth of the kernel function number, the coherence-based criterion [22] is used for sparsification and a recurrent kernel algorithm is proposed for online learning applications. Previously, we have proposed a linear recurrent kernel (LRKOL) algorithm for online learning [19]. The novelty of the algorithm presented in this paper lies in: (1) The network structure is novel. In the previous LRKOL algorithm, the feedback is the one-step previous output; while in the algorithm proposed in this paper, the feedback is from the kernel state vector (the output of the hidden-layer) and it is recycled at each training step. (2) The training algorithm is different. In the proposed algorithm, it applies an adaptive training method to tune the kernel weight and recurrent weight simultaneously in a unified framework. While in the previous LRKOL algorithm, the kernel weight and the recurrent parameter were trained separately.

The organization of this paper is as follows. In Section 2, we introduce some fundamental ideas of kernel online learning algorithms. In Section 3 and Section 4, the proposed recurrent kernel online learning algorithm is presented. The detailed network structure and training method are described followed by the analysis of its weight convergence. In Section 5, numerical simulation results of two examples are presented and finally conclusion is given in Section 6.

2. Kernel online learning algorithms

Suppose that $\mathcal{H}$ is a Hilbert Space and $\langle\cdot,\cdot\rangle_{\mathcal{H}}$ is the inner product in the Hilbert Space. A mapping function $\varphi(\cdot)$ transfers the input feature space $\mathcal{U}$ to a higher dimensional space $\mathcal{H}$. The inner product of two feature vectors in the RKHS become [23]:

$$\langle\varphi(\boldsymbol{u}(i)),\varphi(\boldsymbol{u}(j))\rangle_{\mathcal{H}}=\kappa(\boldsymbol{u}(i),\boldsymbol{u}(j)) \tag{1}$$

where $k(\cdot,\cdot)$ is a positive definite and symmetric kernel function. The inner product of two feature vectors in the higher dimensional feature space can be easily computed by (1) without knowing the exact function form of $\varphi(\cdot)$, which is usually called the *kernel trick*. This idea is extensively used in the kernel methods.

Given the feature input-desired output sequence $\{\boldsymbol{u}(j),d(j)\}_{j=1}^{t}$, kernel methods aim to find a function $f(\cdot)$ to best fit the output $f(\boldsymbol{u}(t))=\langle f_t(\cdot),\kappa(\cdot,\boldsymbol{u}(t))\rangle_{\mathcal{H}}$. By virtue of the representer theorem [23], the function $f_t(\cdot)$ can be expressed as linear form in the RKHS

$$f_t(\cdot)=\boldsymbol{\omega}^T(t)\varphi(\cdot) \tag{2}$$

where $\boldsymbol{\omega}(t)$ is the weight coefficient and it can be expressed as a linear combination of the feature vectors in the RKHS

$$\boldsymbol{\omega}(t)=\sum_{j=1}^{t}\alpha_j(t)\varphi(\boldsymbol{u}(j)). \tag{3}$$

By the kernel trick, the output is

$$f_t(\cdot)=\sum_{j=1}^{t}\alpha_j(t)\kappa(\cdot,\boldsymbol{u}(j)) \tag{4}$$

where $\kappa(\cdot,\boldsymbol{u}(j))$ is a kernel function with its center being the input feature vector $\boldsymbol{u}(j)$ and $\alpha_j(t)$ is the kernel weight coefficient. However, in classical kernel methods, the kernel function number becomes very large as the number of training data continuously increases. The large number of kernel functions makes the algorithms not only be in danger of over-fitting but also with a high computational complexity. In online learning context, to curb the growing size of network, representative kernel function centers should be chosen. Recently, different sparsification methods were proposed to address this issue. They aimed to select a compact dictionary with finite size using different criteria including the coherence-based criterion, approximate linear dependency (ALD) criterion, novelty criterion and the information theoretic method [24,5,22,25]. In sparse kernel representation, supposed that a sparse dictionary $\mathcal{D}(t)=\{\boldsymbol{c}_j(t),\}_{j=1}^{m}$ with m members is obtained, the estimated function becomes

$$f_t(\cdot)=\sum_{j=1}^{m}\alpha_j(t)\kappa(\cdot,\boldsymbol{c}_j(t)) \tag{5}$$

where the number of kernel functions is largely reduced.

3. State recurrent kernel online learning algorithm

The novel recurrent kernel algorithm introduces a feedback loop in the hidden-layer of the network. Compared with the feed-forward kernel algorithm, the output of the proposed recurrent kernel algorithm depends both on the current kernel evaluation vector and the previous kernel state. The explicit network structure is shown in Fig. 1. Consider the current input $\boldsymbol{u}(t)$, the estimated output $y(t)$ is determined by the current kernel evaluation vector $\boldsymbol{k}(t)$ and the previous kernel state $\boldsymbol{s}(t-1)$ in the recurrent loop, which can be formulated as

$$\begin{cases}\boldsymbol{s}(t)=\boldsymbol{\Lambda}(t)\boldsymbol{s}(t-1)+\boldsymbol{k}(t)\\ y(t)=\boldsymbol{s}(t)^T\boldsymbol{\alpha}(t)\end{cases} \tag{6}$$

where $\boldsymbol{k}(t)\in\mathcal{R}^{m\times 1}$ is the kernel evaluation vector of the feature input based on the existing dictionary $\mathcal{D}(t)=\{\boldsymbol{c}_1(t),\ldots,\boldsymbol{c}_m(t)\}$ and it is defined as

$$\boldsymbol{k}(t)=[\kappa(\boldsymbol{u}(t),\boldsymbol{c}_1(t)),\ldots,\kappa(\boldsymbol{u}(t),\boldsymbol{c}_m(t)]^T. \tag{7}$$

The state matrix $\boldsymbol{\Lambda}(t)=\mathrm{diag}\{\lambda_1(t),\ldots,\lambda_m(t)\}$ is the diagonal matrix form of the recurrent weight $\boldsymbol{\lambda}(t)=[\lambda_1(t),\ldots,\lambda_m(t)]^T\in\mathbb{R}^{m\times 1}$ and