A Dictionary Updating Scheme Incorporating Words Elimination into Quantized Kernel Least-Mean-Squares for Changing Environments

Lei Sun*,†, Badong Chen∗, Shengyu Nan†, Zhiping Lin*, Kar-Ann Toh§

* School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, 639798
† Institute of Artificial Intelligence and Robotics, Xi’an Jiaotong University, Xi’an, PR China, 710049
‡ School of Information and Electronics, Beijing Institute of Technology, Beijing, PR China, 100081
§ School of Electrical and Electronic Engineering, Yonsei University, Seoul, South Korea, 120-749
E-mail addresses: sunlei@ntu.edu.sg (L. Sun), chenbd@mail.xjtu.edu.cn (B. Chen), n_worm712@163.com (S. Nan), ezplin@ntu.edu.sg (Z. Lin), and katoh@ieee.org (K. Toh).

Abstract—Learning under time-varying environment is a challenging task since one has to deal with the ever changing distribution of data. A common and yet effective solution is to learn the data online and keep up with any ongoing changes. The Quantized Kernel Least-Squares (QKLMS) is an effective tool for online dictionary learning where the network size is capped by the quantization dictionary size. However, due to the lack of a mechanism to eliminate outdated words, learning can become irrelevant over time. In this paper, a mechanism to remove irrelevant words in the dictionary is proposed for QKLMS. Our experimental results based on chaotic time sequence prediction validate the capability of the developed method for time-varying data adaptation.

Index Terms—Changing environment learning; Online learning; Vector quantization; Kernel learning machine

I. INTRODUCTION

Modern real world learning applications often encounter the problem of changing environment where the change is termed as concept drift [1], [2]. To adapt to the changing environment, online learning (also known as incremental learning) methods are desirable since input data are sequentially processed where an environmental adaptation methodology can be implemented to keep up with the change [3]. Online kernel learning has been widely recognized as state-of-the-arts [4] where a radial basis function (RBF) can be adopted to map the input samples to a high dimensional feature space with desirable discriminability. For example, kernel online learning [5] and kernel adaptive filtering [4] are all RBF based online learning methods. However, a common problem to these RBF based learning methods is that the size of learning network is linearly growing with respect to the increasing of input samples. This linear and unlimited growth in network size pose both computational and memory challenges particularly when continuous tracking or monitoring is needed.

To address the network growing problem in online kernel learning, a variety of sparsification techniques have been proposed to curb the growth of the learning network. For example, quantized kernel least-mean-squares (QKLMS) has been proposed in [6] where a quantization mechanism is adopted for sample selection. Its variant version namely quantized kernel recursive least squares (QKRLS) using recursive learning strategy is subsequently proposed in [7]. The two quantization based methods show effectiveness of using the computationally simple vector quantization (VQ) for sparse representation. It is noted that there is no words elimination mechanism in QKLMS and the words are permanently kept in dictionary regardless of whether the input data distribution is changing. In [8], a fixed-budget quantization scheme has been proposed where a dynamic dictionary updating scheme is adopted to avoid explosion in dictionary size. Inspired by this work, we explore a dynamic dictionary updating scheme by simply deleting those outdated words to adapt to the changing environment.

In an changing environment with a moderate or slow drift [3], the input data distribution is varying moderately upon time. In this paper, a words elimination mechanism is proposed for inclusion in the quantization scheme to adapt the QKLMS algorithm to the changing environment. By eliminating those outdated words from dictionary, the quantization output gradually represents the current learning data chunk instead of representing all the input data. That is, the quantization outputs are tightly related to the dynamic change of current data chunk and a significant computational memory reduction is expected. The performance of the proposed method is shown by simulation on a chaotic sequence with time changing distribution.

II. REVIEW OF QUANTIZED KERNEL LEAST-MEAN-SQUARES

The goal of a RBF learning network is to learn a nonlinear function \( f : \mathbb{R}^d \rightarrow \mathbb{R} \) based on an input sequence \( \{(x_i, y_i)\}_{i=1}^N \) where \( x_i \in \mathbb{R}^d \) are the input data and \( y_i \in \mathbb{R} \) are the targets to be matched by the learned function \( f \), and the sequence size \( N \) can approach infinity in a time sequence setting. For each input \( x_i \), the output of a RBF learning network with \( K \) nodes...
is a linear combination of the nodes
\[ f(x_i) = \sum_{k=1}^{K} \alpha_k \phi_k(x_i) \] (1)

where \( \phi_k(\cdot) \) is the \( k \)th node and \( \alpha_k \) denotes the \( k \)th weight. By using squared errors, the learning goal is to minimize the sum of squared fitting errors
\[ J(\alpha) = \sum_{i=1}^{N} (y_i - f(x_i))^2 \]

where \( \alpha \) denotes the weight vector. For simplicity, in this paper, the Gaussian kernel function is adopted as a representative RBF function:
\[ \phi_k(x_i) = \exp \left( -\frac{||x_i - u_k||^2}{\sigma^2} \right), \quad k = 1, \ldots, K \] (2)

where \( u_k \) and \( \sigma^2 \) denote the kernel centers and the kernel width of Gaussian kernel function, respectively.

Kernel least mean squares (KLMS) is an online implementation of the RBF based kernel learning machine [4]. In the KLMS settings, the input data are sequentially processed and a kernel node is added to the RBF learning network upon each input data. In other words, the \( j \)th (\( j = 1, \ldots, N \)) input data \( x_j \) is taken as the \( j \)th kernel center (\( u_j = x_j \)) to parameterize the \( j \)th kernel node as
\[ \phi_j(x_i) = \exp \left( -\frac{||x_i - x_j||^2}{\sigma^2} \right) \]

and the network size is linearly growing with respect to the number of input data.

To curb the linear growth of network size in KLMS, the QKLMS algorithm [6] uses a simple vector quantization (VQ) scheme (called primitive quantization scheme (PQS) hereafter) to represent the input samples by a compressed quantization subset. Consequently, a significant reduction in network size is expected since the kernel nodes are now parameterized by the compressed subset as the subset size is limited.

![Fig. 1. Sample quantization model](image)

The quantization scheme PQS [6], [7] is illustrated in Fig. 1. In this figure, the input samples are encoded to a set of outputs by the quantizer. Specifically, each of the input samples \( x_i \) is compared with the elements \( c_q \) from the quantization dictionary \( \{c_q\}_{q=1}^{Q} \) to obtain the quantized outputs \( \hat{x}_i \) by
\[ q^* = \arg \min_{q=1,\ldots,Q} D(x_i, c_q), \]
\[ \hat{x}_i = g(x_i) = c_{q^*}, \quad i = 1, \ldots, N, \] (3)

where \( D(x_i, c_q) = ||x_i - c_q|| \) denotes the Euclidean distance between the two points \( x_i \) and \( c_q \). The quantization dictionary elements \( c_q \), namely dictionary words, are updated by the input samples. For example [6], an input sample is newly added to the dictionary if the minimal distance between the dictionary words and the input sample is beyond a threshold \( \varepsilon \) (called quantization radius parameter) as
\[ c_{Q+1} = x_i, \quad Q = Q + 1, \quad \text{if} \quad \min_{q=1,\ldots,Q} D(x_i, c_q) > \varepsilon. \] (4)

It is noted that there is no dictionary words elimination mechanism in the PQS (and then the QKLMS) which means that once a word is added to the dictionary, it will be permanently kept there.

### III. DICTIONARY UPDATING WITH WORDS ELIMINATION IN QKLMS

The lack of dictionary words elimination mechanism results in at least two drawbacks. One drawback is that once an outlier (in the sense of “negative” impact on learning performance) is added to the dictionary, the newly added word could negatively impact on learning performance during all the subsequent learning process. The other drawback is that when learning in a changing environment, the input data span might not be timely represented by the stored dictionary words.

The representation of the timely changing input data span is important to learning which is shortly shown by means of significance evaluation [9] of a kernel node. For the \( i \)th input \( x_i \), the QKLMS output with \( Q \) dictionary words is [6]
\[ f(x_i) = \eta \sum_{q=1}^{Q} \alpha_q \phi_q(x_i) = \eta \sum_{q=1}^{Q} \alpha_q k_{q}(x_i, c_q) \] (5)

where \( k_{q}(x_i, c_q) = \phi_q(x_i) = \exp \left( -\frac{||x_i - c_q||^2}{\sigma^2} \right) \) denotes the kernel node corresponding to the \( q \)th dictionary word and \( \alpha_q \) denotes the \( q \)th coefficient. The significance of the \( q \)th kernel node is defined as the resulting error by removing the \( q \)th kernel node in learning network as [9]
\[ S_q(i) = \eta |\alpha_q| \sum_{j=1}^{i} k_{q}(x_j, c_q) \]
\[ = \eta |\alpha_q| \int_{\mathbb{R}^d} k_{q}(x, c_q) p(x) dx, \] (6)

where \( p(x) \) denotes the probability density function (PDF) of \( x \). By the Parzen window estimator [10], \( p(x) \) can be estimated by the Gaussian kernels given by \( \hat{p}(x) = 1/N \sum_{j=1}^{N} k_{q}(x, x_j) \) where we use the same kernel width \( \sigma^2 \) as that in (5) for simplicity. By integral calculations in (6), the \( q \)th node significance can be expressed as
\[ S_q(i) = \eta |\alpha_q| \sqrt{\frac{\pi \sigma^2}{i}} \sum_{j=1}^{i} \exp \left( -\frac{||x_i - c_q||^2}{\sigma^2} \right). \] (7)

In (7), it is noted that the significance is related to two arguments: the absolute value of coefficient \( \alpha_q \) and the distances between the \( q \)th dictionary word \( c_q \) and the samples \( x_j, j = 1, \ldots, i \). For the \( q \)th kernel node, it is clear that the neighbouring samples of \( c_q \) make more contributions to the significance than the distant samples. That is to say, if there is no sample around the \( q \)th dictionary word \( c_q \), its corresponding kernel node significance tends to insignificant. In other words,
a dictionary word makes sense if there are some samples located in its neighborhood. This analysis implicates that the quantization dictionary need to be adaptively updated upon the input data distribution. Especially, under the changing environment assumption where the data distribution is timely changing, the quantization dictionary need to be timely updated. For example, an existing dictionary word makes no sense if there is no input sample around the word in the current data chunk, and hence the word can be removed from the dictionary. In summary, an quantization dictionary words elimination mechanism can be adopted without significance losing. This motivates our proposal of a modified vector quantization scheme to adaptively match the span of the timely learning data chunk.

Algorithm 1 Vector quantization updating scheme
Input: \( \{x_i\}_{i=1}^{N} \in \mathbb{R}^d \), \( x_i \in \mathbb{R}^d \).
Output: \( x_i \in \{c_q\}_{q=1}^{Q} \), \( c_q \in \mathbb{R}^d \).

1: Parameter settings: \( t_{th}, \varepsilon \)
2: Initialization: \( Q = 1 \), \( C = [x_1], t = [t_{th}] \).
3: while \( x_i \) is available do
4: \( q^* = \text{arg min}_{q=1,..,Q} D(x_i, c_q) \).
5: if \( D(x_i, c_{q^*}) \geq \varepsilon \) then
6: \( C = [C, x_i], t = [t, t_{th}], Q = Q + 1, q^* = Q \).
7: end if
8: \( x_i = c_{q^*}, t[q^*] = t_{th} \).
9: \% included dictionary words elimination procedure
10: for \( j = 1, \ldots, i \) do
11: if \( t[j] < 0 \) then
12: \( C[j] = [], t[j] = [], Q = Q - 1 \).
13: end if
14: end for
15: end while

Based on the PQS, the modified quantization scheme (MQS) is given in Algorithm 1. In this algorithm, \( C \) denotes the quantization dictionary which is also the quantized output set, \( Q \) denotes the number of elements in the dictionary. The quantization radius parameter \( \varepsilon \) determines whether a word is to be added to the quantization dictionary. The mentioned three parameters are the same as that in PQS [6], however, there is a newly included time label vector \( t \) to control the updating of the words in dictionary. The vector \( t \) records the time label of each dictionary word. Each time label of dictionary word will be reset to \( t_{th} \) when the word is activated by an input sample quantized to the word. The time label will decrease by 1 upon each input sample and a dictionary word will be regarded as “deactivated” when its time label is decreased to 0. Consequently, the deactivated word will be removed from the quantization dictionary. Replacing the PQS in QKLMS by the MQS and keeping other parts in QKLMS unchanged, the QKLMS is thus updated to modified QKLMS (MQKLMS).

IV. Simulation

In this section, two experiments are implemented to respectively evaluate the MQKLMS on simulated stationary data sequence and drifting data sequence. Mackey-Glass time sequence is adopted as the chaotic time sequence in experiments. Mackey-Glass equation was first developed to model a physiological control systems of electrolytes, oxygen, glucose, and blood cells in the pressure to the brain and various organs [11]. It generates a nonlinear time-delay of periodic and chaotic sequence which has been widely recognized as a benchmark sequence for sequential algorithm evaluation [4], [10], [6], [12]. The adopted Mackey-Glass equation is

\[
\frac{dx(t)}{dt} = \frac{0.2x(t-\tau)}{1 + x(t-\tau)} - 0.1x(t),
\]

where \( \tau = 30 \) and \( x(0) = 1.2 \). A Mackey-Glass time sequence is imposed by a zero-mean additive Gaussian noise with variance 0.01. A Mackey-Glass time sequence example with 1800 samples is shown in the top panel of Fig. 2.

![Fig. 2. Mackey-Glass time sequence prediction. A Mackey-Glass time sequence with 1800 samples is shown in the top panel. The add-on bias component is shown in the middle panel to simulate the drifting component in input data. Time sequence mixed with the drifting component is shown in the bottom panel.](image)

In experiments, the problem setting for Mackey-Glass time sequence prediction is as follows: the previous 5 points \( x_i = [x(i-5), \ldots, x(i-1)]^T \) are used as the input vector to predict the current value \( x(i) \) which is the desired response. Testing Mean-squared-errors (MSE) is calculated based on the average of predictions of the coming 50 samples where MSE is defined as

\[
\text{MSE} = \frac{1}{N_t} \sum_{j=i}^{i+N_t-1} (x_j - \hat{x}_j)^2
\]

where \( j \) denotes the index of the testing data with the total number of \( N_t = 50 \) samples, and \( \hat{x}_j \) denotes an estimation of \( x_j \) (see eq. (5)).

The KLMS and QKLMS are two included methods for comparison. In all experiments, the Gaussian kernels are adopted where kernel width are set at \( \sigma^2 = 1 \) for all algorithms. The quantization radius parameters in QKLMS and MQKLMS are
both set at $\epsilon = 0.5$ and the learning rates are set at 0.005. In MQKLMS, the time threshold is set at $t_{th} = 100$. The final MSE performance is obtained from the average of 100 runs for all algorithm evaluations.

The fluctuations in QKLMS and MQKLMS are relatively small which means that they are more immune to the drifting components than KLMS. The learning network size on the drifting time sequence is shown in the right column of Table I. Comparing the two columns in table, the learning network size in QKLMS is increased from 30 in (a) to about 84 in (b). This increment of learning network size can be attributed to the expansion of data span from (a) to (b). In (a), the data is stationary and the data span (about $[-0.5, 0.5]$) is stable which results in a relatively small learning network size of 30 in QKLMS. However, there is an add-on drifting component in (b) and the resulted data span is extended to about $[-2.5, 2.5]$. Since there is no dictionary words elimination mechanism in QKLMS, the dictionary size increases to match the enlarged data span which results in the increased kernel node number of 84 in (b). On the contrary, the MQKLMS adopts a dictionary words elimination mechanism. Hence, the MQKLMS learning network sizes in (a) and (b) are similar which means that the MQKLMS adapts to the add-on drifting components well in terms of learning network size. In Fig. 3, it is observed that elimination of the timely deactivated dictionary words does not decrease the learning MSE performance. In summary, the MQKLMS shows its better adaptation capability to drifting data than that of QKLMS.

![Fig. 3. Mackey-Glass time sequence prediction.](image)

<table>
<thead>
<tr>
<th>Algorithms</th>
<th># nodes in (a)</th>
<th># nodes in (b)</th>
</tr>
</thead>
<tbody>
<tr>
<td>KLMS</td>
<td>1800</td>
<td>1800</td>
</tr>
<tr>
<td>QKLMS</td>
<td>30</td>
<td>83.9</td>
</tr>
<tr>
<td>MQKLMS</td>
<td>19.6</td>
<td>17.7</td>
</tr>
</tbody>
</table>

(a) denotes the results on simulated stationary data and (b) denotes the results on simulated drifting data. In the first experiment, prediction is implemented on the original Mackay-Glass time sequence simulating stationary input data and the learning curve is shown in Fig. 3 (a). In this figure, it can be observed that the learning performance of all the three methods: KLMS, QKLMS and MQKLMS are comparable and overall the same. However, the numbers of kernel nodes of the three methods are quite different. For comparison, each of the number of kernel nodes in the last evaluation step is shown in the middle column of Table I. From this table, it is seen that the 1800 nodes in KLMS are largely reduced to 30 in QKLMS, and the number is further reduced to about 20 in MQKLMS. In summary, evaluations on the simulated stationary time sequence show that the MQKLMS attains not only learning network size reduction but also comparable performance.

Next, in order to evaluate the three methods over changing environment, an extra shifting component is added to the original signal to simulate data drifting which is shown in the middle panel of Fig. 2. It can be seen that the drifting component is slowly changing in index region [1000, 1300]. The time sequence mixed with this drifting component is shown in the bottom panel of Fig. 2. The changing regions are highlighted with two vertical dashed lines in figure.

Evaluation performance on this drifting time sequence is shown in Fig. 3 (b). In this figure, a fluctuation in KLMS is clearly seen which can be attributed to the input data drifting.

V. CONCLUSION

A dynamic quantization method by simply eliminating outdated words in dictionary has been proposed for online kernel learning. It has been shown that those timely deactivated dictionary words could hardly contribute significantly to the learning of current data chunk and removal of these deactivated words reduces the learning network size. The timely deactivated dictionary words were removed in order to adapt the quantization outputs to input data changing. Evaluation by prediction on chaotic time sequence with drifts validated the proposed method. The results showed that the proposed method adapted to drifting data better than that of QKLMS.

ACKNOWLEDGEMENT

This research was partly supported by Basic Science Research Program through the National Research Foundation of Korea funded by the Ministry of Education, Science and Technology (NRF-2012R1A1A2042428), by Beijing Institute of Technology Basic Science Research Program (BIT-20130542011) funded by BIT, China, by National Natural Science Foundation of China (No.61372152) and by 973 Program (No.2015CB351703).

REFERENCES


