An Online Active Broad Learning Approach for Real-Time Safety Assessment of Dynamic Systems in Nonstationary Environments

Zeyi Liu, Yi Zhang, Zhongjun Ding, and Xiao He, *Senior Member, IEEE*

***Abstract*— Real-time safety assessment of the complex dynamic systems in nonstationary environments is of great significance for avoiding the potential hazards. In this case, the update procedure with high assessment accuracy and training speed is crucial and meaningful in the dynamic streaming setting. Generally, the performance of most online learning approaches will be negatively affected by limited annotated samples in such a setting. Moreover, the time cost of advanced conventional methods with retaining procedures is relatively high, constraining their practicality. In this article, a novel online active broad learning approach, termed OABL, is proposed. In detail, the effectiveness of the broad learning system in the framework of online active learning is first revealed and verified. A reasonable dynamic asymmetric query strategy is then designed with a limited annotation budget to actively annotate the relatively valuable samples, which is beneficial to mitigating the negative effects of class imbalance. In this context, the advantage of the human-in-the-loop characteristic is also effectively used to control the evolution direction of the learner during the incremental update, which makes it better able to adapt to complex and nonstationary environments. Several related experiments are conducted with the realistic data of JiaoLong deep-sea manned submersible. Results show the effectiveness and practicality of the proposal compared with the existing advanced approaches.**

***Index Terms*— Broad learning system (BLS), dynamic system, online active learning, real-time safety assessment (RTSA).**

I. Introduction

SAFETY assessment of dynamic systems such as the deep-sea manned submersible (DSMS) is of great significance [1], [2], [3]. Since the system environment constantly changes during the operation process, such as diving or ascent (i.e., nonstationary environments), submariners must pay close attention to unknown risks that cannot be avoided during operation. In this case, their work efficiency may be reduced to a certain extent. Hence, the design of real-time safety assessment (RTSA) approaches is an emerging and essential topic that can reduce the possibility of significant losses. Taking the JiaoLong DSMS as an example, the submariner needs to provide certain feedback in time to adjust the evolution direction of the assessment model. However, several objective constraints need to be overcome to solve such RTSA tasks.

Numerous advanced studies in the field of real-time fault diagnosis (RTFD) have been widely applied in real-life scenarios [4], [5], [6], [7] However, the related technologies have some limitations to be directly introduced for resolving the RTSA problems. Generally, a system's safety level can be defined as the impact degree of its operation on potential damage to the surrounding people and the environment. Therefore, it is relatively easier for the safety assessment results to be influenced by the nonstationary environments. In other words, the underlying distributions of monitored data are changed constantly, which means that the cases of concept drift widely exist and are hard to be analyzed [8], [9], [10]. In addition, more typical features can be summarized as follows.

1) *Human-in-the-loop characteristic*: There are experienced submariners in the DSMS. However, it is impossible to annotate the full unlabeled data sequentially. Namely, the feedback of humans is essential but limited.
2) *Limited actual physical memory space*: Limited by the space of the submersible, it is difficult for the model to access large-scale historical data. Hence, the design of incremental update procedures in online learning is needed.
3) *Extremely class imbalance characteristic*: The amount of data in a *safe level* (positive class) is much larger than that in an *unsafe level* (negative class), which makes the actual effect of the existing model greatly limited.

It seems reasonable to tackle the RTSA problems in complex scenarios with a learning-based framework. However, learning in nonstationary environments is still a challenging issue [11], [12]. As discussed in [13], four data-driven methods are commonly used to create the learner for applications in nonstationary environment: naive predictor, nearest cluster predictor, single-layer linear network, and multilayer perceptron (MLP). Evaluations of the technique using multiple indicators

Manuscript received 20 January 2022; revised 2 June 2022 and 25 August 2022; accepted 11 November 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 61733009, in part by the National Key Research and Development Program of China under Grant 2017YFA0700300, and in part by the Huaneng Group Science and Technology Research Project. *(Corresponding author: Xiao He.)*

Zeyi Liu is with the Department of Automation, Tsinghua University, Beijing 100084, China.

Yi Zhang is with the College of Shipbuilding Engineering, Harbin Engineering University, Harbin 150001, China, and also with the National Deep Sea Center, Qingdao 266061, China (e-mail: zy592@ndsc.org.cn).

Zhongjun Ding is with the National Deep Sea Center, Qingdao 266061, China (e-mail: dzj@ndsc.org.cn).

Xiao He is with the Department of Automation, Beijing National Research Center for Information Science and Technology (BNRist), Tsinghua University, Beijing 100084, China (e-mail: hexiao@mail.tsinghua.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2022.3222265.

Digital Object Identifier 10.1109/TNNLS.2022.3222265

such as false positive rate (FPR) and false negative rate (FNR) showed that the structure of the MLP had the best performance [13]. Hence, it is believed that the utilization of models with neural network architecture is effective for resolving such issues [14], [15], [16].

However, only a small number of existing literature tried to develop the models with the neural network architecture to process time-series data in the streaming setting. The main reason is that the number of parameters is too large to be adjusted, which tends to cause higher time costs. In addition, such models are prone to underfitting under the constraints of limited annotation. As a result, some advanced online learning methods have been developed such as *perceptron* [17] and passive-aggressive (PA) algorithm [18]. Such type of method attempts to update the learner with the utilization of the gradient of the loss function $J(\cdot)$ and learning rate η at each time [i.e., $w_t = w_{t-1} + \eta \nabla J(\widehat{y}_t, y_t)$]. To improve the convergence rate and effectiveness as much as possible, plenty of online learning algorithms using second-order information have also been proposed, such as the confidence weighted-based (CW) algorithm [19], [20]. Although these methods provide a meaningful idea of updating learner weights, it is unrealistic for real-life industrial applications to obtain the true label in real-time to calculate the gradient of the loss function. Furthermore, it is hard to set the hyperparameters reasonably in real-life scenarios, especially for the concept drift cases [10], [21], [22], [23]. In this context, numerous advanced online learning approaches have been proposed with the codesign of the classifiers, which are generally extended by the classical methods for handling such issues. For instance, Huang et al. [24] developed an online sequential extreme learning system (OSELM) algorithm to update the model sequentially. An online bagging (OB) approach was proposed in [25]. While in [26], the adaptive windowing (ADWIN) technique [27] was then introduced to detect the concept drift. Furthermore, the adaptive random forest (ARF) algorithm, extended by the classical random forest algorithm, was also proposed recently [28].

In general, the model can be updated more reasonably if the selection of samples is annotated appropriately. In other words, the design of a query strategy is beneficial to improving the assessment effects for resolving the RTSA problems. To tackle such an issue, the field of online active learning has drawn more and more attention in recent years. Since the update procedures of several existing online learning models rely on the calculation of loss, their assessment accuracy may not be high enough for handling RTSA tasks with limited annotation. Another typical idea is to query the samples with high uncertainty [29]. For instance, Liu et al. [30] proposed a dual-query strategy to handle gradual drift and abrupt drift. However, the negative impact of class-imbalanced data is simply considered by only few existing literature. One typical work is the cost-sensitive online active learning (CSOAL) approach proposed in [31]. Furthermore, an online asymmetric active learning (OAAL) approach was developed to query the samples based on different preferences. While in [32], Zhang et al. introduced a novel query strategy that tries to exploit the second-order information of the samples. Although such methods have achieved good results, there are still some limitations. For instance, the sample value is greatly affected by the settings of the query parameters. Moreover, the used classifier models are generally strictly restricted.

In [33], Chen and Liu proposed the concept of the broad learning system (BLS), which is inherited from the random vector functional-link neural network (RVFLNN) [34]. BLS uses ridge regression to calculate the pseudoinverse required by the pretrained model, which avoids directly calculating the pseudoinverse of a huge matrix. In addition, the authors provided an idea of using the generalized inverse calculation to design incremental model updates [35]. Although BLS has attracted more and more attention [36], [37], [38], the existing literature has limited discussions on its performance of specific tasks under the framework of online active learning. How to control the evolution direction of the model still has further exploration significance.

In this article, a novel online active broad learning (OABL) approach is proposed for the RTSA tasks of dynamic systems. The main contributions are summarized as follows.

1) To the best of our knowledge, it is the first study to reveal and verify the effectiveness of the BLS in the framework of online active learning.
2) A novel dynamic asymmetric query strategy (DAQS) is designed to reduce the annotation cost of the users (submariners) in class imbalance scenarios, which is beneficial to improving the assessment accuracy.
3) Several experiments are used with the realistic data of the JiaoLong DSMS to show the effectiveness and practicality of the proposal. The results show that our proposed approach outperforms the other advanced methods.

The remainder of this study is organized as follows. In Section II, the proposed OABL approach is presented in detail. The experimental analysis is also presented in Section III to show the effectiveness and practicality of the proposed methodology. Related discussion and comparison are also mentioned. In Section IV, this article is summarized by some concluding remarks.

II. Proposed OABL Approach

A. Problem Statement

In this work, a classification task is considered as the general procedure of the RTSA tasks. For any period of monitoring data of the JiaoLong DSMS, the data come in the form of an imbalanced data stream. Notably, the underlying distributions of monitored data are changed constantly. Assume that there is a data stream as $\mathcal{D} = \{(x_1, y_1), \ldots, (x_{|\mathcal{D}|}, y_{|\mathcal{D}|})\}$. An initial labeled data $\mathcal{L}_0 = \{(x_1, y_1), \ldots, (x_{|\mathcal{L}_0|}, y_{|\mathcal{L}_0|})\}$ with $|\mathcal{L}_0|$ samples can be obtained to construct a pretrained classifier. Each instance $x_i \in \mathbb{R}^d$ $(i \in \{1, \ldots, n\})$ is a d-dimensional feature vector, and $y_i \in \{0, 1, \ldots, m\}$ is the class label of x_i for classification.

The annotation budget $\mathcal{B}$ is defined as the maximum number of queries for the submariners under some specific constraints. Our task is to assess the safety level of the dynamic systems with the limited annotation budget $\mathcal{B}$. The general process is

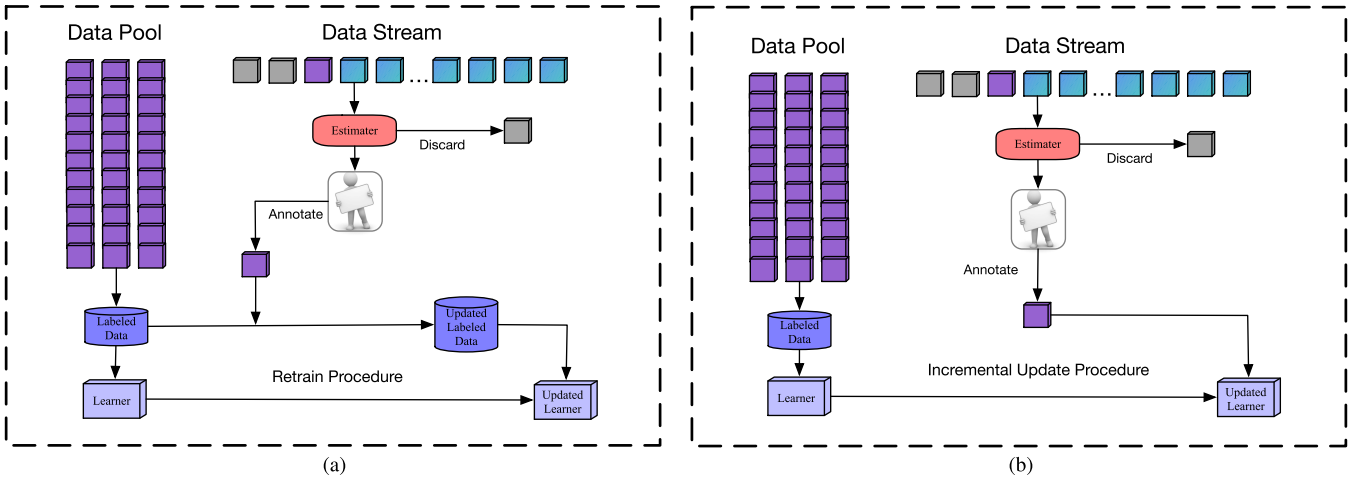


Fig. 1. Schematic of two typical online active learning frameworks. (a) Schematic of the online active learning framework with the retraining procedure. (b) Schematic of the online active learning framework with the incremental update procedure.

to judge whether a real-time instance x_t should be labeled with the constraint of limited budget $\mathcal{B}_t$ and query strategy θ. If the current sample is judged by the estimator to be annotated, the submariners are required to provide the actual label of x_t. Otherwise, the current sample can be discarded. To solve such problems, two typical frameworks in the field of online active learning can be obtained, as shown in Fig. 1. The classifier can be updated accordingly by performing a procedure of retraining [as shown in Fig. 1(a)] or incremental updating [as shown in Fig. 1(b)]. The query procedure will stop when either the task is finished or the annotation budget is exhausted.

Concerning the main typical features mentioned in Section I, it is not feasible to record the whole historical data. In this context, the retraining procedure should not be considered since it will make the time cost relatively high. Instead, we aim to design a data-driven classifier that can modify the model parameters in real-time incrementally. Moreover, the query strategy should also be carefully considered to control the evolution direction of the model in the imbalanced data stream. Hence, it is needed to design a data-driven classifier that can modify the model parameters incrementally and reasonably in real-time with the framework of online active learning.

B. Model Description

In this article, an OABL approach for RTSA of dynamic systems is proposed. The detailed implementation procedure is illustrated carefully in the following. Notably, OABL follows the structure of an incremental BLS [33], which is a type of learning system without the need for deep architecture. It is mainly composed of input, feature mapping nodes, enhancement nodes, and output. The weights between input and feature mapping nodes and between feature mapping nodes and enhancement nodes are generated randomly. Therefore, BLS does not have many hyperparameters and complex structures that require long-term training procedures.

For the RTSA task, related experienced engineers and submariners are able to annotate several samples based on the historical data initially. For any initial labeled data $L_0 \in \mathbb{R}^{|\mathcal{L}_0|\times d}$ and encoded output $Y_0 \in \mathbb{R}^{|\mathcal{L}_0|\times m}$, the mapping relationship between the input nodes and the ith features nodes can be obtained as

$$Z_i = \phi\left(L_0 W_{e_i} + \beta_{e_i}\right) \tag{1}$$

where $i \in \{1, 2, \ldots, N_1\}$ and $Z^{N_1} = [Z_1, Z_2, \ldots, Z_{N_1}]$. N_f represents the number of feature nodes for any group. W_{e_i} and β_{e_i} denote the weights and bias, respectively, which are randomly generated. We have N_1 groups of feature nodes totally. As mentioned in [33], the linear inverse problem can be applied to exploit the characteristics of the sparse autoencoder. The initial W_{e_i} can be fine-tuned to obtain relatively better features.

For the jth group of enhancement nodes, we have

$$H_j = \xi\left(Z^{N_1} W_{h_j} + \beta_{h_j}\right) \tag{2}$$

where $j \in \{1, 2, \ldots, N_2\}$ and $H^{N_2} = [H_1, H_2, \ldots, H_{N_2}]$. N_e represents the number of enhancement nodes for any group. Similarly, W_{h_j} and β_{h_j} also denote the weights and bias, respectively, which are randomly generated. $\xi(\cdot)$ represents a nonlinear mapping such as tanh function.

Hence, the mapping between the initial labeled data $\mathcal{L}_0$ and the encoded output Y_0 with such architecture can be obtained as

$$\begin{aligned} Y_0 &= \Big[Z_1, Z_2, \ldots, Z_{N_1} \Big| \ \xi(Z^{N_1} W_{h_1} + \beta_{h_1}) \\ &\qquad + \xi(Z^{N_1} W_{h_2} + \beta_{h_2}) + \cdots + \xi(Z^{N_1} W_{h_{N_2}} + \beta_{h_{N_2}})\Big] W^{N_2} \\ &= \Big[Z_1, Z_2, \ldots, Z_{N_1} \Big| H_1, H_2, \ldots, H_{N_2}\Big] W^{N_2} \\ &= \Big[Z^{N_1} \Big| H^{N_2}\Big] W^{N_2} \\ &= \Phi_0(\mathcal{L}_0, W_e, \beta_e, W_h, \beta_h) \end{aligned} \tag{3}$$

where Φ_0 represents the initial pretrained model.

As a result, we have

$$W^{N_2} = \Big[Z^{N_1} \Big| H^{N_2}\Big]^{\dagger} Y_0 \tag{4}$$

where $[Z^{N_1}|H^{N_2}]^{\dagger}$ represents the pseudoinverse of $[Z^{N_1}|H^{N_2}]$. In detail, W^{N_2} can be solved using *ridge regression* approximation of $[Z^{N_1}|H^{N_2}]^{\dagger}$ with (5)

$$W^{N_2} = \arg\min_{W} \|Y_{\mathcal{L}} - AW\| + \lambda \|W\|_2^2 \tag{5}$$

where $\lambda \in (0, 1)$ is a hyperparameter to control the degree of penalty for numerical larger weights. As discussed in [35], we have

$$W^{N_2} = \left(\lambda I + \left[Z^{N_1}\middle|H^{N_2}\right]\left[Z^{N_1}\middle|H^{N_2}\right]^T\right)^{-1}\left[Z^{N_1}\middle|H^{N_2}\right]^T Y_0 \tag{6}$$

where I denotes the identity matrix. Such a solution is equivalent with (5) since it gives an approximation to the *Moore–Penrose generalized inverse* [39]. In this case, the inverse problem degenerates into the least-square problem and leads the solution to the original pseudoinverse if $\lambda \to 0$. For simplicity, we let $A_{N_2}^{N_1}$ denote $[Z^{N_1}|H^{N_2}]$ in the following. Hence, we have

$$W^{N_2} = \lim_{\lambda \to 0}\left[\lambda I + A_{N_2}^{N_1}\left(A_{N_2}^{N_1}\right)^T\right]^{-1}\left(A_{N_2}^{N_1}\right)^T Y_0. \tag{7}$$

Clearly, it is necessary to evaluate whether it is valuable enough to be queried for any new instance $x_t \in \mathbb{R}^d$ at time t. With the annotation budget $\mathcal{B}$, the new instance should be discarded if there is a high degree of confidence that such a new instance will not change the evolution direction of the model. In this case, with the model Φ_{t-1} and corresponding W_e, β_e, W_h, and β_h, we have

$$A_t = \Big[\phi\big(x_t W_{e_1} + \beta_{e_1}\big), \ldots, \phi\big(x_t W_{e_{N_1}} + \beta_{e_{N_1}}\big) \times \Big| \xi\big(Z_t^{N_1} W_{h_1} + \beta_{h_1}\big), \ldots, \xi\big(Z_t^{N_1} W_{h_{N_1}} + \beta_{h_{N_2}}\big)\Big] \tag{8}$$

where

$$Z_t^{N_1} = \Big[\phi\big(x_t W_{e_1} + \beta_{e_1}\big), \phi\big(x_t W_{e_2} + \beta_{e_2}\big), \ldots, \phi\big(x_t W_{e_{N_1}} + \beta_{e_{N_1}}\big)\Big]. \tag{9}$$

A_t denotes the mapping results of x_t with the model Φ_{t-1}. In this context, the encoded prediction $\hat{Y}_t$ can be determined as

$$\hat{Y}_t = \Phi_{t-1}(x_t, W_e, \beta_e, W_h, \beta_h) = A_t^T W^{N_2} \tag{10}$$

where $\hat{Y}_t$ denotes the belief degree of each safety level for x_t. Without loss of generality, two classes are assumed here as *safe level* (positive class C^+) and *unsafe level* (negative class C^-). In other words, $m = 2$ so that $\hat{Y}_t \in \mathbb{R}^{\{1\times 2\}}$ and $Y_0 \in \mathbb{R}^{|\mathcal{L}_0|\times 2}$. The number of positive class is assumed to be larger than the number of negative class empirically based on the real-life scenarios.

For dealing with the RTSA tasks, the number of samples with the *safe level* is generally much larger than the number of samples with the *unsafe level*. The importance of samples near the classification boundary should be relatively higher. In addition, the exploitation of minority class samples for updating will also be of great significance to the improvement of model performance for this type of imbalanced data stream. One typical idea is to judge the samples individually based on these two criteria. However, it is not flexible and may result in more budget to be spent. In this context, we hope that the criterion should be more stringent for judging the value of the samples with the majority class. Meanwhile, for the minority class samples, we hope that the judgment criterion of its value should be appropriately slacked. Furthermore, the model is desired to have the adaptation ability to the environment as good as possible if concept drift occurs. Specifically, if concept drift happens at time t, the prediction results of the current model for x_t tend to be biased. On one hand, the differences in the belief degrees of the prediction concerning the safety level may reduce (i.e., the uncertainty of the prediction increases). On the other hand, the imbalance ratio may change dynamically. In the existing literature, fixed query parameters for different classes generally need to be set, which may not be conducive to dealing with problems caused by *nonstationary environments* such as concept drift.

It is common and reasonable to judge whether the query procedure should be executed based on a well-defined Bernoulli random variable $\delta_t \in \{0, 1\}$ [40]. As mentioned above, the budget for annotation is limited. The experts only need to annotate the label if $\delta_t = 1$ at time t. Inspired by [32] and [41], in this article, a novel DAQS is provided to deal with the imbalance issue of such RTSA tasks. With (10), the *prediction margin* τ_t between such two classes with Y_t can be simply expressed as

$$\tau_t = \hat{Y}_t[C_t^+] - \hat{Y}_t[C_t^-] \tag{11}$$

where $\hat{Y}_t[C_t^+]$ and $\hat{Y}_t[C_t^-]$ are the query parameters which denote the belief degree of prediction for positive class and negative class, respectively. Clearly, $\tau_t > 0$ means the model prefers to predict that x_t should be classified as a positive class and vice versa. A lower value of τ_t usually means that x_t is closer to the classification boundary. In other words, the *value* of instance x_t should be relatively higher in this case.

Let η_t^+ and η_t^- denote the imbalance tradeoff ratio for positive class and negative class, respectively. For time 1 to $t-1$, the number of samples that are predicted as positive class and negative class can be easily recorded which are denoted as $|C_{t-1}^+|$ and $|C_{t-1}^-|$, respectively. If $\tau_t > 0$, we have $|C_t^+| = |C_{t-1}^+| + 1$,while $|C_t^-| = |C_{t-1}^-|$. If $\tau_t < 0$, we have $|C_t^+| = |C_{t-1}^+|$, while $|C_t^-| = |C_{t-1}^-| + 1$. In this context, we have

$$\eta_t^+ = \begin{cases} \dfrac{|C_t^+|}{|C_t^+| + |C_t^-|} = \dfrac{|C_{t-1}^+| + 1}{|C_{t-1}^+| + |C_{t-1}^-| + 1}, & \text{if } \tau_t > 0 \\ \dfrac{|C_t^+|}{|C_t^+| + |C_t^-|} = \dfrac{|C_{t-1}^+|}{|C_{t-1}^+| + |C_{t-1}^-| + 1}, & \text{if } \tau_t < 0 \end{cases} \tag{12}$$

and

$$\eta_t^- = \begin{cases} \dfrac{|C_t^-|}{|C_t^+| + |C_t^-|} = \dfrac{|C_{t-1}^-|}{|C_{t-1}^+| + |C_{t-1}^-| + 1}, & \text{if } \tau_t > 0 \\ \dfrac{|C_t^-|}{|C_t^+| + |C_t^-|} = \dfrac{|C_{t-1}^-| + 1}{|C_{t-1}^+| + |C_{t-1}^-| + 1}, & \text{if } \tau_t < 0. \end{cases} \tag{13}$$

To avoid the samples with high uncertainty in the early stage being forcibly ignored, it is reasonable to set $|C_0^-|$ and $|C_0^+|$ as a small value such as 0.1. For any time t, we have

$$Pr(\delta_t = 1) = \begin{cases} \dfrac{\eta_t^-}{\eta_t^- + |\tau_t|}, & \text{if } \tau_t > 0 \\ \dfrac{\eta_t^+}{\eta_t^+ + |\tau_t|}, & \text{if } \tau_t < 0. \end{cases} \tag{14}$$

As shown in (14), the following relationships can be easily observed with the assumption that $|C_t^+| \gg |C_t^-|$.

1) If $\tau_t > 0$, the relatively smaller the $|\tau_t|$, the relatively larger the $Pr(\delta_t = 1)$.
2) If $\tau_t < 0$, the relatively smaller the $|\tau_t|$, the relatively larger the $Pr(\delta_t = 1)$.
3) For any specific $|\tau_t|$, the cases for $\tau_t > 0$ are overall easier to obtain a numerically smaller value of $Pr(\delta_t = 1)$ than the cases for $\tau_t < 0$.

In addition, it is also reasonable to judge whether x_t should be queried by setting the fixed threshold ϵ directly. If $Pr(\delta_t = 1)$ is numerically larger than ϵ, x_t is then annotated. Notably, the samples of the minority class are easier to be selected to execute the query procedure with the strategy proposed in this article. The preference will dynamically change with the learning process. Submariners can adjust the setting of different ϵ due to their annotation capabilities.

If x_t is not believed to be queried, such an instance can be directly discarded. Otherwise, our next step is to incrementally update the learner Φ_{t-1}. On this basis, the updating mapping matrix can be formulated as

$${}^{(t)}A_{N_2}^{N_1} = \begin{bmatrix} {}^{(t-1)}A_{N_2}^{N_1} \\ A_t^T \end{bmatrix}. \tag{15}$$

As demonstrated in [33], the associated pseudoinverse updating algorithm can be defined as

$$\left({}^{(t)}A_{N_2}^{N_1}\right)^{\dagger} = \left[\left({}^{(t-1)}A_{N_2}^{N_1}\right)^{\dagger} - B_t D_t^T \middle| B_t \right] \tag{16}$$

where

$$D_t^T = A_t^T \left({}^{(t-1)}A_{N_2}^{N_1}\right)^{\dagger} \tag{17}$$

$$C_t = A_t^T - D_t^T \left({}^{(t-1)}A_{N_2}^{N_1}\right) \tag{18}$$

and

$$B_t^T = \begin{cases} C_t^{\dagger}, & \text{if } C_t \neq 0 \\ \left(1 + D_t^T D_t\right)^{-1} \left({}^{(t-1)}A_{N_2}^{N_1}\right)^{\dagger} D_t, & \text{if } C_t = 0. \end{cases} \tag{19}$$

Hence, the updated weight is defined as

$${}^{(t)}W^{N_2} = {}^{(t-1)}W^{N_2} + \left(Y_t^T - A_t^T {}^{(t-1)}W^{N_2}\right) B_t \tag{20}$$

where Y_t represents the encoded label of x_t provided by the submariners.

To demonstrate the logic flow as clearly as possible, the general procedure of the OABL approach can be summarized as Algorithm 1.

Algorithm 1 General Procedure of OABL Approach

Input: Initial labeled data: $\mathcal{L}_0$, unlabeled data: $\mathcal{U}_0$, initial encoded label Y_0, query strategy: θ, budget $\mathcal{B}$, current remained budget for time t: $\mathcal{B}_t$, instance for time t: $x_t \in \mathbb{R}^d$
Output: The incremental learner Φ.

```
Obtain the mapping nodes Z^{N_1} with (1);
Obtain the enhancement nodes H^{N_2} with (2);
Calculate the W^{N_2}, A_{N_2}^{N_1}, and (A_{N_2}^{N_1})^† with (3) to (7);
Let B_0 = B;
while dynamic system is not STOP do
    Receive an instance x_t;
    Calculate A_t with (8) and (9);
    Obtain the encoded prediction Ŷ_t with (10);
    Draw σ_t ∈ {−1, +1} with (12)–(14);
    if B_t > 0 then
        if σ_t = +1 then
            Query the actual encoded label of Y_t;
            Calculate the combined transformed data (t)A_{N_2}^{N_1}
            with (15);
            Obtain the associated pseudoinverse ((t)A_{N_2}^{N_1})^†
            with (16)–(19);
            Calculate the updated weight (t)W^{N_2} via (20);
            Let (t)W^{N_2} → W^{N_2} ;
            (Incrementally update of the learner Φ);
            Let B_t = B_{t−1} − 1;
            Let L_t = L_{t−1} ⋃ {x_t};
            Let U_t = U_{t−1} \ {x_t};
        else
            Discard the instance x_t;
            Let B_t = B_{t−1};
            Let L_t = L_{t−1};
            Let U_t = U_{t−1};
        end
    else
        STOP Annotation
    end
    if task is finished then
        Let dynamic system STOP;
    else
        pass;
    end
end
```

III. Experiment

A. Experimental Setting

The data used in this article were collected and provided by the *National Deep Sea Center* in Qingdao, Shandong, China. The initial data are collected in the 151st exploration task for the JiaoLong DSMS on March 19th, 2017. The form of data is the multivariate time series with around 11 features for about 3 h. The subsystem of the JiaoLong DSMS such as *life support system* is selected for analysis. Our purpose is to assess the safety level, such as *safe level* and *unsafe level*, for the tested data stream.

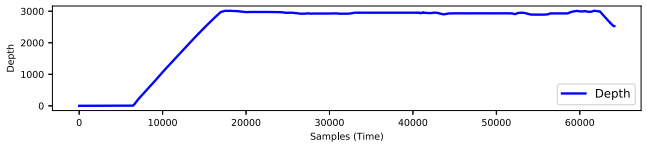


Fig. 2. Schematic of the process monitoring data for *depth*.

The seven variables of the life support system represent the real-time monitoring indicators, including *oxygen (O_2) concentration in the cabin*, *carbon dioxide (CO_2) concentration in the cabin*, *cabin pressure* (Pa), *cabin humidity*, and *cabin temperature* (°C). In addition, *backup 2* and *backup 3* are also considered, representing the oxygen concentration of another set of oxygen sensors and the carbon dioxide concentration of another set of carbon dioxide sensors, respectively. Two settings of oxygen and carbon dioxide sensors are set up in the cabin as a backup to monitor the oxygen and carbon dioxide concentration at any time. Hence, the monitoring data of backup 2 and backup 3 are the values of another set of oxygen and carbon dioxide sensors. Moreover, four settings of other external monitoring variables containing *salinity*, *temperature*, *depth*, and *speed* are selected. The monitoring data are sourced from the value of the conductivity temperature depth (CTD) sensor installed outside the JiaoLong DSMS, which monitors the real-time seawater salinity, temperature, depth, and the speed of ascending and descending.

During the experiment, the JiaoLong DSMS gradually descended from the shore to around 3000 m. The trend of the descent process over time is shown in Fig. 2. The related ten remaining monitoring indicators are also shown in Fig. 3. It is worth noting that the used labels are derived from the records of real voyage logs obtained in the JiaoLong DSMS. The total number of samples is 64 138. Around 12 samples of the data stream (0.02%) are randomly selected as the initial labeled data to pretrain the model. Considering the simplicity of the original data, we add some Gaussian noise to change the underlying distributions of the data to enlarge the influence of concept drift. The overall imbalance ratio of safe level and unsafe level is around 5.086 : 1.

B. Evaluation

In this article, the model is assumed to predict the sample at each timestamp after the related operation is executed. We then compare the true label with the prediction. Two different evaluation methods are considered in the following experiment.

On one hand, the overall effects of the model can be expressed by the indicator of *accuracy rate*, which can be defined as

$$\mathcal{A}_t^k = \frac{\left|C_t^{\alpha}\right| + \left|C_t^{\beta}\right|}{\left|C_t^{+}\right| + \left|C_t^{-}\right|} \tag{21}$$

where $|C_t^{\alpha}|$ denotes the total number of samples that are predicted correctly without updates from time 1 to t. And $|C_t^{\beta}|$ indicates the total number of samples predicted correctly with updates from time 1 to t. With n random tests, the average accuracy rate $\overline{\mathcal{A}}$ and standard deviation (SD) (i.e., $\overline{\mathcal{A}} \pm \sigma$) can be obtained.

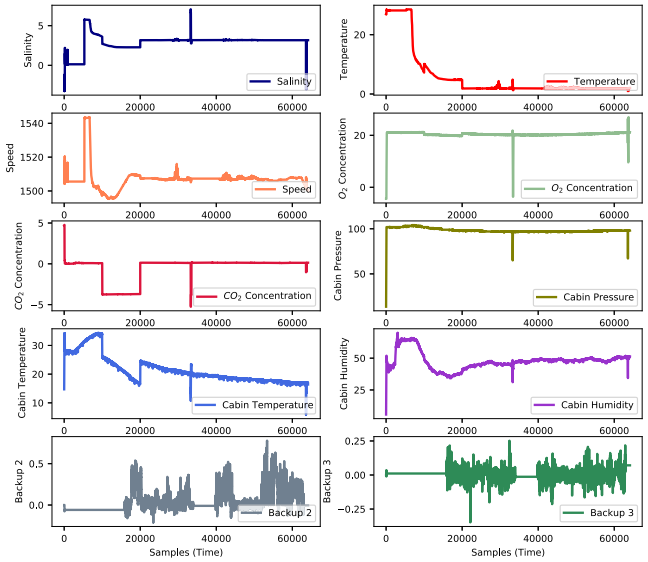


Fig. 3. Schematic of the process monitoring data for related ten remaining monitoring indicators.

On the other hand, the effects of the model can also be expressed as the indicator of *average performance* at each specific time point. Let $\mathcal{P}_t^k$ represent the *average performance* at time t for the k-th random test with the following rules:

$$\mathcal{P}_t^k = \begin{cases} 1, & \text{if } \arg\max_y Y_t^k = \arg\max_y \widehat{Y}_t^k \\ 0, & \text{otherwise} \end{cases} \tag{22}$$

where $\arg\max_y Y_t^k$ and $\arg\max_y \widehat{Y}_t^k$ denote the ground-truth safety level and the predicted safety level for the k-th random test, respectively. The average performance $\overline{\mathcal{P}}$ and SD (i.e., $\overline{\mathcal{P}} \pm \sigma$) can be obtained similarly with n random tests. The larger the value of evaluation indicators (i.e., $\overline{\mathcal{A}}$ and $\overline{\mathcal{P}}$), the better the overall performance of the model.

Moreover, $\overline{\mathcal{A}}$ tends to describe the overall performance of the model. Due to the cumulative effect, prediction errors in the early stage may have a greater impact on the outcome of $\overline{\mathcal{A}}$ than in the later stage. In comparison, $\overline{\mathcal{P}}$ tends to describe the average performance of each point. Therefore, if a model performs better on $\overline{\mathcal{A}}$ and worse on $\overline{\mathcal{P}}$, it is believed to perform better in the early stage of RTSA than in the later stage.

C. Results and Comparison

As mentioned in Section I, for solving RTSA problems, most batch learning algorithms will be inapplicable since they are designed to process batches of data with mainly offline learning algorithms. Most of them attempt to use historical data and labels to train the learner. In this context, many online learning approaches are proposed to handle such issues [8], [42]. In this article, three different experiments are used to validate the advantages of the proposed OABL approach, which are explained as follows. The experiments in this section are executed in Python 3.7. Notably, the main implementation of these experiments is based on open-source

packages such as *ALiPy* [43], *scikit-learn* [44], and *scikit-multiflow* [45]. In detail, the typical classifiers are implemented based on the *scikit-learn*. While the advanced online learning approaches, such as ARF and OB, are implemented based on the *scikit-multiflow*. For the implementation of BLS, refer to the uniform resource location (URL).[1] Furthermore, the OSELM is implemented based on the *pyoselm* Python library.[2]

1) Experiment I: In this subexperiment, five typical classifiers are considered as the comparison to show the learning efficiency.

1) *Support vector machine (SVM) model with radial basis function (RBF) kernel function*: a well-known classifier that is defined with the largest margin in the feature space [46].
2) *Logistic regression (LR) model*: an effective classifier that is widely used to deal with real-life online learning industrial problems [47].
3) *MLP model*: a classical fully connected neural network structure to train the data with the backpropagation algorithm [48].
4) *K-nearest neighbor (KNN) model*: a model which tries to classify the data based on the distance measure. The retraining procedure to be performed iteratively is needed [49].
5) *BLS model*: an RVFLNN-based model [33].

Our primary purpose is to test the learning efficiency for solving the RTSA tasks. Hence, three different scenarios with full annotation are considered. For the SVM, LR, MLP, and KNN models, the retraining procedure is used to update these models. In contrast, the corresponding incremental update procedure is used for the BLS. We set $\lambda = 0.001$, $N_1 = 10$, $N_2 = 10$, $N_f = 100$, and $N_e = 100$. More detailed settings of default parameters can be referred to [43], [44], [45]. Furthermore, $\xi(\cdot)$ is set as the *sigmoid* function. It is worth noting that the optimal hyperparameters are not needed. The main reason is that the retraining procedure to update the model is not realistic for solving the RTSA tasks. Therefore, the learning efficiency of different models with default parameters on the RTSA datasets should be more concerned. As mentioned in Section III-A, only 12 samples are still exploited to obtain the pretrained model.

The detailed performances with two random tests are shown in Table I. On one hand, the performance of BLS under some settings is close to (or even exceeds) the performance of other classical models. Therefore, it is believed that BLS can be effectively used to handle the RTSA tasks in nonstationary environments. On the other hand, the ratio of consumed time required for BLS compared with SVM shows a decreasing trend with the increase in $|\mathcal{D}|$, which indicates that BLS has high learning efficiency under full annotation. The main reason is that it only needs to calculate the pseudoinverse once in the subsequent incremental update. In addition, BLS naturally does not need to adjust the hyperparameters (such as MLP) due to its structural characteristics inherited from RVFLNN, making it is more suitable for solving real-life RTSA problems. The idea of the update process is similar to the mentioned online updating methods, such as the *perceptron* algorithm and its variants, which can still be seen as fine-tuning based on loss concerning the original learner weight to some degree, as shown in (20). In addition, the *Greville theorem* was introduced to complete the update, which shows its high interpretability [33].

TABLE I
DETAILED PERFORMANCES WITH TWO RANDOM TESTS IN EXPERIMENT I

Datasets	*Model*	$\overline{\mathcal{A}} \pm \sigma$	$\overline{\mathcal{P}} \pm \sigma$	*Time*
$\lvert\mathcal{D}\rvert = 250$	SVM	0.833 ± 0.01	0.824 ± 0.00	$1.00 * \mathcal{T}$
	LR	0.975 ± 0.00	0.970 ± 0.00	$1.14 * \mathcal{T}$
	MLP	0.833 ± 0.01	0.824 ± 0.00	$1.05 * \mathcal{T}$
	KNN	0.907 ± 0.02	0.918 ± 0.02	$0.99 * \mathcal{T}$
	BLS	**0.976 ± 0.01**	**0.972 ± 0.02**	$1.05 * \mathcal{T}$
$\lvert\mathcal{D}\rvert = 1000$	SVM	0.834 ± 0.01	0.844 ± 0.00	$1.00 * \mathcal{T}$
	LR	0.975 ± 0.00	0.974 ± 0.00	$1.21 * \mathcal{T}$
	MLP	0.824 ± 0.00	0.823 ± 0.01	$1.06 * \mathcal{T}$
	KNN	0.953 ± 0.01	0.968 ± 0.00	$1.10 * \mathcal{T}$
	BLS	**0.974 ± 0.00**	**0.974 ± 0.01**	$1.01 * \mathcal{T}$
$\lvert\mathcal{D}\rvert = 2000$	SVM	0.834 ± 0.01	0.845 ± 0.00	$1.00 * \mathcal{T}$
	LR	0.974 ± 0.00	0.976 ± 0.01	$1.23 * \mathcal{T}$
	MLP	0.814 ± 0.02	0.807 ± 0.02	$1.09 * \mathcal{T}$
	KNN	0.966 ± 0.01	**0.979 ± 0.01**	$1.15 * \mathcal{T}$
	BLS	**0.975 ± 0.01**	0.976 ± 0.01	$0.98 * \mathcal{T}$

Notes: The consumed time for different classifiers in such 3 scenarios is represented by the time consumption corresponding to the SVM (denoted as $\mathcal{T}$).

Furthermore, the performance of MLP is not satisfactory due to the small number of initial labeled samples that lead to its underfitting. The main idea of LR is to reduce the weight of samples significantly far away from the classification plane and relatively increase the importance of the samples most relevant to the classification through nonlinear mapping. In this case, LR can get better performance relative stably with the retraining procedure to update the model. However, the effects of the support-vector-based methods, represented by SVM, only depend on the samples classified as support vectors in the current environment. Therefore, such methods cannot perform efficiently in nonstationary environments since the selection of support vectors may need to change constantly. Concerning the KNN model, its performance improves with the increase in $|\mathcal{D}|$ due to the information that can be exploited becoming richer. However, the cost of model training will also increase. Moreover, as mentioned in [8], the performance is not confirmed due to the concept drift in RTSA problems. Therefore, the design of the incremental update process is critical for solving RTSA problems.

2) Experiment II: In this subexperiment, three advanced online learning approaches are considered to show the advantages of BLS in handling online learning tasks in nonstationary environments, which are.

1) *Online sequential extreme learning machine (OSELM) model*: a variant of the extreme learning machine that can update the model incrementally [24].

[1] https://broadlearning.ai
[2] https://github.com/leferrad/pyoselm

TABLE II
DETAILED RESULTS WITH THE SETTING OF EXPERIMENT II WHERE $\mathcal{B} = 10\%|\mathcal{D}|$

Settings	$\|\mathcal{D}\| = 500$				$\|\mathcal{D}\| = 1000$			
	$\overline{\mathcal{A}} \pm \sigma$	$\overline{\mathcal{P}} \pm \sigma$	*Type*	*Counter*	$\overline{\mathcal{A}} \pm \sigma$	$\overline{\mathcal{P}} \pm \sigma$	*Type*	*Counter*
Random Strategy + OSELM	0.823 ± 0.00	0.777 ± 0.07	✓✓ ✓✗ ✗✓ ✗✗	126 19 1039 316	0.839 ± 0.01	0.826 ± 0.04	✓✓ ✓✗ ✗✓ ✗✗	200 40 2279 481
Random Strategy + ARF	0.892 ± 0.01	0.917 ± 0.04	✓✓ ✓✗ ✗✓ ✗✗	140 0 1236 124	0.944 ± 0.02	0.965 ± 0.01	✓✓ ✓✗ ✗✓ ✗✗	224 3 2672 101
Random Strategy + OB	0.877 ± 0.04	0.925 ± 0.04	✓✓ ✓✗ ✗✓ ✗✗	143 3 1245 109	0.912 ± 0.02	0.936 ± 0.02	✓✓ ✓✗ ✗✓ ✗✗	282 8 2527 183
Random Strategy + BLS	**0.953 ± 0.02**	**0.951 ± 0.03**	✓✓ ✓✗ ✗✓ ✗✗	**143** **2** **1284** **71**	**0.953 ± 0.01**	**0.967 ± 0.01**	✓✓ ✓✗ ✗✓ ✗✗	**230** **6** **2671** **93**

Notes: The accuracy of the results is related to the number of decimal digits reserved and the specific test periods.
✓✓: The total number of instances that are predicted correctly with updates; ✓✗: The total number of instances that are mispredicted with updates
✗✓: The total number of instances that are predicted correctly without updates; ✗✗: The number of instances that are mispredicted without updates;

2) *ARF model*: a variant of random forest that can update the model incrementally with the ADWIN technique [28].
3) *OB model*: a variant of the bagging technique that can update the model incrementally with the ADWIN technique [26].

All of them perform the update procedure incrementally. To avoid the influence of the query strategy, we use the random strategy uniformly in this subexperiment. Specifically, for any instance x_t, it is randomly annotated with a certain probability. It is worth noting that each judgment is independent. The probability of being selected depends on the annotation capabilities of submersibles. Similar to *Experiment I*, only 12 samples are required to get the pretrained model. The overall performance of these approaches under three random tests, namely, $\overline{\mathcal{A}}$ and $\overline{\mathcal{P}}$, is recorded. Notably, the number of hidden nodes of OSELM is set as 12 since only 12 samples can be considered to obtain the pretrained OSELM [24]. The default hyperparameters of the ARF and OB models are used since the impact of hyperparameters on the performance of these advanced algorithms is relatively limited. For more details, refer to [45]. Two kinds of data stream ($|\mathcal{D}| = 500$ and $|\mathcal{D}| = 1000$) are considered. We have $\mathcal{B} = 5\%|\mathcal{D}|$, which means that only up to 150 and 300 samples can be annotated, respectively.

The detailed results are displayed in Table II. In most cases of the RTSA tasks, BLS and ARF outperform OB and OSELM with the random strategy. Notably, the total number of instances that are mispredicted with updates decreases significantly under conditions using the ARF, OB, and BLS models. The main reason is that OB and ARF use some methods (such as the ADWIN technique) to detect drift. Such ensemble models can be adjusted accordingly [8]. Furthermore, as shown in (20), the incremental update process of BLS is based on the *Greville theorem*. Therefore, it is less affected by the difference in the underlying distributions concerning the data to a certain extent. Moreover, it can be seen that the performance of ARF is generally better than that of OB and OSELM. As a result, the design of the classifier models for solving real-life RTSA tasks is crucial and should be considered carefully.

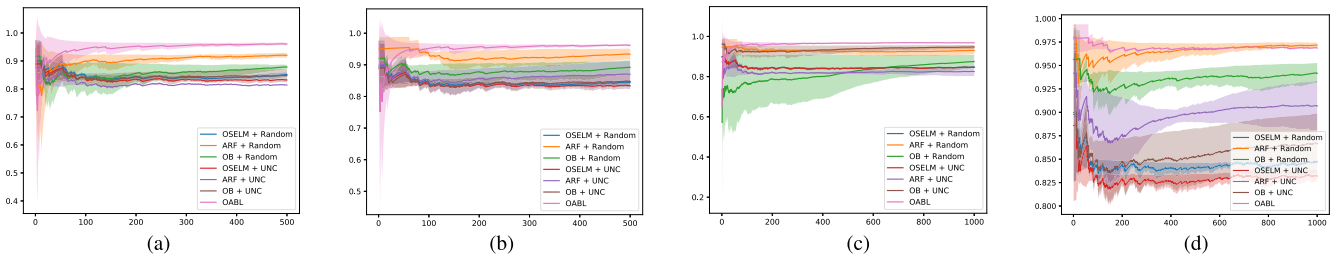
3) Experiment III: This subexperiment considers the settings for OSELM, ARF, and OB with different query strategies. The proposed OABL is also used to show its superiority. Two typical query strategies, *random strategy* and *uncertainty strategy*, are introduced. In detail, the main idea of the uncertainty strategy is to annotate x_t if its difference in the belief degrees exceeds a specific threshold (set to 0.20 in this experiment). Three kinds of annotation budgets are considered: $5\%|\mathcal{D}|$, $10\%|\mathcal{D}|$, and $20\%|\mathcal{D}|$. Similarly, only 12 samples are used to obtain the pretrained model. The ϵ of OABL is set as 0.30 uniformly.

The overall performances are shown in Table III. Moreover, the learning curves for different indicators are shown in Figs. 4 and 5. The OABL outperforms the existing representative methods in almost all the scenarios, illustrating its effectiveness and practicality for solving RTSA problems. Specifically, the effect of the uncertainty strategy may not be as effective as the random strategy. The main reason is that the former relies too much on the belief degrees of prediction. The proposed DAQS can effectively reduce the number of queries required by BLS to ensure the accuracy of the model. In addition, as shown in (12) to (14), it prefers to select potential minority

TABLE III
PERFORMANCES WITH THE SETTING OF EXPERIMENT III

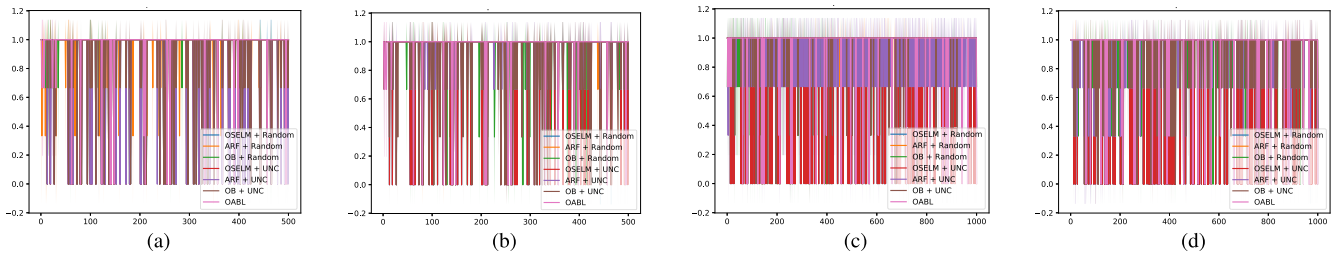
Dataset	*Strategy*	*Model*	$\mathcal{B}$ = 5%$\lvert\mathcal{D}\rvert$		$\mathcal{B}$ = 10%$\lvert\mathcal{D}\rvert$		$\mathcal{B}$ = 20%$\lvert\mathcal{D}\rvert$	
			$\overline{\mathcal{A}} \pm \sigma$	$\overline{\mathcal{P}} \pm \sigma$	$\overline{\mathcal{A}} \pm \sigma$	$\overline{\mathcal{P}} \pm \sigma$	$\overline{\mathcal{A}} \pm \sigma$	$\overline{\mathcal{P}} \pm \sigma$
$\lvert\mathcal{D}\rvert$ = 250	*Random Strategy*	*OSELM*	0.836 ± 0.01	0.865 ± 0.06	0.751 ± 0.05	0.712 ± 0.17	0.857 ± 0.00	0.855 ± 0.01
		ARF	0.949 ± 0.02	0.935 ± 0.05	0.963 ± 0.01	0.943 ± 0.04	0.945 ± 0.01	0.961 ± 0.02
		OB	0.880 ± 0.01	0.909 ± 0.04	0.856 ± 0.01	0.875 ± 0.03	0.881 ± 0.02	0.908 ± 0.04
	Uncertainty Strategy	*OSELM*	0.830 ± 0.01	0.824 ± 0.00	0.710 ± 0.06	0.785 ± 0.08	0.838 ± 0.00	0.840 ± 0.01
		ARF	0.965 ± 0.01	0.951 ± 0.02	0.830 ± 0.01	0.824 ± 0.00	0.878 ± 0.02	0.916 ± 0.06
		OB	0.813 ± 0.02	0.805 ± 0.05	0.840 ± 0.00	0.847 ± 0.02	0.868 ± 0.02	0.908 ± 0.06
	OABL*		**0.969 ± 0.01**	**0.964 ± 0.01**	**0.973 ± 0.00**	**0.969 ± 0.00**	**0.971 ± 0.01**	**0.967 ± 0.00**
$\lvert\mathcal{D}\rvert$ = 500	*Random Strategy*	*OSELM*	0.847 ± 0.01	0.857 ± 0.01	0.823 ± 0.00	0.777 ± 0.07	0.845 ± 0.01	0.847 ± 0.02
		ARF	0.902 ± 0.02	0.930 ± 0.02	0.892 ± 0.01	0.917 ± 0.04	0.929 ± 0.01	0.951 ± 0.03
		OB	0.857 ± 0.03	0.886 ± 0.01	0.877 ± 0.04	0.925 ± 0.04	0.881 ± 0.02	0.916 ± 0.04
	Uncertainty Strategy	*OSELM*	0.837 ± 0.01	0.832 ± 0.00	0.825 ± 0.00	0.826 ± 0.01	0.839 ± 0.01	0.834 ± 0.00
		ARF	0.820 ± 0.01	0.814 ± 0.00	0.895 ± 0.01	0.906 ± 0.00	0.862 ± 0.03	0.913 ± 0.07
		OB	0.845 ± 0.02	0.872 ± 0.04	0.854 ± 0.02	0.891 ± 0.05	0.845 ± 0.02	0.873 ± 0.04
	OABL*		**0.946 ± 0.02**	**0.965 ± 0.01**	**0.971 ± 0.00**	**0.968 ± 0.00**	**0.951 ± 0.02**	**0.965 ± 0.00**
$\lvert\mathcal{D}\rvert$ = 1000	*Random Strategy*	*OSELM*	0.846 ± 0.00	0.847 ± 0.00	0.839 ± 0.01	0.826 ± 0.04	0.844 ± 0.00	0.848 ± 0.00
		ARF	0.929 ± 0.01	0.950 ± 0.03	0.944 ± 0.02	0.965 ± 0.01	0.965 ± 0.01	**0.974 ± 0.00**
		OB	0.817 ± 0.09	0.909 ± 0.06	0.912 ± 0.02	0.936 ± 0.02	0.936 ± 0.01	0.955 ± 0.02
	Uncertainty Strategy	*OSELM*	0.845 ± 0.00	0.846 ± 0.00	0.829 ± 0.00	0.835 ± 0.01	0.830 ± 0.00	0.838 ± 0.01
		ARF	0.823 ± 0.01	0.855 ± 0.08	0.859 ± 0.02	0.894 ± 0.04	0.897 ± 0.00	0.895 ± 0.05
		OB	0.936 ± 0.02	0.959 ± 0.02	0.844 ± 0.02	0.875 ± 0.04	0.856 ± 0.03	0.900 ± 0.05
	OABL*		**0.963 ± 0.01**	**0.968 ± 0.00**	**0.973 ± 0.00**	**0.972 ± 0.00**	**0.969 ± 0.00**	0.969 ± 0.00

Notes: The accuracy of the results is related to the number of decimal digits reserved and the specific test time periods.
*: The proposed OABL approach.



(a) (b) (c) (d)

Fig. 4. Learning curves of $\mathcal{A}$ for different approaches with different settings in Experiment III. (a) $|\mathcal{D}| = 500$ and $|\mathcal{B}| = 5\%|\mathcal{D}|$. (b) $|\mathcal{D}| = 500$ and $|\mathcal{B}| = 20\%|\mathcal{D}|$. (c) $|\mathcal{D}| = 1000$ and $|\mathcal{B}| = 5\%|\mathcal{D}|$. (d) $|\mathcal{D}| = 1000$ and $|\mathcal{B}| = 20\%|\mathcal{D}|$.



(a) (b) (c) (d)

Fig. 5. Learning curves of $\mathcal{P}$ for different approaches with different settings in Experiment III. (a) $|\mathcal{D}| = 500$ and $|\mathcal{B}| = 5\%|\mathcal{D}|$. (b) $|\mathcal{D}| = 500$ and $|\mathcal{B}| = 20\%|\mathcal{D}|$. (c) $|\mathcal{D}| = 1000$ and $|\mathcal{B}| = 5\%|\mathcal{D}|$. (d) $|\mathcal{D}| = 1000$ and $|\mathcal{B}| = 20\%|\mathcal{D}|$.

class samples and high-uncertainty samples, which is essential to update the model in the imbalanced data stream.

Since it inherits from the RVFLNN structure, OABL can only rely on a small number of samples for model training. As shown in Table III, it has a relatively stable and excellent performance in this case. Considering the characteristics of the RTSA tasks, the underlying distribution of earlier historical data and currently collected data may be quite different.

Therefore, using fewer samples for initial model training has high practical significance. Submersibles can restart the assessment model in some unique scenarios to avoid the negative impact of the nonstationary environment as much as possible. Hence, the proposed OABL is more suitable for handling RTSA tasks.

In general, it is helpful to avoid the budget being exhausted at the early stage with the utilization of the Bernoulli variables directly. However, a threshold to assist is more recommended if the imbalance ratio is low. The smaller the ϵ, the more the number of queries required from experts. Although the accuracy of prediction would be higher with the increased number of queries, the need for the annotation budget would also increase. Hence, the selection of ϵ to trade off the preferences for prediction accuracy and the annotation budget can be changed based on specific nonstationary environments.

IV. Conclusion

In this article, a novel OABL approach has been proposed to deal with the RTSA tasks of complex dynamic systems. To the best of our knowledge, it is the first study to reveal and verify the effectiveness of BLS in the framework of online active learning. A reasonable DAQS has also been proposed to deal with the issues of class imbalance and limited annotation budget. Several experiments have been used to demonstrate the practicality of the proposal using the realistic data of the JiaoLong DSMS.

In addition, improving the structure concerning BLS to adapt concept drift is a significant issue worth future exploration. The theoretical analysis of query strategies should also be further studied.

Acknowledgment

The authors would like to thank the reviewers' suggestions regarding the experimental analysis and expression details and the editor's encouragement.

References

[1] W. Cui, "Development of the Jiaolong deep manned submersible," *Mar. Technol. Soc. J.*, vol. 47, no. 3, pp. 37–54, May 2013.

[2] H. Zhang, C. Zhang, J. Dong, and K. Peng, "A new key performance indicator oriented industrial process monitoring and operating performance assessment method based on improved Hessian locally linear embedding," *Int. J. Syst. Sci.*, pp. 1–18, Jul. 2022.

[3] Z. Liu, Y. Deng, Y. Zhang, Z. Ding, and X. He, "Safety assessment of dynamic systems: An evidential group interaction-based fusion design," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–14, 2021.

[4] H. Chen, B. Jiang, S. X. Ding, and B. Huang, "Data-driven fault diagnosis for traction systems in high-speed trains: A survey, challenges, and perspectives," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 3, pp. 1700–1716, Mar. 2022.

[5] X. Li, J. Xu, Z. Chen, S. Xu, and K. Liu, "Real-time fault diagnosis of pulse rectifier in traction system based on structural model," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 3, pp. 2130–2143, Mar. 2022.

[6] F. M. Shakiba, M. Shojaee, S. M. Azizi, and M. Zhou, "Generalized fault diagnosis method of transmission lines using transfer learning technique," *Neurocomputing*, vol. 500, pp. 556–566, Aug. 2022.

[7] H. Chen and B. Jiang, "A review of fault detection and diagnosis for the traction system in high-speed trains," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 2, pp. 450–465, Feb. 2020.

[8] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, "Learning under concept drift: A review," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 12, pp. 2346–2363, Dec. 2019.

[9] M. Bahri, A. Bifet, J. Gama, H. M. Gomes, and S. Maniu, "Data stream analysis: Foundations, major tasks and tools," *WIREs Data Mining Knowl. Discovery*, vol. 11, no. 3, p. e1405, May 2021.

[10] S. Wang, L. L. Minku, and X. Yao, "A systematic study of online class imbalance learning with concept drift," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 10, pp. 4802–4821, Oct. 2018.

[11] K. Malialis, C. G. Panayiotou, and M. M. Polycarpou, "Online learning with adaptive rebalancing in nonstationary environments," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 10, pp. 4445–4459, Oct. 2021.

[12] H. Chen, Z. Liu, C. Alippi, B. Huang, and D. Liu, "Explainable intelligent fault diagnosis for nonlinear dynamic systems: From unsupervised to supervised learning," *TechRxiv*, Jun. 2022, doi: 10.36227/techrxiv.19101512.v1.

[13] C. OReilly, A. Gluhak, M. A. Imran, and S. Rajasegarar, "Anomaly detection in wireless sensor networks in a non-stationary environment," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 3, pp. 1413–1432, 3rd Quart., 2014.

[14] X. Zhao et al., "A safety framework for critical systems utilising deep neural networks," in *Proc. Int. Conf. Comput. Saf., Rel., Secur.* Cham, Switzerland: Springer, 2020, pp. 244–259.

[15] A. Sedaghati, N. Pariz, M. Siahi, and R. Barzamini, "A new fractional-order developed type-2 fuzzy control for a class of nonlinear systems," *Int. J. Syst. Sci.*, pp. 1–19, Jan. 2021.

[16] Z. Hu and P. Jiang, "An imbalance modified deep neural network with dynamical incremental learning for chemical fault diagnosis," *IEEE Trans. Ind. Electron.*, vol. 66, no. 1, pp. 540–550, Jan. 2019.

[17] Y. Freund and R. E. Schapire, "Large margin classification using the perceptron algorithm," *Mach. Learn.*, vol. 37, no. 3, pp. 277–296, 1999.

[18] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer, "Online passive-aggressive algorithms," *J. Mach. Learn. Res.*, vol. 7, pp. 551–585, Dec. 2006.

[19] M. Dredze, K. Crammer, and F. Pereira, "Confidence-weighted linear classification," in *Proc. 25th Int. Conf. Mach. Learn. (ICML)*, vol. 307, W. W. Cohen, A. McCallum, and S. T. Roweis, Eds., Helsinki, Finland, 2008, pp. 264–271.

[20] K. Crammer, A. Kulesza, and M. Dredze, "Adaptive regularization of weight vectors," *Mach. Learn.*, vol. 91, no. 2, pp. 155–187, May 2013.

[21] L. L. Minku, A. P. White, and X. Yao, "The impact of diversity on online ensemble learning in the presence of concept drift," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 5, pp. 730–742, May 2010.

[22] J. Xuan, J. Lu, and G. Zhang, "Bayesian nonparametric unsupervised concept drift detection for data stream mining," *ACM Trans. Intell. Syst. Technol.*, vol. 12, no. 1, pp. 1–22, Feb. 2021.

[23] P. Ksieniewicz, "The prior probability in the batch classification of imbalanced data streams," *Neurocomputing*, vol. 452, pp. 309–316, Sep. 2021.

[24] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, nos. 1–3, pp. 489–501, 2006.

[25] N. C. Oza and S. Russell, "Experimental comparisons of online and batch versions of bagging and boosting," in *Proc. 7th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, D. Lee, M. Schkolnick, F. J. Provost, and R. Srikant, Eds., San Francisco, CA, USA, 2001, pp. 359–364.

[26] A. Bifet, G. Holmes, and B. Pfahringer, "Leveraging bagging for evolving data streams," in *Proc. Eur. Conf. Mach. Learn. Knowl. Discovery Databases (ECML PKDD)*, in Lecture Notes in Computer Science, vol. 6321, J. L. Balcázar, F. Bonchi, A. Gionis, and M. Sebag, Eds., Barcelona, Spain. Cham, Switzerland: Springer, 2010, pp. 135–150.

[27] A. Bifet and R. Gavaldà, "Learning from time-changing data with adaptive windowing," in *Proc. 7th SIAM Int. Conf. Data Mining*, Minneapolis, MI, USA. Philadelphia, PA, USA: SIAM, 2007, pp. 443–448.

[28] H. M. Gomes et al., "Adaptive random forests for evolving data stream classification," *Mach. Learn.*, vol. 106, nos. 9–10, pp. 1469–1495, 2017.

[29] B. Settles, "Active learning literature survey," Dept. Comput. Sci., Univ. Wisconsin-Madison, Tech. Rep. TR-1648, 2009.

[30] S. Liu et al., "Online active learning for drifting data streams," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jul. 21, 2021, doi: 10.1109/TNNLS.2021.3091681.

[31] P. Zhao and S. C. H. Hoi, "Cost-sensitive online active learning with application to malicious URL detection," in *Proc. 19th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Chicago, IL, USA, Aug. 2013, pp. 919–927.

[32] Y. Zhang et al., "Online adaptive asymmetric active learning with limited budgets," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 6, pp. 2680–2692, Jun. 2021.
[33] C. L. P. Chen and Z. L. Liu, "Broad learning system: An effective and efficient incremental learning system without the need for deep architecture," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 1, pp. 10–24, Jan. 2018.
[34] Y.-H. Pao, S. M. Phillips, and D. J. Sobajic, "Neural-net computing and the intelligent control of systems," *Int. J. Control*, vol. 56, no. 2, pp. 263–289, 1992.
[35] C. L. P. Chen, Z. Liu, and S. Feng, "Universal approximation capability of broad learning system and its structural variations," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 4, pp. 1191–1204, Apr. 2019.
[36] X. Gong, T. Zhang, C. L. P. Chen, and Z. Liu, "Research review for broad learning system: Algorithms, theory, and applications," *IEEE Trans. Cybern.*, vol. 52, no. 9, pp. 8922–8950, Sep. 2021.
[37] X. Pu and C. Li, "Online semisupervised broad learning system for industrial fault diagnosis," *IEEE Trans. Ind. Informat.*, vol. 17, no. 10, pp. 6644–6654, Oct. 2021.
[38] Z. Liu, J. Zhang, X. He, Q. Zhang, G. Sun, and D. Zhou, "Fault diagnosis of rotating machinery with limited expert interaction: A multicriteria active learning approach based on broad learning system," *IEEE Trans. Control Syst. Technol.*, early access, Aug. 29, 2022, doi: 10.1109/TCST.2022.3200214.
[39] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 42, no. 1, pp. 80–86, 1970.
[40] N. Cesa-Bianchi, C. Gentile, and L. Zaniboni, "Worst-case analysis of selective sampling for linear classification," *J. Mach. Learn. Res.*, vol. 7, pp. 1205–1230, Jul. 2006.
[41] X. Zhang, T. Yang, and P. Srinivasan, "Online asymmetric active learning with imbalanced data," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, B. Krishnapuram, M. Shah, A. J. Smola, C. C. Aggarwal, D. Shen, and R. Rastogi, Eds., San Francisco, CA, USA, Aug. 2016, pp. 2055–2064.
[42] Y. Sun, K. Tang, Z. Zhu, and X. Yao, "Concept drift adaptation by exploiting historical knowledge," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 10, pp. 4822–4832, Oct. 2018.
[43] Y.-P. Tang, G.-X. Li, and S.-J. Huang, "ALiPy: Active learning in Python," 2019, *arXiv:1901.03802*.
[44] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Jul. 2011.
[45] J. Montiel, J. Read, and A. Bifet, "Scikit-multiflow: A multi-output streaming framework," *J. Mach. Learn. Res.*, vol. 19, no. 1, pp. 2914–2915, 2018.
[46] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.
[47] S. Dreiseitl and L. Ohno-Machado, "Logistic regression and artificial neural network classification models: A methodology review," *J. Biomed. Inform.*, vol. 35, nos. 5–6, pp. 352–359, 2002.
[48] M. W. Gardner and S. R. Dorling, "Artificial neural networks (the multilayer perceptron)—A review of applications in the atmospheric sciences," *Atmos. Environ.*, vol. 32, nos. 14–15, pp. 2627–2636, Aug. 1998.
[49] L. E. Peterson, "K-nearest neighbor," *Scholarpedia*, vol. 4, no. 2, p. 1883, 2009.

Zeyi Liu received the B.E. degree from the School of Computer and Information Science, Southwest University, Chongqing, China, in 2021. He is currently pursuing the Ph.D. degree with the Department of Automation, Tsinghua University, Beijing, China.

His research interests include machine learning, information fusion, fuzzy logic, and their applications such as safety assessment and fault diagnosis.

Dr. Liu is a member of the Chinese Association of Automation (CAA) and the Chinese Association for Artificial Intelligence (CAAI). He severs as a reviewer of refereed journals and conferences such as the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, and the IEEE TRANSACTIONS ON CYBERNETICS.

Yi Zhang received the B.E. and master's degrees in ship and ocean engineering from Harbin Engineering University, Harbin, China, in 2012 and 2014, respectively, where she is pursuing the Ph.D. degree.

She is now a pilot of deep-sea manned submersible with the National Deep Sea Center, Qingdao, China. She used to dive the JiaoLong manned submersible in the Northwest Pacific Ocean, the Southwest Indian Ocean, the South China Sea, and other waters. Her largest recorded dive was 6800 m. Her research interests include damage analysis and health monitoring of the structure system of deep-sea manned submersible.

Zhongjun Ding received the B.E. degree in industrial electrical automation from the Wuhan University of Technology, Wuhan, China, in 1996, and the Ph.D. degree in marine geophysics from the Ocean University of China, Qingdao, China, in 2013.

He is now a Professor-Level Senior Engineer and the Deputy Chief Engineer with the National Deep Sea Center, Qingdao. As the Assistant and the Deputy General Commander of JiaoLong's experimental application voyage, he completed the on-site technical support work of China Ocean voyage 31, 35, and 37 continuously. He has been in charge of many major projects, such as the National Key Research and Development Program of China and National Marine Public Welfare Projects. He has authored more than 30 articles and more than 10 invention patents. His main research interests include manned submersible operation support and manned deep submersible exploration technology.

Xiao He (Senior Member, IEEE) received the B.E. degree in information technology from the Beijing Institute of Technology, Beijing, China, in 2004, and the Ph.D. degree in control science and engineering from Tsinghua University, Beijing, in 2010.

He is currently a tenured Professor with the Department of Automation, Tsinghua University. He has authored more than 100 articles in refereed international journals. His research interests include fault diagnosis and fault-tolerant control, networked systems, cyber-physical systems, and their applications.

Dr. He is now a Full Member of Sigma Xi, the Scientific Research Society, and a Senior Member of the Chinese Association of Automation (CAA). He is an Associate Editor of the *Control Engineering Practice*.