Regular Paper

Low-Cost and Steady On-Line Retraining of MLP with Guide Data

Yuya Kaneda[1,2,a)] Qiangfu Zhao[1,b)] Yong Liu[1,c)]

Received: December 26, 2016, Accepted: May 16, 2017

Abstract: The decision boundary making (DBM) algorithm was proposed by us to induce compact and high performance machine learning models for implementation in portable/wearable computing devices. To upgrade performance of DBM-initialized models, we may use all observed data to retrain the model, but the computational cost is high. To reduce the cost, we may use the newly observed datum only, but this often degrades the performance of the model. To solve the problem, we propose on-line training algorithm with guide data (OLTA-GD) in this paper. OLTA-GD updates the model using only a few guide data along with the newest datum. The guide data are selected from all available data. Here, guide data selection is a key point. For this purpose, this paper investigates two methods. The first method is random selection, and the second one is cluster center based. In the second method, the cluster centers are obtained using k-means algorithm. Experimental results show that, OLTA-GD can upgrade the models more steadily than backpropagation (BP) algorithm, and the first selection method is better. For the guide data, around 5 data are usually enough to upgrade the performance steadily, and thus the computational cost is basically not increased compared with the BP.

Keywords: decision boundary making, multilayer perceptron, on-line training, guide data

1. Introduction

In recent years, portable/wearable computing devices (P/WCDs) (e.g., smart phones, smart tablets, and smart watches) are becoming more and more popular. Users can carry P/WCDs around, and use various applications for different purposes in their daily lives. In our research, we aim to assist the users using aware agents (A-agents) embedded in the P/WCD applications to improve their quality of lives. The A-agents are machine learning models that can be aware of the information related to situations, locations, intentions, health, etc., and provide proper hints for the user to make decisions. To embed various A-agents in a P/WCD, the implementation costs of the A-agents must be low because the computing resources are limited. For example, the CPU can be slower, the memory space can be smaller, and the battery can be weaker, compared with desk-top machines.

In our early study, we proposed the decision boundary making (DBM) algorithm [8] for inducing compact and high accuracy machine learning models. High accuracy models generally require high computational costs while low-cost models usually do not have high accuracy. To resolve this dilemma, we focused on the decision boundary (DB) of the given problem. The basic idea of the DBM algorithm is to reconstruct the DB defined by a high accuracy model using a low-cost model. This is realized by generating a new data set first to fit the DB defined by the high performance model, and then inducing a low-cost model using the new data. In our study, we use a support vector machine (SVM) [19] to find the DB, and a single hidden layer multilayer perceptron (MLP) [7] to reconstruct the DB. Using SVM, it is possible to approximate the true DB using the support vectors (SVs). However, the number of SVs can be very large if the training set is large. Using MLP, we can control the size of the model by fixing the number of hidden neurons. Therefore, if the MLP can approximate the DB defined by an SVM, we can reduce the model size and preserve the performance. This has been confirmed by our earlier studies [8].

Several techniques have been proposed to reduce the implementation costs of machine learning models. A typical method for reducing the cost of an SVM is to reduce the training set size. It is known that the size of an SVM, or the number of SVs, is usually proportional to the number of training data [18]. Reduced SVM (RSVM) has been proposed based on the training data reduction [10], [11]. Note that in using RSVM, some important SVs might be lost in the data reduction step. Therefore, the performance of the RSVM is theoretically upper-bounded by the SVM induced from the original training set. Another way for finding compact SVM is to reduce the number of SVs directly. Dong et al. proposed a new kernel function for the SVM [5], and reduced the number of SVs with the kernel function. From the experimental results, the method could decrease the number of SVs. However, how to control the model size is still a problem. Geebelen et al. proposed a method called smoothed separable case approximation (SSCA) to reduce the SVs by removing or re-

[1] Department of Computer Science and Engineering, The University of Aizu, Aizu-Wakamatsu, Fukushima 965–8580, Japan
[2] Department of Automated Driving, Integrated Control System Development Division, Honda R&D Co., Ltd. Automobile R&D Center, Tochigi 321–3321, Japan
a) Yuya_01_Kaneda@n.t.rd.honda.co.jp
b) qf-zhao@u-aizu.ac.jp
c) yliu@u-aizu.ac.jp

labeling outliers [6]. The outliers are noisy data for training good models. However, if there is no outlier data in the given training set, the method cannot reduce the SVs. Currently, a chip-based method was proposed to implement large-scale neural networks in a special-purpose P/WCD [2]. Unfortunately, this method cannot be used easily in existing P/WCDs.

In this study, we focus on on-line retraining of an existing machine learning model. The main purpose is to upgrade the model performance steadily by incorporating new information obtained from users. The main issue here, again, is to implement the retraining process with low-cost, because our targets are P/WCD users. We consider retraining of an existing model here rather than starting from scratch because a relatively weak initial model can often be designed off-line using data available from the Internet or from other resources. A compact initial model can be obtained by using the DBM algorithm. The main point here is to propose a method for upgrading the model performance steadily and efficiently.

Actually, there is another aspect of the on-line retraining. For many applications, the DBs of different users are usually different. One way to accommodate the difference is to design a model fitting to each user. Through off-line training, we can obtain a "common sense" model, and through on-line retraining, we can customize the model to each user. This is a kind of transfer learning [20]. The base knowledge or "common sense" is first extracted from the data of many users, and is then transferred to adapt each user.

There are some on-line training algorithms that have low computational costs. K. Crammer et al. proposed online passive-aggressive algorithm [4]. This is a family of on-line training algorithms for updating the hyperplain. Although the algorithms can be extended to non-linear cases, the performance may not be improved steadily through on-line training. Also, too many data are used for the non-linear cases, therefore it is difficult to implement in P/WCD applications and learn on-line in a P/WCD environment. M. Cillins used on-line perceptron algorithm to solve a highly non-linear tagging problem [3]. It seems that the main reason of success is using Viterbi decoding, which can absorb the non-linearity effectively. However, it is not sure that this method is also good for solving other problems. S. Shai et al. proposed a primal estimated sub-gradient solver (Pegasos) for SVM [17]. They used a simple stochastic sub-gradient descent algorithm, then the computational cost for the training is low. However, this also needs many data for non-linear problems, therefore the computational cost of classification may be high for P/WCDs.

To upgrade an existing model steadily, we propose an on-line training algorithm with guide data (OLTA-GD). For on-line retraining, we may use the newest datum only to update the model. This way, we can reduce the computational cost, but the performance of the model can be greatly degraded when some "novel" data are observed. On the other hand, we may use all data observed so far to retrain the model. This way, we can preserve the model performance, but the cost will be very large. To make a compromise, we may use the mini-batch approach, which uses a block of some new data to retrain the model. However, in case we want to incorporate the new information in real time, the mini-batch approach cannot be used. The OLTA-GD proposed in this study retrains the model using several already observed data along with the newly observed one. Provided that the guide data are selected properly, we can upgrade the model both efficiently and effectively.

There are many methods for guide data selection. In this paper, we investigate two selection methods. The first method selects the guide data randomly from a set containing all observed data. The second method is cluster center based selection. In this method, the cluster centers are found by using k-means [13]. The basic idea of the cluster center based method is to select a guide datum from each cluster. Using OLTA-GD, model retraining will not depend too much on a single datum.

The structure of this paper is as follows. Section 2 introduces the DBM algorithm briefly. Section 3 explains the OLTA-GD with guide data generation methods. Section 4 provides experimental results on several public databases. Finally, Section 5 shows the conclusions and some topics for future work.

2. Decision Boundary Making Algorithm

The DBM algorithm was proposed to induce compact and high performance machine learning models for P/WCDs. The main idea of the DBM algorithm is to emulate the DB of a high performance machine learning model using a low-cost machine learning model. We employ an SVM model for the high performance model, and use a single hidden layer MLP model for the low-cost model. To reconstruct the DB, we generate a new training set approximated to the DB of the SVM, and then obtain an MLP model based on the new training set. The training and classification phases are shown in **Fig. 1**. In the classification phase, we use only the MLP model, so we do not need to keep the SVM model after training.

We set some parameters for the DBM algorithm to generate better new training sets. To add data close to the DB, we use support vectors (SVs) of the SVM model. We generate N_{DBM} data near each SV, and a parameter ϵ controls the neighborhood area. In other words, N_{DBM} data are generated in the ϵ-neighborhood of each SV. However, if we add these N_{DBM} data directly into the new training set without conditions, negative effect data for obtaining high performance models might be included in the set. To remove the negative effect data, we set two conditions given in

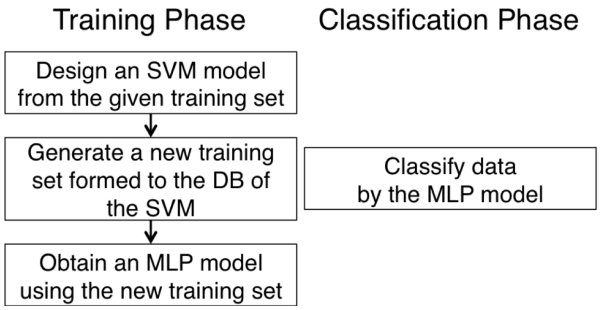

Fig. 1 Brief flows of the DBM algorithm. There are three steps in the training phase, and one step in the classification phase. The DBM algorithm first obtains an SVM model, generates a new training set approximated to the DB of the SVM, and then designs an MLP model based on the new training set in the training phase. Data are classified using only the MLP model.

Eqs. (1) and (2), where $f_{SVM}(X)$ is the output value of the SVM model for a datum X, δ_{DB} is a parameter of the DBM algorithm to control a negative effect data area, and X_{SV} is the SV corresponding to the generated datum X. If a generated datum X satisfies Eq. (1) or Eq. (2), the datum X is not added into the new training set. The new training set finally consists of the newly generated data, the SVs of the SVM, and the given training data. However, the given training set may contain some noisy data, and the obtained model may have lower performance. We call these data outliers. To reduce the negative effect of the outliers, we remove them from the given training set so that they will not appear in the new training set. The outlier definition is shown in Eq. (3), where $y \in \{-1, +1\}$ is the class label of a datum X, and $\delta_{outlier}$ is a given parameter of the DBM algorithm to control the outlier data area. If a datum in the given training data set satisfies Eq. (3), it is not added to the new training set.

$$|f_{SVM}(X)| < \delta_{DB} \tag{1}$$

$$|f_{SVM}(X)| > |f_{SVM}(X_{SV})| \tag{2}$$

$$y \times f_{SVM}(X) < -\delta_{outlier} \tag{3}$$

The detailed training phase of the DBM algorithm in the above discussion is reported in Algorithm 1.

Algorithm 1 Obtaining an MLP model from a given training set Ω by the DBM algorithm

1: Obtain an SVM model based on the given training set Ω
2: Detect noisy data from Ω by using Eq. (3) as $U_{outlier} = \{X_{outlier1}, \ldots, X_{outlierN_{outlier}}\}$ ($N_{outlier}$ is the number of noisy data)
3: Initialize a SV set as $U_{SV} = \{X_{SV,1}, X_{SV,2}, \ldots, X_{SV,N_{SV}}\}$ ($X_{SV,i}$ is the i-th SV of the SVM)
4: Initialize a new training set as $\Omega_{new} = U_{SV} + \Omega - U_{outlier}$
5: **for each** X_{SV} in U_{SV} **do**
6: **for** $i = 1$ to N_{DBM} **do**
7: Create a vector V that each element is random value in $[-\epsilon, +\epsilon]$
8: $X_{new} = X_{SV} + V$
9: **if** $|f_{SVM}(X_{new})| < \delta_{DB}$ or $|f_{SVM}(X_{new})| > |f_{SVM}(X_{SV})|$ **then**
10: Continue
11: **end if**
12: Set the label of X_{new} by $sgn(f_{SVM}(X_{new}))$
13: Add the datum X_{new} into the new training set Ω_{new} $\Omega_{new} = \Omega_{new} + X_{new}$
14: **end for**
15: **end for**
16: Obtain an MLP model using the new training set Ω_{new}

3. On-Line Training Algorithm with Guide Data

To upgrade performance of existing models on P/WCD in real time, the retraining algorithm should be steady and have low calculation cost. If the retraining algorithm is not steady, sometimes the model performance can be extremely decreased, and the reliability of the models will be lower. Moreover, the computational resources of the P/WCD are limited. To retrain the model on P/WCD, we have to save the calculation cost.

To accommodate the problem, we propose OLTA-GD. The OLTA-GD updates a model initialized by the DBM algorithm using an observed datum and some guide data. Note that, the observed datum is a new datum obtained in real time. When an observed datum is received, an average gradient is calculated from the observed datum and some guide data, and the gradient is used for updating model. The update equation for an observed datum $X_{observed}$ is given in Eq. (4), where $\mathbf{W}^{(t)}$ is the weight vector of the MLP model at time t, N_{guide} is the number of guide data, $X_{guide,i}$ is the i-th guide datum, and $g_{MLP}(X, \mathbf{W})$ is the objective function of MLP for a datum X and a weight vector $\mathbf{W}$. To optimize the objective function, we use the backpropagation (BP) algorithm [16] to update the model. If we use many data for the guide data, the influence of the observed datum $X_{observed}$ for updating becomes low and the calculation cost becomes high. However, if we use too less data for the guide data, effectiveness of the guide data is reduced. We have to use proper guide data size for steady and low-cost training.

$$\mathbf{W}^{(t+1)} = \mathbf{W}^{(t)} - \frac{\alpha}{N_{guide}+1}\left(\sum_{i=1}^{N_{guide}} \frac{\partial g_{MLP}(X_{guide,i}, \mathbf{W}^{(t)})}{\partial \mathbf{W}} + \frac{\partial g_{MLP}(X_{observed}, \mathbf{W}^{(t)})}{\partial \mathbf{W}}\right) \tag{4}$$

Another problem is how to define the guide data. In this study, we use all observed data including those generated data in DBM training, and update the candidate set by adding each observed datum. However, if we use the whole set for the guide data, the calculation cost is particularly increased because the new training set has more data than the original training set. To reduce the guide data, we propose two guide data selection methods. These methods are random selection, and cluster center based selection. In the following sub-sections, we introduce each guide data selection method in detail.

3.1 Guide Data Generation Based on Random Selection

The random selection picks up guide data randomly from the candidate set. In this method, the selection process is simple and the cost is low. This method picks up data directly. The number of guide data N_{guide} is a given parameter in the method. After updating the model, this method adds the newly observed datum in the candidate set.

As for the guide data, it is affected by the data distribution of the candidate set. In our proposed method, the initial candidate set is new training data generated by the DBM algorithm. The new training set has many data near the DB, the data distribution is biased to the DB. Therefore, selected guide data set will have many data near the DB.

3.2 Guide Data Generation by Cluster Center Based Selection

This method uses a clustering method for the candidate set. The basic idea is to pick up a datum from each cluster. In this method, the cluster centers are found by k-means algorithm [13]. **Figure 2** shows an example of clustering by k-means algorithm.

Off-line training and on-line training flows are given in **Fig. 3**. In the off-line training phase, the partitioning step is proceeded

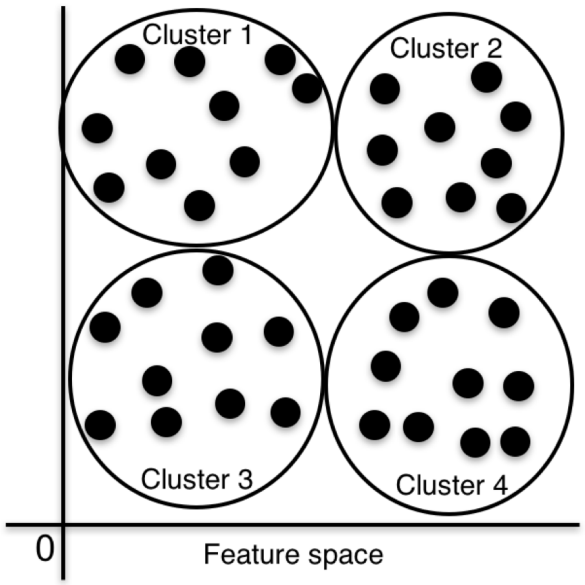

Fig. 2 The figure shows an example of clustered candidate set based on k-means algorithm. To pick up data from each cluster, we get guide data uniformly from the feature space.

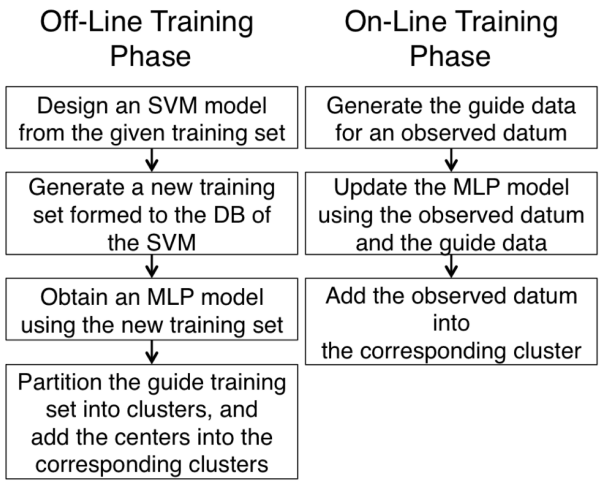

Fig. 3 This figure shows off-line training and on-line training flows of the DBM algorithm for the cluster center based selection method. The off-line training phase is based on the training phase of the original DBM algorithm (see Fig. 1). One step is added into the off-line training phase to separate the candidate set into clusters and add the cluster centers. In the on-line training phase, we generate guide data set for an observed datum at beginning, update the MLP model by using the observed datum and the guide data, and then update the candidate set by adding the observed datum.

at the end for clustering. After obtaining an MLP model, it separates the candidate set by the k-means algorithm, and adds the cluster centers into the corresponding clusters of the candidate set to increase data. And in the on-line training phase, there are mainly three steps. When an observed datum is coming, it first generates guide data set by the cluster center based selection, and updates the MLP model using the guide data set and the observed datum. After that, it adds the observed datum into the corresponding cluster of the candidate set.

As for the data selection, this method selects $k-1$ data for guide data set. It picks up a datum randomly from each cluster, then the number of picked up data becomes k. However, we remove the datum of the p-th cluster from the guide data, where the observed datum $X_{observed}$ belongs in p-th cluster. Average gradients are calculated from the observed datum $X_{observed}$ and the guide data. If we use all the picked up data for the guide data, doubled data in the p-th cluster are included in the average gradients. To make the uniform distribution of the guide data and the observed datum, the p-th cluster datum is not used for updating the model. Therefore, the number of guide data N_{guide} in this method is $k-1$.

As for the random sampling from each cluster, we do not use fixed guide data (e.g., use cluster centers) for each model updating. If we use the same data for each updating, the other data do not have to be kept in on-line training phase, then the storage space can be conserved. However, we will update models many times over on-line training times, it means that the models learn the guide data many times, then the performance will be decreased because the models fit to the guide data too much. To avoid it, we pick up a datum randomly from each cluster in each on-line training.

The detailed flow of the guide data selection using the cluster center based selection is given in Algorithm 2.

Algorithm 2 Obtaining a guide data set U_{guide} having $k-1$ data for the observed datum $X_{observed}$ by the cluster center based selection.

1: Initialize the guide data set U_{guide}
2: Classify the observed datum $X_{observed}$ by k-means, and the index of the cluster is p
3: **for** i=1 to k **do**
4: **if** i=p **then**
5: Continue
6: **end if**
7: Pick up a datum $X_{guide,i}$ randomly from i-th cluster in the candidate set
8: Add the datum $X_{guide,i}$ into the guide data set U_{guide}
9: **end for**

In this method, the data distribution of guide data becomes uniform in the feature space.

3.3 The Calculation Cost

The calculation cost of OLTA-GD is related to the number of guide data N_{guide} and the model size. This updates weights by BP-based method, and the calculation cost for the BP algorithm is $O(N_{ite}N_iN_h)$ or $O(N_{ite}N_hN_o)$, where N_{ite} is the number of training iterations, N_i is the number of input neurons, N_h is the number of hidden neurons, and N_o is the number of output neurons. In this study, N_i is usually bigger than N_o, thus the training cost for the BP is $O(N_{ite}N_iN_h)$. For the cost of OLTA-GD, we use average gradients of each observed datum and some guide data, and update models only one time per iteration for each observed datum. Therefore, the total computational cost is $O(N_iN_hN_{guide})$ for each updating. If we use large model, the calculation cost becomes very high. However, we use compact model because the background is to use models for P/WCDs. Especially, we often use 10 hidden neurons, therefore the calculation cost is not so high. Moreover, if we do not use so many data for guide data, the cost does not become huge, then we can update model by OLTA-GD on P/WCDs.

4. Experiments

In this section, we investigate the accuracy performance and the steadiness of OLTA-GD with the two guide data selection methods by experiments using some public databases taken from the machine learning repository of the University of California at Irvine [1]. The used databases are statlog german credit data set (German), Indian Liver patient data set (ILP), mushroom data set (Mushroom), ozone level detection data set (Ozone), QSAR biodegradation data set (QSAR) [14], and seismic bumps data set (Seismic) [12]. In this experiment, we use databases that have many data in more cases because we would like to investigate the performance transitions by on-line trainings. **Table 1** shows the parameters of the databases.

For each database, 100 times of 5-fold cross validation [9] was conducted. The number of training data is $N_t = [N_d \times \frac{4}{5}]$, and the number of testing data is $N_d - N_t$ for each database. For the result, we calculate the accuracy of a confusion matrix and for testing data, and it is averaged over 100×5 runs. We normalize the training and testing data by using a rescaling method. The computer system configuration and environment used in the experiment are shown in **Table 2**.

4.1 Experimental Design

In this paper, we proposed OLTA-GD with two guide data selection methods. To compare the performance, we defined three methods with the abbreviations as follows:

- DBM-BP: MLP initialized by DBM algorithm, and updated on-line by BP [16] algorithm.
- DBM-RS: MLP initialized by DBM algorithm, and updated by OLTA-GD using the random selection.
- DBM-CCS: MLP initialized by DBM algorithm, and updated by OLTA-GD using the cluster center based selection.

To evaluate the performance, we calculated recognition rate (RR) based on the confusion matrix for test set and accuracy reduction counts (ARCs) for several different reduction levels (RLs). RR was used to measure the overall performance of the trained models, and ARC was used to measure the steadiness of the on-line training process. The definition of RR is given in Eq. (5), and the definition of ARC is Eq. (6), where $\mathrm{RR}^{(t)}$ is the RR value at time t. ARC is the number of times that the model performance is degraded over a special level in the on-line updating. In real applications, the model performance should not be degraded because it affects to the reliability of applications. The ARC is a measurement of the performance. In this paper, we used 1% and 5% for the RL. By RL = 1% setting, we confirmed steadiness for a small rate, and we also confirmed the performance for a large rate which is RL = 5%. We investigated the significance of performance reduction by the two settings. For RR, the higher values mean the better performance, and for ARC, the lower values indicate better steadiness. Also, to confirm the performance of on-line training, we calculated rising or falling (RF) value for each result, the RF value was computed by Eq. (7), where t_{end} is the end time of on-line training in each database. If the RF value is positive, the performance is upgraded by on-line training.

$$\mathrm{RR} = \frac{\text{The number of correct data}}{\text{The number of test data}} \tag{5}$$

$$\mathrm{ARC} = \begin{cases} \mathrm{ARC} + 1 & (\mathrm{RR}^{(t+1)} - \mathrm{RR}^{(t)} \leq -\mathrm{RL}) \\ \mathrm{ARC} & (\text{otherwise}) \end{cases} \tag{6}$$

$$\mathrm{RF} = \mathrm{RR}^{(t_{end})} - \mathrm{RR}^{(0)} \tag{7}$$

As for data normalization, the rescaling normalization converts the range $[\min(col_i), \max(col_i)]$ to $[F_{\min}, F_{\max}]$ for each feature, where col_i is a set of i-th features in given training set, $\min(X)$ and $\max(X)$ return the minimum and maximum values from set X, and $F_{\min}$ and $F_{\max}$ are given parameters ($F_{\min} < F_{\max}$). In our experiment, the rescaled range was fixed to $[-1, +1]$.

In MLP training, we used the BP algorithm to train the model. For the BP algorithm, the learning rate was fixed to 0.5. The maximum number of training epochs was 1,000. The number of input neurons of the MLPs was N_f. The number of hidden neurons was fixed to 10. The number of output neurons was 1.

For DBM parameters, we set $\epsilon = 0.1$, $\delta_{DB} = 0.1$, $\delta_{outlier} = 0.2$, and $N_{DBM} = 10$. About SVM settings, we used soft-margin SVM. The training algorithm was sequential minimal optimization [15]. The kernel function was radial basis function ($\kappa(x_1, x_2) = \exp(-\|x_1 - x_2\|^2)$), and the training parameter C was set to 1.

In OLTA-GD, we changed the number of guide data N_{guide} in 1 to 10 per 1, and specified values 15, 20, 25, and 30. As for the specified values, we confirmed the performance of too many guide data settings.

Table 1 Features of public databases from Ref. [1]. These databases have two classes.

	Number of Classes (N_c)	Number of Features (N_f)	Number of Data (N_d)
German	2	24	1,000
ILP	2	10	583
Mushroom	2	22	8,123
Ozone	2	72	2,536
QSAR	2	41	1,055
Seismic	2	18	2,584

Table 2 Machine specifications and environments.

Machine	Apple iMac 21.5-inch, Late 2013
OS	Mac OS X 10.9.
CPU	Intel Core i5 2.7 GHz
Memory	8 GB
Program Language	C++
Compiler	Apple LLVM version 6.0

4.2 How to Divide Training Data Sets for Our Purpose

In this study, we compare the performance of on-line training methods. The on-line training updates a model initialized by an off-line training algorithm. Therefore, we have to prepare two training sets from a given training set which is allocated by the cross validation method. The two training sets are off-line training set and on-line training set for off-line training and on-line training respectively. We set $N_{off-line}$ data for the off-line training set, and the other data are allocated to the on-line training set. In this experiment, we set the number of off-line training data $N_{off-line}$ to 100, and the other data were assigned to the on-line

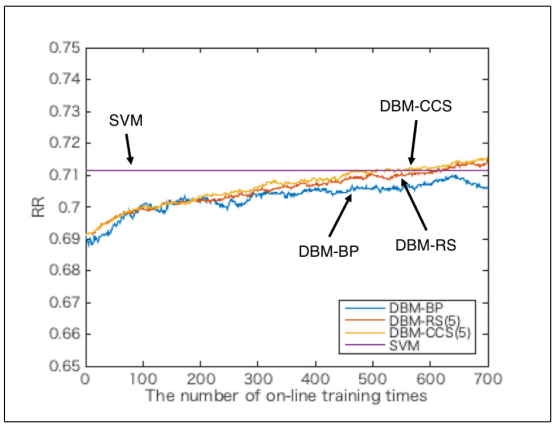

Fig. 4 RR transition graph for each method with $N_{guide} = 5$ setting in German database.

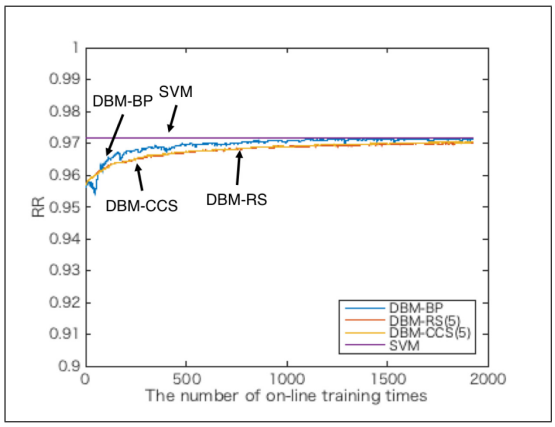

Fig. 7 RR transition graph for each method with $N_{guide} = 5$ setting in Ozone database. In this figure, DBM-RS/DBM-CCS are almost the same.

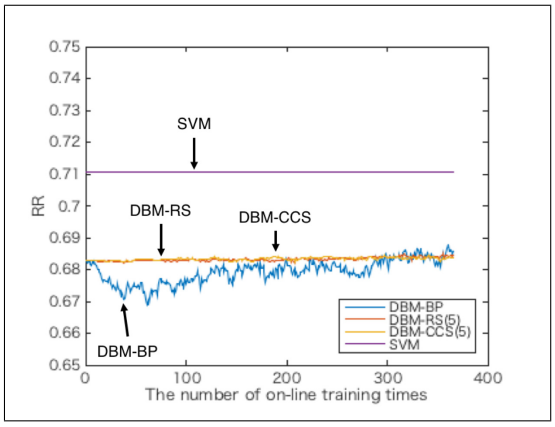

Fig. 5 RR transition graph for each method with $N_{guide} = 5$ setting in ILP database.

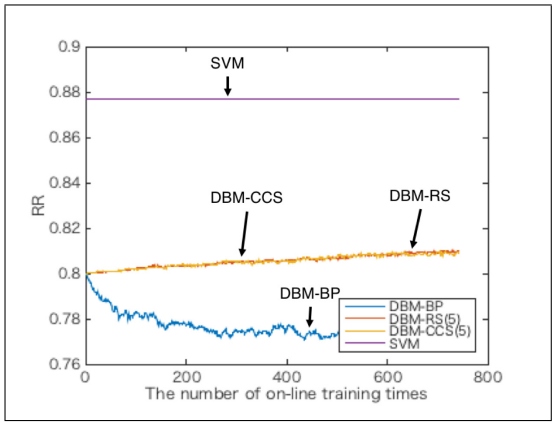

Fig. 8 RR transition graph for each method with $N_{guide} = 5$ setting in QSAR database.

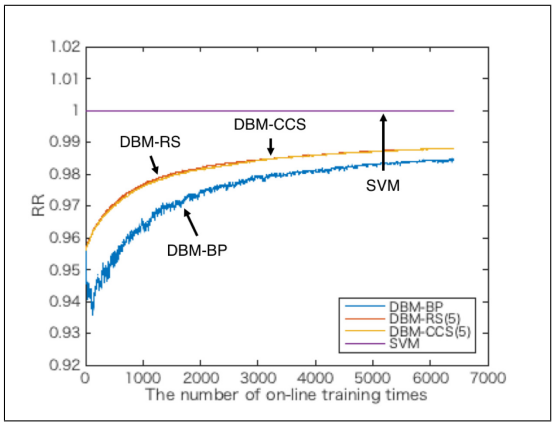

Fig. 6 RR transition graph for each method with $N_{guide} = 5$ setting in Mushroom database. In this figure, DBM-RS/DBM-CCS are almost the same.

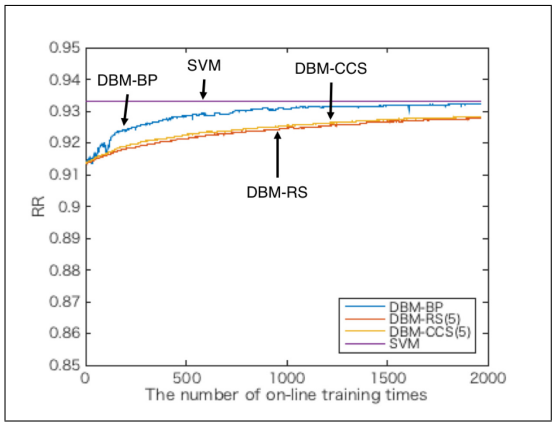

Fig. 9 RR transition graph for each method with $N_{guide} = 5$ setting in Seismic database.

training set because we use many data for on-line training.

4.3 Analysis and Discussion on Performance Comparison of Comparison Methods

Figures 4–9 reveal averaged RR transition graphs through on-line training of all methods with $N_{guide} = 5$ setting in each database. For other N_{guide} settings, the trends are similar to the results of $N_{guide} = 5$ setting. **Figures 10–15** show RF values of all methods by changing N_{guide} settings in each database. In the figures, we show RR results of SVMs obtained off-line training with all training data by some horizontal lines for the upper limits. The RF value presents the performance improvement of a method by on-line training. **Figures 16–21** indicate ARC value results of each method and every database by changing N_{guide} settings. **Figure 22** shows the averaged RF transition graph of all databases for DBM-RS and DBM-CCS by changing N_{guide} settings. And **Fig. 23** gives the averaged ARC transition graph of all databases for DBM-RS and DBM-CCS with all N_{guide} settings.

4.3.1 Discussion on RR Results

In Figs. 4–9, we use horizontal lines to show the performance of the SVMs obtained via off-line training. These lines are in

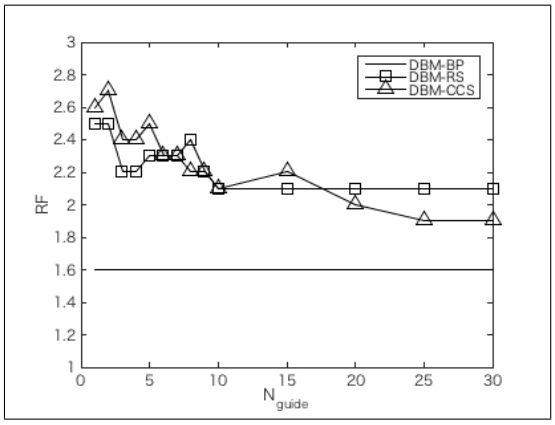

Fig. 10 RF value transition graph for each method by changing N_{guide} setting in German database.

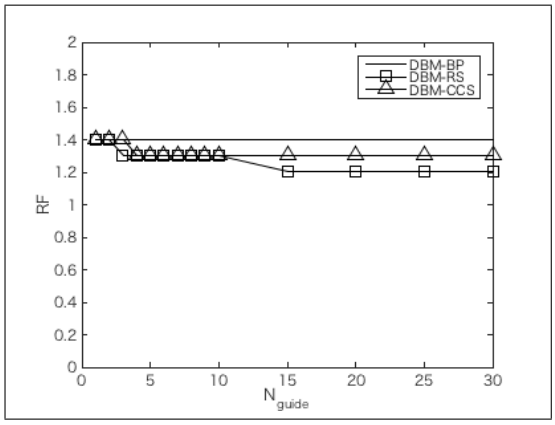

Fig. 13 RF value transition graph for each method by changing N_{guide} setting in Ozone database.

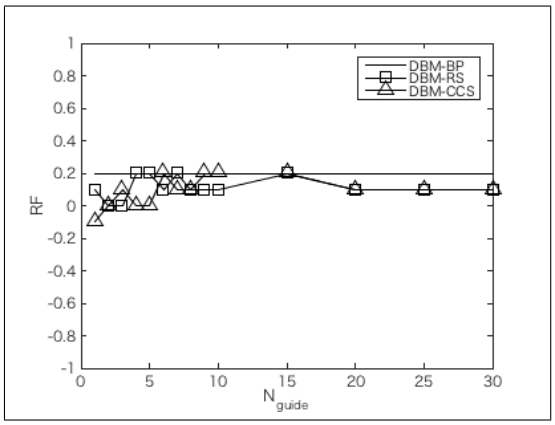

Fig. 11 RF value transition graph for each method by changing N_{guide} setting in ILP database.

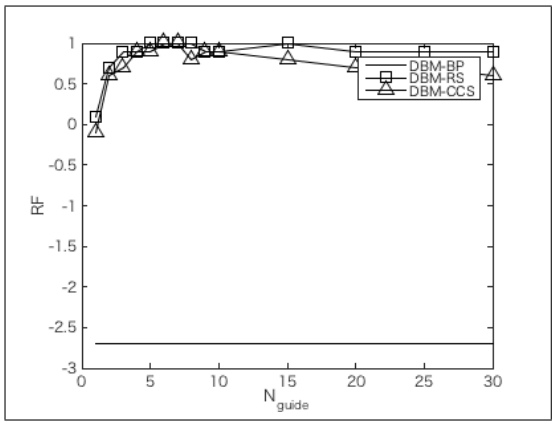

Fig. 14 RF value transition graph for each method by changing N_{guide} setting in QSAR database.

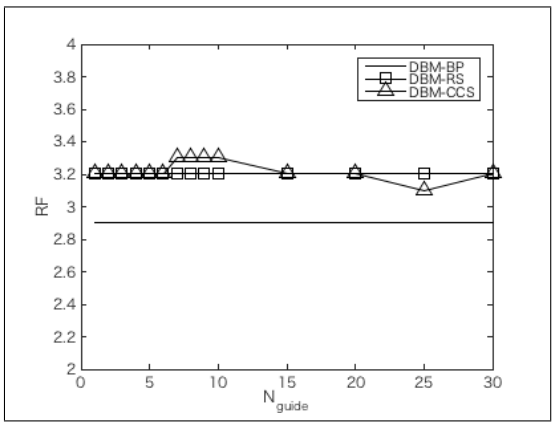

Fig. 12 RF value transition graph for each method by changing N_{guide} setting in Mushroom database.

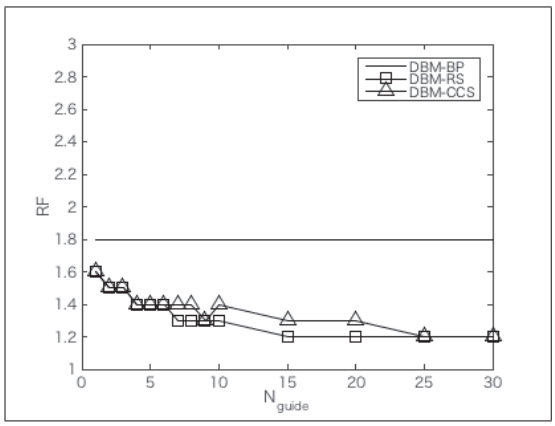

Fig. 15 RF value transition graph for each method by changing N_{guide} setting in Seismic database.

fact the upper limits for on-line training. From Fig. 4, Fig. 7, and Fig. 9, we can see that the proposed methods can obtain results (after on-line training) comparable to or better than off-line SVMs for the German, Ozone, and Seismic databases. From Fig. 6, we can see that the proposed method can obtain relatively good results, and the performance can be close to that of the off-line SVM if we have more data. From Fig. 5 and Fig. 8, however, we can see that it is difficult to obtain results comparable to the off-line SVMs through on-line training. The main reason we think is that these two datasets may have little redundancy. Usually, for datasets with little redundancy, it is difficult to obtain a good model using a small part of the data.

To see all of RR transition graphs given in Figs. 4–9, the performance improvement of our proposed methods are comparable to or better than DBM-BP. In some cases, DBM-BP is better than the proposed methods in early on-line training stages, or equivalent to the proposed methods. However, DBM-BP performance is often fluctuated or decreased through on-line training. In Seismic database, DBM-BP performance is a little better than the proposed method in the early stage of on-line training, but DBM-BP performance is decreased and/or fluctuated over on-line training in QSAR, and ILP databases. In comparison between Seismic

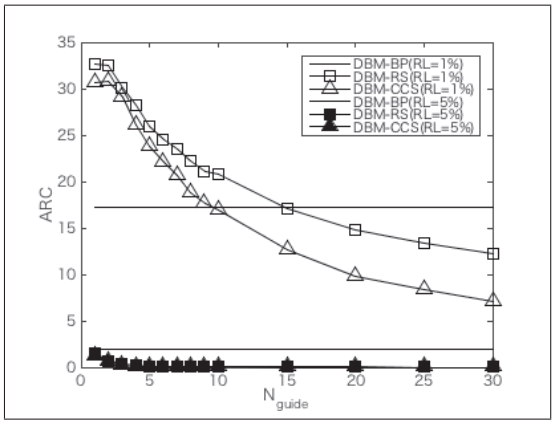

Fig. 16 ARC value transition graph for each method by changing N_{guide} setting in German database.

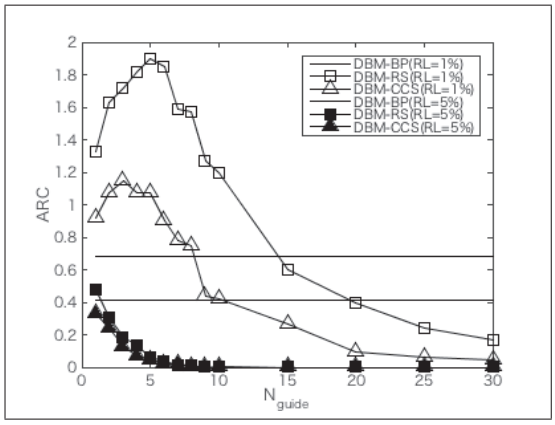

Fig. 19 ARC value transition graph for each method by changing N_{guide} setting in Ozone database.

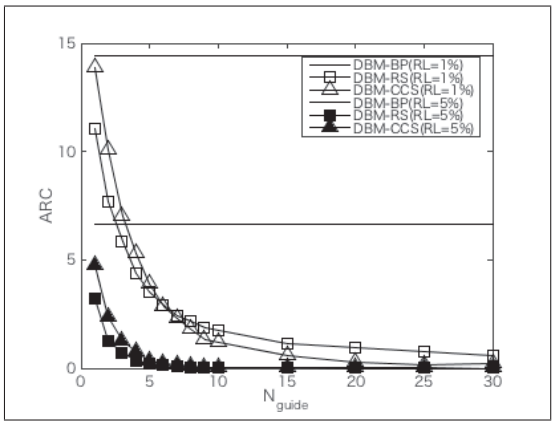

Fig. 17 ARC value transition graph for each method by changing N_{guide} setting in ILP database.

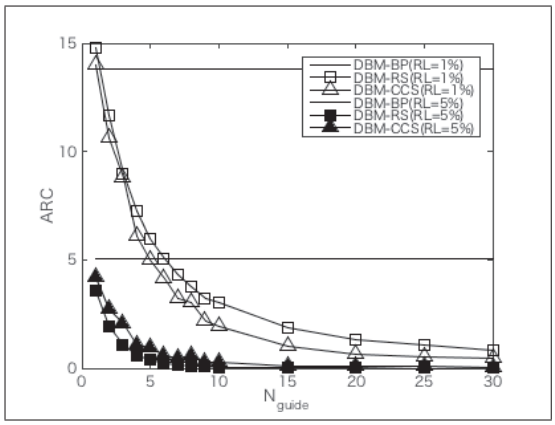

Fig. 20 ARC value transition graph for each method by changing N_{guide} setting in QSAR database.

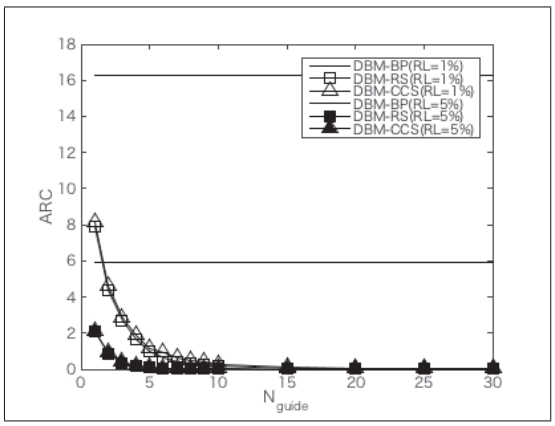

Fig. 18 ARC value transition graph for each method by changing N_{guide} setting in Mushroom database.

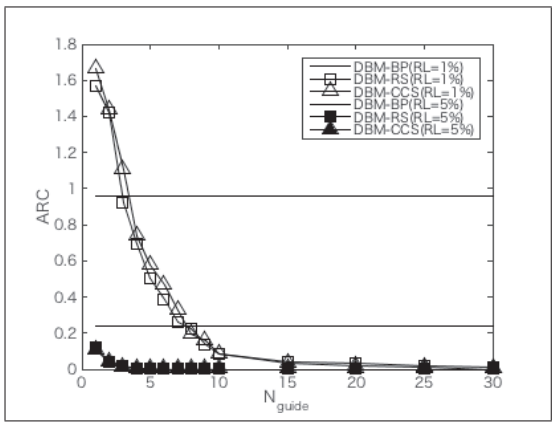

Fig. 21 ARC value transition graph for each method by changing N_{guide} setting in Seismic database.

and ILP databases, the difference is the difficulty for on-line training. For easy data sets, we can improve the model steadily even if we do not use guide data. Actually, the guide data set is a "damper" for on-line training. On the other hand, for difficult data sets such as ILP database, it is necessary to use a relatively strong damper to control the training process, to make it better steady. However, in the Seismic database, the proposed methods increase the performance gradually, and the RR difference between DBM-BP and the proposed methods is less than 1% by the end of on-line training. The performance of the proposed methods may become better than DBM-BP if we continue to update the models. We expect that the performance of OLTA-GD is upgraded steadily, so the result is one of our expectations. The BP algorithm only sometimes decreases the performance by on-line training in real time, and the OLTA-GD can improve the performance gradually from the results. The amounts of increasing the performance are different by databases, for example, the amount is around 2% in German database, and is around less than 1% in ILP database. The amounts are shown in Figs. 10–15 by RF values. In many cases, such as ILP, Mushroom, Ozone and Seismic databases, the differences are less than 1%. However, in the other cases, the performance of DBM-RS/DBM-CCS with proper set-

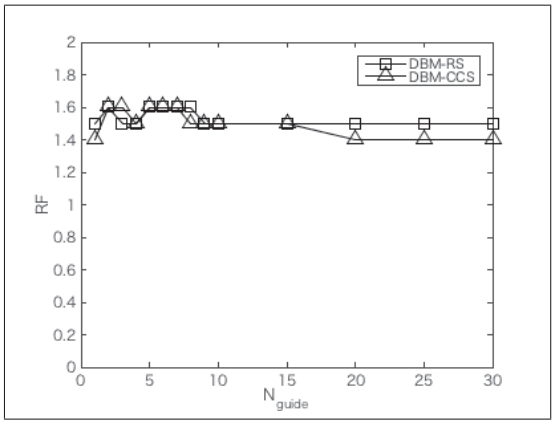

Fig. 22 RF transition graph of averaged all database results by changing N_{guide} setting for DBM-RS and DBM-CCS methods.

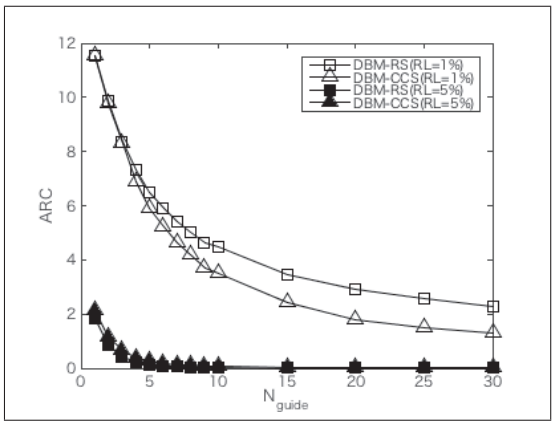

Fig. 23 ARC transition graph of averaged all database results by changing N_{guide} setting for DBM-RS and DBM-CCS methods.

ting shows higher than 1% compared with DBM-BP. Therefore, the proposed methods have equivalent or better performance than the BP algorithm only.

4.3.2 Discussion on Steady Performance Improvement

The steadiness of DBM-RS/DBM-CCS is better than DBM-BP, and these are almost the same from Figs. 16–21. In RL = 5% setting, the proposed methods are better than DBM-BP in all cases. This means that the performance of the proposed methods are not decreased too much many times compared with DBM-BP. In RL = 1% setting, if we use few guide data, sometimes the proposed methods are worse than DBM-BP. However, the proposed methods are better when N_{guide} is increased. Thus, we can get better results when we use a proper N_{guide} setting for the proposed methods. In comparison between DBM-RS and DBM-CCS, these between the ARC values less than 1 in many cases, although sometimes the difference is more than 1 depends on the N_{guide} setting. For an example, in German database, DBM-RS and DBM-CCS have almost the same ARC values when N_{guide} is set around 5. However, when N_{guide} is increased, the ARC difference is also increased. In this sense, DBM-CCS is better than DBM-RS. To see the averaged results of all databases, in Fig. 23, the averaged ARC values are almost the same when N_{guide} is more than 10. If we use more data for guide data, the ARC values can be reduced further. However, to save calculation cost, we should avoid using too many guide data. Therefore, if we use $N_{guide} \leq 10$, the steadiness of DBM-RS and DBM-CCS are almost the same.

4.3.3 Discussion on Guide Data

For OLTA-GD using the DBM algorithm, random selection is enough to obtain good guide data set for upgrading the model performance steadily. To see the averaged RR results in Fig. 22, the performance of two selection methods are almost the same when N_{guide} is less than 10. However, over $N_{guide} = 10$, the performance of them becomes worse. Therefore, if we set larger value to N_{guide}, model performance improvement is degraded. In comparison among the two methods, the performance of the random selection is little better than the cluster center based selection when N_{guide} is more than a certain value. The differences of selected data may be the data distributions. The random selection picks up data randomly, then the distribution is related to the data distribution of the candidate set. The candidate set has many data near the DB, then the most selected data will be near the DB. On the other hand, the cluster center based selection picks up a datum from each cluster, so the distribution is related to the clusters. The clusters are found by k-means algorithm, then the distribution is uniform the feature space. To upgrade the model performance steadily, data close to the DB should be used. Thus, the random selection is better for updating performance. Also to see the steadiness in Fig. 23, these are almost the same, but are different when we use many data for guide data set in RL = 1% setting. The cluster center based selection is better for the situation, but we should use less data to save the calculation cost for updating. When we use less than 10 data, the steadiness and the accuracy performance are almost the same in both methods, and the calculation cost of the random selection is less than the cluster center based selection. Therefore, the random selection is better for our purpose from the experimental results.

4.3.4 Discussion on Better N_{guide} Setting

As for the number of guide data, we recommend to use $N_{guide} = 5$ for our purpose. From Fig. 22, the performance are almost the same when we set less than 10 for N_{guide}. On the other hand, to see Fig. 23, the difference of RL = 1% between $N_{guide} = 5$ and 10 is around 1-2. However in RL = 5%, the ARC value is almost converged when $N_{guide} = 5$. In addition, the calculation cost of $N_{guide} = 10$ is almost double compared with $N_{guide} = 5$ setting. We update models on P/WCDs, thus the calculation cost should be conserved. From these reasons, we should use around five data for guide data to upgrade model performance steadiness in our purpose.

4.3.5 Overall Discussion

From the above discussions, the OLTA-GD using the random selection with $N_{guide} = 5$ setting is good for our purpose. The OLTA-GD can upgrade the model performance steadily. For guide data selection, random selection can select good enough data for the guide data set. When we use five data for the guide data set, the steadiness is almost converged. Therefore, the method using the setting is the best method among the comparison methods.

5. Conclusions

In this paper, we have proposed OLTA-GD to retrain existing models initialized by the DBM algorithm and investigated

the performance. For the guide data set, we proposed two guide data selection methods; namely random selection and cluster center based selection. The random selection picks up guide data randomly from candidate set which contains all available data. The cluster center based selection partitions the candidate set into some clusters by k-means algorithm in off-line phase, and picks up a guide datum from each cluster. From experimental results, the OLTA-GD could upgrade model performance more steadily than BP algorithm, and the random selection is good enough method for the guide data selection. For guide data setting, we should use around five data for the random selection method.

As for future work, we would like to improve the performance of OLTA-GD. In the experimental results, OLTA-GD performs poorly for some cases. To improve the performance, we will focus on updating candidate set. In current study, we add observed data into the candidate set without conditions. Sometimes, noisy data may be added into the set. In the random selection, the effectiveness is influenced by the data distribution of the candidate set. By avoiding data that are noisy or far from the DB, the effectiveness of updating model may be improved. We will find proper conditions for this purpose. We also would like to apply the OLTA-GD to real applications. In this paper, we focus on theoretically part of on-line training algorithms. Customizing to each user using real problems on P/WCDs is also one of our purposes. We will investigate the model transition for each user by on-line training in the future study.

Acknowledgments This work was supported by Grant-in-Aid for JSPS Fellows Grant Number 15J10477.

References

[1] Bache, K. and Lichman, M.: UCI machine learning repository (2013).
[2] Chen, Y.-H., Krishna, T., Emer, J. and Sze, V.: Eyeriss: An energy-efficient reconfigurable accelerato for deep convolutional neural networks, *IEEE International Solid-State Circuits Conference, ISSCC 2016*, Digest of Technical Papers (2016).
[3] Collins, M.: Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms, *Proc. ACL-02 Conference on Empirical Methods in Natural Language Processing*, *EMNLP '02*, Vol.10, pp.1–8, Stroudsburg, PA, USA, Association for Computational Linguistics (2002).
[4] Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S. and Singer, Y.: Online passive-aggressive algorithms, *J. Mach. Learn. Res.*, Vol.7, pp.551–585 (2006).
[5] Dong, E.Q. and Huang, X.: A new compact support kernel of support vector machines, *14th Asia-Pacific Conference on Communications* (*APCC 2008*), pp.1–4 (2008).
[6] Geebelen, D., Suykens, J.A.K. and Vandewalle, J.: Reducing the number of support vectors of svm classifiers using the smoothed separable case approximation, *IEEE Trans. Neural Networks and Learning Systems*, Vol.23, No.4, pp.682–688 (2012).
[7] Hornik, K., Stinchcombe, M. and White, H.: Multilayer feedforward networks are universal approximators, *Neural Netw.*, Vol.2, No.5, pp.359–366 (1989).
[8] Kaneda, Y., Pei, Y., Zhao, Q. and Liu, Y.: Improving the performance of the decision boundary making algorithm via outlier detection, *Journal of Information Processing*, Vol.23, No.4, pp.497–504 (2015).
[9] Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection, *Proc. 14th International Joint Conference on Artificial Intelligence*, *IJCAI'95*, Vol.2, pp.1137–1143, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc. (1995).
[10] Lee, Y.J. and Huang, S.Y.: Reduced support vector machines: A statistical theory, *IEEE Trans. Neural Networks*, Vol.18, No.1, pp.1–13 (2007).
[11] Lin, K.-M. and Lin, C.-J.: A study on reduced support vector machines, *IEEE Trans. Neural Networks*, Vol.14, No.6, pp.1449–1459 (2003).
[12] Sikora, M. and Wrobel, L.: Application of rule induction algorithms for analysis of data collected by seismic hazard monitoring systems in coal mines, *Archives of Mining Sciences*, Vol.55, No.1, pp.91–114 (2010).
[13] MacQueen, J.: Some methods for classification and analysis of multivariate observations, *Proc. 5th Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, pp.281–297, Berkeley, California, University of California Press (1967).
[14] Mansouri, K., Ringsted, T., Ballabio, D., Todeschini, R. and Consonni, V.: Quantitative structure - activity relationship models for ready biodegradability of chemicals, *Journal of Chemical Information and Modeling*, Vol.53, pp.867–878 (2013).
[15] Platt, J.: Sequential minimal optimization: A fast algorithm for training support vector machines, Technical Report MSR-TR-98-14, Microsoft Research (1998).
[16] Rumelhart, D.E., Hinton, G.E. and Williams, R.J.: Learning representations by back-propagating errors, *Nature*, Vol.323, No.6088, pp.533–536 (1986).
[17] Shalev-Shwartz, S., Singer, Y., Srebro, N. and Cotter, A.: Pegasos: primal estimated sub-gradient solver for svm, *Mathematical Programming*, Vol.127, No.1, pp.3–30 (2011).
[18] Steinwart, I.: Sparseness of support vector machines – some asymptotically sharp bounds, *Advances in Neural Information Processing Systems 16*, MIT Press (2004).
[19] Vapnik, V.N.: *Statistical learning theory*, Wiley, 1st edition (1998).
[20] Yang, L., Hanneke, S. and Carbonell, J.: A theory of transfer learning with applications to active learning, *Machine Learning*, Vol.90, No.2, pp.161–189 (2013).

Yuya Kaneda was born in 1990. He received his B.E. from the University of Aizu in 2012, his M.E. in 2014 and Ph.D. degree in 2017. His research interests include computational intelligence, evolutionary computing, neural networks, awareness computing, and automated driving systems. He is a member of IEEE, IPSJ, and JSAI.

Qiangfu Zhao received his Ph.D. degree from Tohoku University of Japan in 1988. He joined the Department of Electronic Engineering, Beijing Institute of Technology, China in 1988, first as a post doctoral fellow and then an associate professor. He was an associate professor from October 1993 at the Department of Electronic Engineering, Tohoku University, Japan. He joined the University of Aizu, Japan from April 1995 as an associate professor, and became a tenure full professor in April 1999. Professor Zhao's research interests include image processing, pattern recognition and understanding, computational intelligence, neurocomputing, evolutionary computation and awareness computing.

Yong Liu is currently a senior associate professor at the University of Aizu, Japan. He received his Ph.D. degrees from Wuhan University, China, and the University of New South Wales, Australia, in 1994, and 1999, respectively. His research interests include evolutionary computation and neural networks. He is a member of IEEE.