2006 年情報論的学習理論ワークショップ
2006 Workshop on Information-Based Induction Sciences (IBIS2006)
Osaka, Japan, October 31 - November 2, 2006.

Coping with Active Learning with Model Selection Dilemma: Minimizing Expected Generalization Error

Neil Rubens[*] Masashi Sugiyama[*]

Abstract: Optimally designing the location of training input points (active learning) and choosing the best model (model selection) are two important ingredients of supervised learning and have been studied extensively. However, these two issues seem to have been investigated separately as two independent problems. If training input points and models are simultaneously optimized, the generalization performance would be further improved. In this paper, we therefore propose a new approach called *ensemble active learning* for solving the problems of active learning and model selection at the same time. We demonstrate by the numerical experiments with toy and benchmark data sets that the proposed approach compares favorably with alternative methods.

1 Introduction

In supervised learning, when we are allowed to choose the location of training input points (or queries), we want to determine them so that the generalization error is minimized. This problem is called *active learning* (AL), and has been studied extensively (e.g., [4, 6, 2, 5, 12]). On the other hand, *model selection* (MS) is another important issue in supervised learning: a model (the type and number of basis functions, regularization parameter etc.) that will be trained on the training data is determined so that the generalization error is minimized (e.g., [1, 9, 10, 3, 11, 13]).

Although AL and MS share a common goal of minimizing the generalization error, they seem to have been studied separately as two independent problems. If training input points and models are simultaneously optimized, the generalization performance would be further improved. In this paper, we therefore try to perform AL and MS at the same time, which we call *active learning with model selection* (ALMS).

After the problem formulation (Section 2), we first show that ALMS is not a straightforward task because there exists a dilemma between AL and MS: these two problems can not be solved by simply combining standard AL methods and MS methods (Section 3). Alternative approaches such as a sequential approach may also be considered, but as we point out, they have a number of limitations (see Section 3). To overcome these limitations, we introduce a new approach which we refer to as *ensemble active learning* (EAL) (Section 4). EAL has a natural *minimax* property. Therefore it is stable and thus reliable in practice. Simulations with toy and benchmark data sets clearly underline the effectiveness of our new approach (Section 5).

[*]Department of Computer Science, Tokyo Institute of Technology, Japan, http://sugiyama-www.cs.titech.ac.jp/

2 Problem formulation

The goal of supervised learning is—based on a finite number of training samples $\{(\boldsymbol{x}_i, y_i)\}_i$, to obtain an approximation $\hat{f}(\boldsymbol{x})$ of an unknown learning target function $f(\boldsymbol{x})$ that minimizes a particular generalization error G. In this supervised learning procedure, there are two factors we can control in order to improve generalization performance: training input points $X = \{\boldsymbol{x}_i\}_i$ and a model M. Here a model refers to, e.g., basis functions and regularization constant. The problem of optimizing the location of training input points is called *active learning* (AL):

$$\min_{X} G. \tag{1}$$

On the other hand, the problem of optimizing a model is called *model selection* (MS):

$$\min_{M} G. \tag{2}$$

The generalization error G is generally inaccessible since it includes the unknown learning target function $f(\boldsymbol{x})$ etc. Therefore, the essence of the AL or MS research is to accurately estimate the generalization error. So far, extensive studies have been conducted on AL (e.g., [4, 6, 2, 5, 12]) and MS (e.g., [1, 9, 10, 3, 11, 13]). However, it seems that these issues have been studied separately. Given that both AL and MS share a common goal—minimizing the generalization error G, it would be natural to try to optimize training input points X and models M simultaneously, by which generalization capability would be further improved. We refer to this problem as *active learning with model selection* (ALMS):

$$\min_{M,X} G. \tag{3}$$

This paper is aimed at solving the ALMS problem.

3 Possible approaches of solving ALMS

In this section we discuss strengths and limitations of possible approaches of solving the ALMS problem (3).

Direct batch method A naive and direct solution to this problem would be to simultaneously optimize M and X. However, this direct approach may not be possible due to the *ALMS dilemma*: when selecting training input points with existing AL methods, model M must have been fixed; and when choosing a model with existing MS methods training input points X must have been fixed and corresponding output values must have been gathered. Therefore, this direct batch approach may not be possible.

Sequential method A typical approach of coping with the above ALMS dilemma is with the sequential method: iteratively choosing model and training input points (e.g., [7]). At each step, the most promising model is chosen by an MS method and the next input point is optimized for the chosen model by an AL method.

However, this sequential approach has a number of limitations. In the early stages of learning, the number of training samples is small so the sequential method tends to favor simpler models with small variance. As the number of training samples increases, the sequential method tends to choose more complex models (see Section 5.2 and Figure 3). Since the optimal location of training input points depends *strongly* on the model, training input points that are good for one model may be poor for others (see Section 5.2 and Figure 4). Therefore, training input points chosen by the sequential method at the earlier stages may not be as useful for more complex models chosen eventually, which can heavily degrade the performance.

Another limitation of the sequential method is that most MS methods and AL methods require training input points that are independent and identically distributed (i.i.d.). However, the sequential method violates this common requirement since the choice of the next training input point at each stage is dependent on the earlier stages. Therefore, it would not be so straightforward to develop a theoretically justifiable sequential algorithm.

Random batch method Let us consider the following batch approach which we refer to as the *random batch approach*. First, the initial model is randomly chosen before any single training samples are gathered. Then unlike the sequential method, all training input points are chosen at once for the initially selected model. Then training output values are gathered at the determined training input points, and MS is carried out to determine the final model.

Although this random batch approach is not practical, it allows to overcome some of the limitations of the sequential method such as the drift of the target model during the learning process and the violation of the i.i.d. condition. However, the usefulness of the training input points designed by the random batch method is strongly dependent on the initial model which has to be chosen randomly because of the lack of the prior preference of the models. Therefore if the initial model and finally chosen model are significantly different, training input points designed for the initial model may not be so useful for the finally chosen model.

4 Proposed method: ensemble active learning

As we discussed in the previous section, the direct batch approach may not be feasible, the sequential approach has a number of drawbacks, and the random batch approach is not practical since its performance is influenced by the randomness of the initial model choice. In this section, we propose a new ALMS method that can overcome the above problems.

Basic idea Although the random batch method is unstable, it has the desirable properties: designed training input points are optimal for the initial model and the i.i.d. condition is not violated. If the performance of the random batch method could be stabilized, it would be useful and a promising candidate for use in the ALMS scenario. For this reason, we stabilize the random batch method.

The instability of the random batch method comes from the randomness of the choice of the initial model. When the training input points are chosen for the initial model, in this case the random one, they become *overfitted* to the initial model; since the optimal location of training input points depends *strongly* on the model, training input points that are good for one model may be poor for others (see Section 5.2 and Figure 4). Therefore, we may stabilize the random batch method by not designing training input points X *specifically* for the initial model, but by designing them for *all* of the models. To this end, we determine X so that the *expected* generalization error over *all* models is minimized. A naive approach would be to use

$$\min_X \sum_M G_M(X)P(M), \tag{4}$$

where G_M is the generalization error of the model M and $P(M)$ is the prior probability of the model M. If no prior information on goodness of models is available, the term $P(M)$ will be just assigned the uniform prior and as a result it will become a constant and have no resulting effect.

Normalization However, under certain conditions e.g. if a model has a relatively large magnitude of G than others, this naive approach can still overfit training input points to that model; even though the model with the large generalization error is less likely to be chosen finally in the model

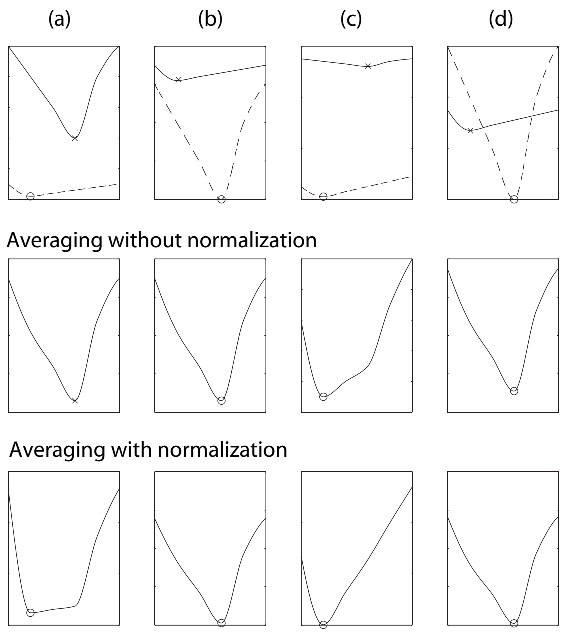

Figure 1: Illustrates the need for and the effects of proposed normalization scheme (5) compared with unnormalized counterpart (4). See the main text for detail.

selection stage. In order to alleviate this problem, we propose normalizing the generalization errors of the models so that all of the models have *equal* influence on the choice of the training input points without regard for the magnitude or the range of G of any given model. This may be achieved by the following criterion:

$$\sum_{M} \frac{G_M(X)}{\sum_{X'} G_M(X')} P(M). \tag{5}$$

To illustrate the effect of this normalization scheme, we show schematic pictures in Figure 1, where the horizontal axis denotes the choice of training input points and the vertical axis is corresponding to generalization error. Four exemplary cases are considered here: same/different mean and same/different magnitude (see each column in the figure). The top row shows the generalization error curves for two different models, denoted by solid and dashed lines, respectively. The dashed line corresponds to a better model since the minimum is smaller than that of the solid line. If we just average the two curves (Eq.(4) with uniform prior), we obtain the curves described in the second row. For the cases (b), (c), and (d), the minimizers of the averaged curve agree with the minimizers of the better model. Therefore AL by Eq.(4) works well. However for the case (a), the averaged curve is dominantly affected by the bad model (solid line in the top row). In this case, just taking the average is not appropriate. On the other hand, when we employ the normalized criterion (5), we obtain the curves described in the bottom row. This improves the result of case (a) and still works well for the other cases.

In the above proposed approach, all of the models contribute to the selection of training input points. Therefore, the proposed method could be interpreted as applying a popular idea of *ensemble learning* to the problem of AL. For this reason, we refer to the proposed approach as *ensemble active learning* (EAL).

Properties By its definition, EAL is thought to have a *minimax* property—we are trying to reduce the worst-case generalization error. Since in the random batch approach, the initial model is selected randomly, the training input points are overfitted to the *randomly* selected model. If the finally selected model is significantly different from the initial model, the designed training input points may be poor for the final model. On the other hand, if the initial model is by chance the same as the final model, the random batch approach provides the optimal solution. Therefore, the resulting performance of the random batch method is highly unstable. On the other hand, in the proposed EAL, training input points are chosen in such a way that they are equally good for all of the models; the designed training input points may not be optimal for any of the given models, but are likely to be useful for most of the models. As a result, EAL allows to *hedge* the risk of choosing training input points overfitted to a *randomly* chosen initial (and possibly inferior) model.

5 Numerical experiments

The EAL idea could be applied to any supervised learning situations. In this section, we consider a linear regression scenario and quantitatively compare the proposed EAL method with other approaches through numerical experiments using the toy and benchmark data sets.

5.1 Setting

Here we briefly describe the basic setting of linear regression that is common to toy and benchmark data sets.

Let us consider the regression problem of learning a real-valued function $f(\boldsymbol{x})$ defined on $\Re^d$ from training samples $\{(\boldsymbol{x}_i, y_i) \mid y_i = f(\boldsymbol{x}_i) + \epsilon_i\}_{i=1}^n$, where d is the dimension of the input vector, n is the number of training samples, and $\{\epsilon_i\}_{i=1}^n$ are i.i.d. noise with mean zero and unknown variance σ^2. We suppose that the training input points $\{\boldsymbol{x}_i\}_{i=1}^n$ are independently taken from a user-defined distribution with density $p(\boldsymbol{x})$.

In this regression scenario, we define the generalization error G of a learned function $\widehat{f}(\boldsymbol{x})$ by the expected squared error for *test* input points. We assume that the test input points are drawn independently from a distribution with density $q(\boldsymbol{x})$. Then G is expressed as

$$G = \int \left(\widehat{f}(\boldsymbol{x}) - f(\boldsymbol{x})\right)^2 q(\boldsymbol{x}) d\boldsymbol{x}. \tag{6}$$

We employ the following linear regression model:

$$\widehat{f}(\boldsymbol{x}) = \sum_{i=1}^{b} \alpha_i \varphi_i(\boldsymbol{x}), \tag{7}$$

where $\{\varphi_i(\boldsymbol{x})\}_{i=1}^{b}$ are fixed linearly independent functions and $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_b)^\top$ are parameters to be learned.

The AL scenario is a typical case of the *covariate shift* [11], where the training input distribution is different from the test input distribution, i.e., $p(\boldsymbol{x}) \neq q(\boldsymbol{x})$. Under the covariate shift, ordinary least squares (OLS) is not generally *consistent* anymore; instead, the following *adaptive importance-weighted least-squares* (AIWLS) works well [11, 13]:

$$\min_{\boldsymbol{\alpha}} \left[\sum_{i=1}^{n} \left(\frac{q(\boldsymbol{x}_i)}{p(\boldsymbol{x}_i)} \right)^{\lambda} \left(\widehat{f}(\boldsymbol{x}_i) - y_i \right)^2 \right], \tag{8}$$

where $0 \leq \lambda \leq 1$. AIWLS has the following property. When $\lambda = 0$, AIWLS is reduced to OLS so is biased. However it tends to have small variance. When $\lambda = 1$, AIWLS is consistent so it has small bias. However, it tends to have large variance. In practice, an intermediate λ often produces good results. Therefore, we have to choose λ appropriately by an MS method (e.g., [11, 13]).

Let $\boldsymbol{X}$ be the *design matrix*, i.e., $\boldsymbol{X}$ is the $n \times b$ matrix with the (i, j)-th element $\varphi_j(\boldsymbol{x}_i)$. Let $\boldsymbol{D}$ be the diagonal matrix with the i-th diagonal element $q(\boldsymbol{x}_i)/p(\boldsymbol{x}_i)$. Then the AIWLS estimator $\widehat{\boldsymbol{\alpha}}$ is computed by $\widehat{\boldsymbol{\alpha}} = \boldsymbol{L}\boldsymbol{y}$, where $\boldsymbol{L} = (\boldsymbol{X}^\top \boldsymbol{D}^\lambda \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{D}^\lambda$ and $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)^\top$.

In order to perform AL, the inaccessible generalization error G has to be estimated. Here we employ a generalization error estimator J suggested in the paper [12]:

$$J = \mathrm{tr}(\boldsymbol{U}\boldsymbol{L}\boldsymbol{L}^\top), \tag{9}$$

where $\boldsymbol{U}$ is the b-dimensional square matrix with the (i, j)-th element $\int \varphi_i(\boldsymbol{x})\varphi_j(\boldsymbol{x})q(\boldsymbol{x})d\boldsymbol{x}$. Note that the above J corresponds to the variance of the learned function if it is multiplied by the noise variance σ^2. In our implementation of AL, we replace G by the above J and optimize the training input density $p(\boldsymbol{x})$.

In order to perform MS, again G should be estimated. We may use typical model selection methods such as cross-validation. However, the current situation contains shift in covariates so the cross-validation is no longer unbiased. There exist some MS methods that can handle the covariate shift (e.g., [11, 13]). However, these MS methods assume that the training samples are i.i.d. Since the sequential method which we compare our EAL method with does not satisfy the i.i.d. assumption, using one of the above MS methods can bias our simulation results. Therefore, we decided to use the true G for MS and only evaluate the AL performance.

In the current setting, a model M is specified by the number and type of basis functions and the tuning parameter λ.

5.2 Toy data set

Here we demonstrate how EAL, random batch and the sequential method behave under a toy one-dimensional setting borrowed from [12].

The target function $f(x)$, depicted in the top graph of Figure 2, is a third order polynomial (see [12] for detail). Let the number of training samples to gather be $n = 100$ and $\{\epsilon\}_{i=1}^{n}$ be i.i.d. Gaussian noise with mean zero and standard deviation 0.3. Let the test input density $q(x)$ be the Gaussian density with mean 0.2 and standard deviation 0.4, which is assumed to be known in this illustrative simulation. We choose the training input density from a set of Gaussian densities with mean 0.2 and standard deviation $0.4c$, where $c = 0.8, 0.9, 1.0, \ldots, 3.8$. These density functions are illustrated in the bottom graph of Figure 2.

Let the number of basis functions be $b = 3$ and the basis functions be

$$\varphi_i(x) = x^{i-1} \text{ for } i = 1, 2, \ldots, b. \tag{10}$$

Since the model is the second order polynomial, the target function is not realizable. Here we use the above fixed basis functions and focus on choosing λ in AIWLS from $\{0, 0.5, 1\}$.

First, we analyze the behavior of the sequential method. In our the implementation, 10 training input points are chosen at each iteration. Figure 3 depicts the transition of the frequency of chosen λ in the sequential learning process (over 1000 trials). This clearly shows that the sequential method tends to favor larger λ (which has larger variance thus higher complexity) as the number of training samples increases.

Next we investigate the dependency between the best training input density (i.e., c) and the model (i.e., λ). For a fixed λ and a fixed c, we draw training input points $\{\boldsymbol{x}_i\}_{i=1}^{n}$ and gather output values $\{y_i\}_{i=1}^{n}$. Then we learn the parameter $\widehat{\boldsymbol{\alpha}}$ by AIWLS and computed the generalization error G. The mean G over 1000 trials as a function of c for each λ is depicted in Figure 4. This graph underlines that the best training input density c crucially depends on the model λ (cf. $\lambda = 0, 0.5$) and training input points that are good for one model could be poor for others. For example, if the training input density is optimized for the model $\lambda = 0$, $c = 1.1$ would be a very good choice. However, $c = 1.1$ is not so suitable for the model $\lambda = 1$.

Figures 3 and 4 illustrate a typical failure mode of the sequential method: the training input points optimized for one model could be poor for others, but the target model drifts during the sequential learning process. Therefore, the training input points designed by the sequential method may be poor for the finally chosen model as could be seen from Figure 5 which describes the box plot of the obtained generalization error of the proposed EAL, unnormalized EAL, random batch and sequential methods. Performance of random batch varies significantly due to the randomness of choice of the initial model. If the initial model and finally chosen model are significantly different, training input points designed for the initial model may not be so useful for the finally chosen model. On the other hand, EAL does not suffer from these problems since the ensemble of models is considered. Thanks to the stabilizing effect, the worst-case performance (95th percentile) is signif-

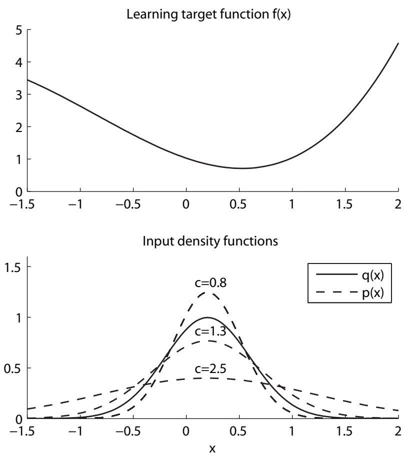

Figure 2: Target function and training/test input densities.

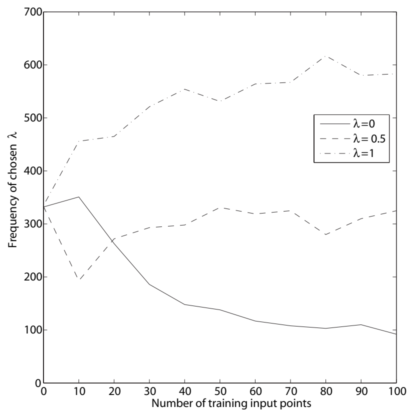

Figure 3: Frequency of chosen λ over 1000 trials as a function of the number of training samples.

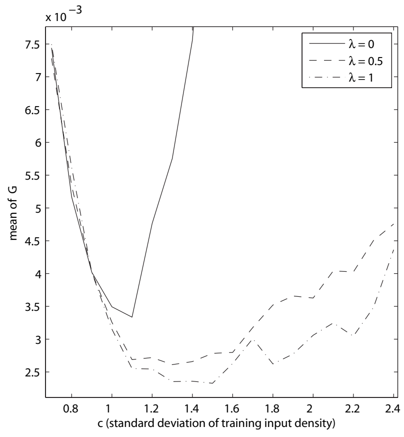

Figure 4: The mean generalization error over 1000 trials as a function of training input density c for each λ.

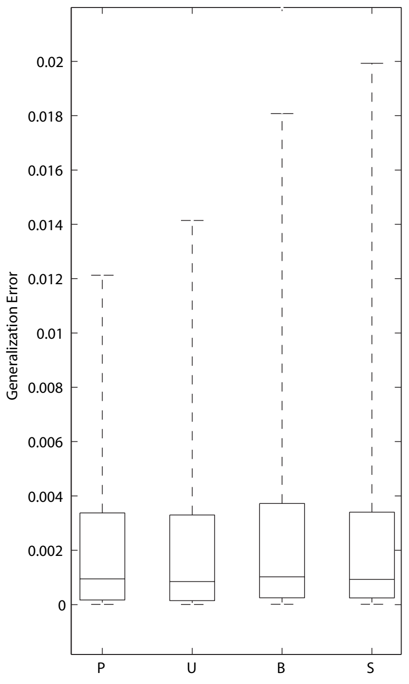

Figure 5: Generalization error of the proposed EAL (P), unnormalized EAL (U), random batch (B) and sequential (S) methods on the toy data set. Box plot's lines correspond to the 5, 25, 50, 75, and 95 percentile values from the bottom.

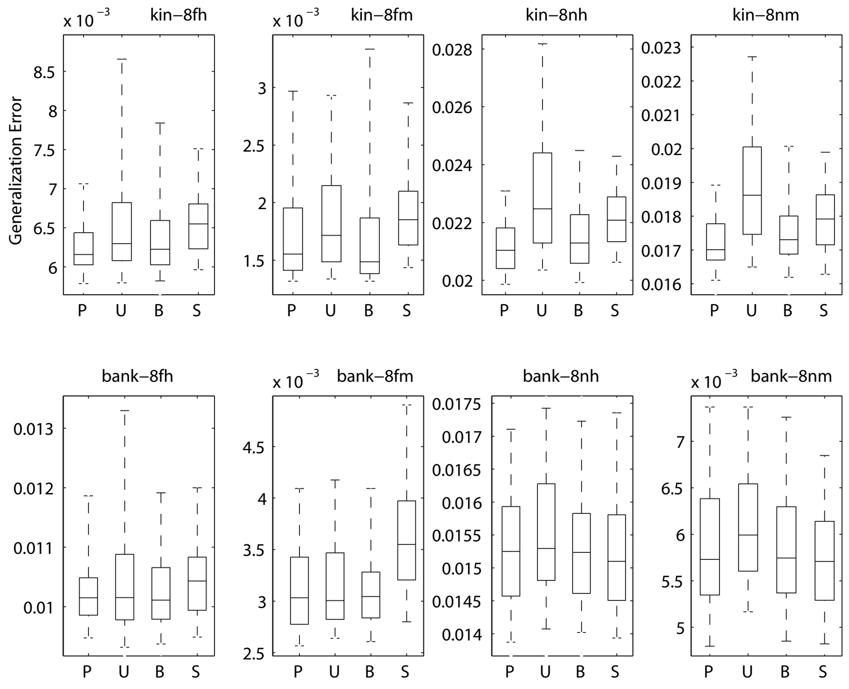

Figure 6: Generalization error of the proposed EAL (P), unnormalized EAL (U), random batch (B) and sequential (S) methods on DELVE data sets. Box plot's lines correspond to the 5, 25, 50, 75, and 95 percentile values from the bottom.

icantly improved. The good effect of the normalization can also be clearly observed in the result.

5.3 Benchmark data sets

Here we evaluate whether the advantages of the proposed method discussed in Section 4 and demonstrated on the toy data set in Section 5.2 will still be valid under more realistic settings.

For this purpose, we use eight regression benchmark data sets provided by DELVE [8]: *Bank-8fm*, *Bank-8fh*, *Bank-8nm*, *Bank-8nh*, *Kin-8fm*, *Kin-8fh*, *Kin-8nm*, and *Kin-8nh*. Each data set includes 8192 samples, consisting of 8-dimensional input points and 1-dimensional output values.

For convenience, every attribute is normalized into $[0, 1]$.

Suppose we are given all 8192 *input* points (i.e., unlabeled samples). Note that output values are kept unknown at this point. From this pool of unlabeled samples, we choose $n = 300$ input points $\{\boldsymbol{x}_i\}_{i=1}^n$ for training and observe the corresponding output values $\{y_i\}_{i=1}^n$. The task is to predict the output values of all 8192 samples.

In this experiment, the test input density $q(\boldsymbol{x})$ is unknown. So we estimate it using the uncorrelated multi-dimensional Gaussian model:

$$q(\boldsymbol{x}) = \exp\left(-\|\boldsymbol{x} - \widehat{\boldsymbol{\mu}}_{MLE}\|^2/(2\widehat{\gamma}_{MLE}^2)\right) / (2\pi\widehat{\gamma}_{MLE}^2)^{\frac{d}{2}}, \tag{11}$$

where $\widehat{\boldsymbol{\mu}}_{MLE}$ and $\widehat{\gamma}_{MLE}$ are the maximum likelihood estimates of the mean and standard deviation obtained from all 8192 unlabeled samples. We select the training input density $p(\boldsymbol{x})$ from the set of uncorrelated multi-dimensional Gaussian densities with mean $\widehat{\boldsymbol{\mu}}_{MLE}$ and standard deviation $c\widehat{\gamma}_{MLE}$, where $c = 0.7, 0.75, 0.8, \ldots, 2.4$.

Let the number of basis functions be $b = 50$ and the basis functions be Gaussian functions with standard deviation h: for $i = 1, 2, \ldots, b$,

$$\varphi_i(\boldsymbol{x}) = \exp\left(-\|\boldsymbol{x} - \boldsymbol{t}_i\|^2/(2h^2)\right), \tag{12}$$

where $\{\boldsymbol{t}_i\}_{i=1}^b$ are template points randomly chosen from the pool of unlabeled samples. In this experiment, we fix the number of basis functions and focus on choosing λ from $\{0, 0.5, 1\}$ and standard deviation h from $\{0.4, 0.6, 0.8, 1\}$.

We again compare the AL methods tested in Section 5.2. In this simulation, we can not create the training input points in an arbitrary location because we only have 8192 samples in the pool. Here, we first create provisional input points following the determined training input density, and then choose the input points from the pool of unlabeled samples that are closest to the provisional input points. In this simulation, the expectation over the test input density $q(\boldsymbol{x})$ in the matrix $\boldsymbol{U}$ is calculated by the empirical average over all 8192 unlabeled samples since the true test error is also calculated as such. For each data set, we run this simulation 100 times, by changing the template points $\{\boldsymbol{t}_i\}_{i=1}^b$ in each run.

The box plots of the obtained generalization error by each method are depicted in Figure 6. These graphs show that the proposed method in most cases outperforms other methods. It is interesting to note that the best-case performance as well as the worst-case performance is also improved. The results show that the proposed EAL approach is promising in improving the generalization performance in supervised learning.

6 Conclusion

So far, the problems of active learning (AL) and model selection (MS) have been studied as two independent problems, although they both share a common goal of minimizing the generalization error. We suggested that by simultaneously performing AL and MS—which we called *active learning with model selection* (ALMS), a better generalization capability could be achieved. We discussed the advantages and limitations of direct batch, sequential, and random batch approaches. To overcome the limitations of the above approaches, we proposed a new approach called *ensemble active learning* (EAL) which has a natural minimax property in the ALMS scenario. We have demonstrated on toy and benchmark data sets that EAL outperforms other approaches.

Acknowledgements The authors would like to thank Koji Tsuda for his fruitful comments. This work is partially supported by MEXT (Grant-in-Aid for Young Scientists 17700142 and Grant-in-Aid for Scientific Research (B) 18300057).

References

[1] H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, AC-19(6):716–723, 1974.

[2] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4:129–145, 1996.

[3] P. Craven and G. Wahba. Smoothing noisy data with spline functions: Estimating the correct degree of smoothing by the method of generalized cross-validation. *Numerische Mathematik*, 31:377–403, 1979.

[4] V. V. Fedorov. *Theory of Optimal Experiments*. Academic Press, New York, 1972.

[5] K. Fukumizu. Statistical active learning in multilayer perceptrons. *IEEE Transactions on Neural Networks*, 11(1):17–26, 2000.

[6] D. MacKay. Information-based objective functions for active data selection. *Neural Computation*, 4(4):590–604, 1992.

[7] D. J. C. MacKay. Bayesian interpolation. *Neural Computation*, 4(3):415–447, 1992.

[8] C. E. Rasmussen, R. M. Neal, G. E. Hinton, D. van Camp, M. Revow, Z. Ghahramani, R. Kustra, and R. Tibshirani. The DELVE manual, 1996. http://www.cs.toronto.edu/~delve/.

[9] J. Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465–471, 1978.

[10] G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6:461–464, 1978.

[11] H. Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90(2):227–244, 2000.

[12] M. Sugiyama. Active learning in approximately linear regression based on conditional expectation of generalization error. *Journal of Machine Learning Research*, 7(Jan):141–166, 2006.

[13] M. Sugiyama and K.-R. Müller. Input-dependent estimation of generalization error under covariate shift. *Statistics & Decisions*, 23(4):249–279, 2005.