Pool-based Agnostic Experiment Design in Linear Regression

Masashi Sugiyama[1] and Shinichi Nakajima[2]

[1] Department of Computer Science, Tokyo Institute of Technology,
2-12-1 O-okayama, Meguro-ku, Tokyo 152-8552, Japan,
sugi@cs.titech.ac.jp, http://sugiyama-www.cs.titech.ac.jp/~sugi/
[2] Nikon Corporation,
201-9 Oaza-Miizugahara, Kumagaya-shi, Saitama 360-8559, Japan
nakajima.s@nikon.co.jp

Abstract. We address the problem of batch active learning (or experiment design) in regression scenarios, where the best input points to label is chosen from a 'pool' of unlabeled input samples. Existing active learning methods often assume that the model is correctly specified, i.e., the unknown learning target function is included in the model at hand. However, this assumption may not be fulfilled in practice (i.e., agnostic) and then the existing methods do not work well. In this paper, we propose a new active learning method that is robust against model misspecification. Simulations with various benchmark datasets as well as a real application to wafer alignment in semiconductor exposure apparatus illustrate the usefulness of the proposed method.

1 Introduction

Active learning (AL) is a problem of optimally designing the location of training input points in supervised learning scenarios [1]. Choice of training input location is particularly important when the sampling cost of output values is very high, e.g., in the analysis of, medical data, biological data, or chemical data. In this paper, we address *batch AL* (a.k.a. experiment dasign), where the location of all training input points are designed in the beginning (cf. *on-line AL* where input points are chosen sequentially).

Population-based vs. Pool-based AL: Depending on the situations, AL can be categorized into two types: *population-based* and *pool-based*.

Population-based AL indicates the situation where we know the distribution of test input points and we are allowed to locate training input points at any desired positions [2–4]. The goal of population-based AL is to find the optimal training input distribution from which we generate training input points.

On the other hand, in pool-based AL, the test input distribution is unknown but samples from the test input distribution are given [5, 6]. The goal of pool-based AL is to choose the best input samples to label from the pool of test input samples. If we have infinitely many test input samples, the pool-based problem is

reduced to the population-based problem. In this paper, we address the problem of pool-based AL and propose a new algorithm.

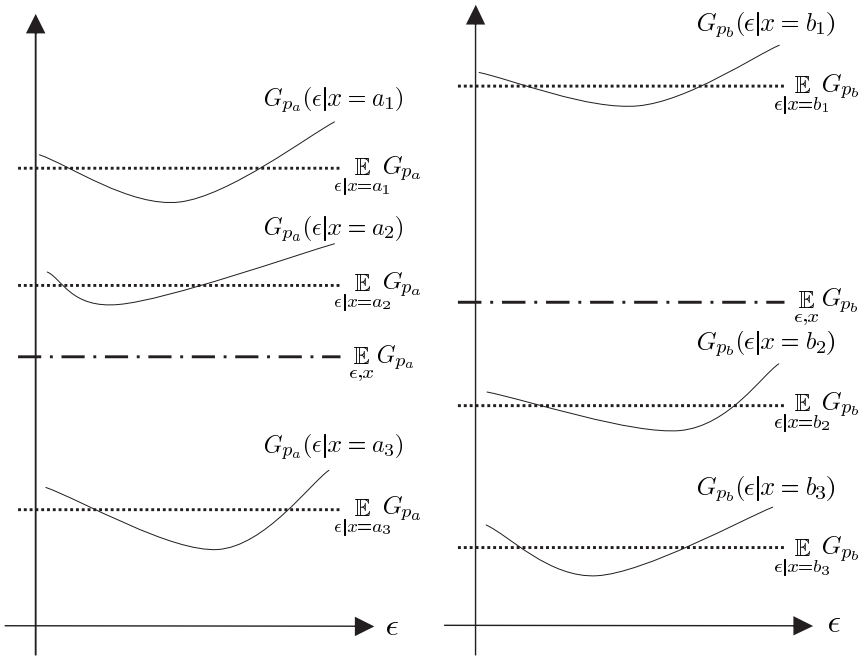
AL for Misspecified Models: In traditional AL research [1, 7, 8], it is often assumed that the model used for function learning is *correctly specified*, i.e., it can exactly realize the learning target function. However, such an assumption may not be satisfied in reality (i.e., agnostic) and the violation of this assumption can cause significant performance degradation [2–4, 6]. For this reason, we do not assume from the beginning that our model is correct in this paper. This highly enlarges the range of application of AL techniques.

In the AL scenarios, the distribution of training input points is generally different from that of test input points since the location of training input points is designed by users. Such a situation is referred to as *covariate shift* in statistics [9]. When we deal with misspecified models, covariate shift has a significant influence—for example, *Ordinary Least-Squares (OLS)* is no longer unbiased even asymptotically. Therefore, we need to explicitly take into account the bias caused by covariate shift. A standard approach to alleviating the influence of covariate shift is to use an *importance-weighting* technique [10], where the term 'importance' refers to the ratio of test and training input densities. For example, in parameter learning, OLS is biased, but *Importance-Weighted Least-Squares (IWLS)* is asymptotically unbiased [9].

Importance Estimation in Pool-based AL: In population-based AL, importance-weighting techniques can be employed for bias reduction in a straightforward manner since the test input distribution is accessible by assumption (and the training input distribution is also known since it is designed by ourselves) [2–4]. However, in pool-based AL, the test and training input distributions may both be unknown and therefore the importance weights cannot be directly computed. A naive approach to coping with this problem is to estimate the training and test input distributions from training and test input samples. However, density estimation is known to be a hard problem particularly in high dimensional problems. Therefore, such a naive approach may not be useful in practice. This difficulty could be eased by employing recently developed methods of *direct importance estimation* [11–13], which allow us to obtain the importance weight without going through density estimation. However, these methods still contain some estimation error.

A key observation in pool-based AL is that we choose training input points from the pool of test input points. This implies that our training input distribution is defined *over* the test input distribution, i.e., the training input distribution can be expressed as a product of the test input distribution and a *resampling bias function*. This decomposition allows us to directly compute the importance weight based on the resampling bias function, which is more accurate and computationally more efficient than the naive density estimation approach and the direct importance estimation approaches.

Single-trial Analysis of Generalization Error: In practice, we are only given a single realization of training samples. Therefore, ideally, we want to have an estimator of the generalization error that is accurate in each *single trial*. However,



(a) Generalization error for training input density p_a

(b) Generalization error for training input density p_b

Fig. 1. Schematic illustrations of the conditional-expectation and full-expectation of the generalization error.

we may not be able to avoid taking the expectation over the training output noise since it is not generally possible to know the realized value of noise. On the other hand, the location of the training input points is accessible by nature. Motivated by this fact, we propose to estimate the generalization error *without* taking the expectation over training input points. That is, we evaluate the unbiasedness of the generalization error in terms of the *conditional* expectation of training output noise given training input points.

To illustrate a possible advantage of this conditional expectation approach, let us consider a simple population-based active learning scenario where only one training sample (x, y) is gathered (see Figure 1). Suppose thatthe input x is drawn from a user-chosen training input distribution and y is contaminated by additive noise ϵ. The solid curves in Figure 1(a) depict $G_{p_a}(\epsilon|x)$, the generalization error for a training input density p_a as a function of the training output noise ϵ given a training input point x. The three solid curves correspond to the cases where the realizations of the training input point x are a_1, a_2, and a_3, respectively. The value of the generalization error for the training input density p_a in the full-expectation approach is depicted by the dash-dotted line, where

the generalization error is expected over both the training output noise ϵ and the training input points x (i.e., the mean of the three solid curves). The values of the generalization error in the conditional-expectation approach are depicted by the dotted lines, where the generalization errors are expected only over the training output noise ϵ, given $x = a_1, a_2, a_3$, respectively (i.e., the mean of each solid curve). The graph in Figure 1(b) depicts the generalization errors for another training input density p_b in the same manner.

In the full-expectation framework, the density p_a is judged to be better than p_b regardless of the realization of the training input point since the dash-dotted line Figure 1(a) is lower than that in Figure 1(b). However, as the solid curves show, p_a is often worse than p_b in single trials. On the other hand, in the conditional-expectation framework, the goodness of the density is adaptively judged depending on the realizations of the training input point x. For example, p_b is judged to be better than p_a if a_2 and b_3 are realized, or p_a is judged to be better than p_b if a_3 and b_1 are realized. That is, the conditional-expectation framework may provide a finer choice of the training input density (and the training input points) than the full-expectation framework.

Contributions of This Paper: We extend two population-based AL methods proposed by [2] and [4] to pool-based scenarios. The pool-based extension of the method proposed in [2] allows us to obtain a closed-form solution of the best resampling bias function; thus it is computationally very efficient. However, this method is based on the full-expectation analysis of the generalization error, so the obtained solution is not necessarily optimal in terms of the single-trial generalization error. On the other hand, the pool-based extension of the method proposed in [4] can give a better solution since it is based on the conditional-expectation analysis of the generalization error. However, it does not have a closed-form solution and therefore some additional search strategy is needed.

To cope with this problem, we propose a practical procedure by combining the above two methods—we use the analytic optimal solution of the full-expectation method for efficiently searching for a better solution in the conditional-expectation method. Extensive simulations show that the proposed AL method consistently outperforms the baseline passive learning scheme and compares favorably with other active learning methods. Finally, we apply the proposed AL method to a real-world wafer alignment problem in semiconductor exposure apparatus and show that the alignment accuracy can be improved.

2 A New Pool-based AL Method

In this section, we formulate the pool-based AL problem in regression scenarios and describe our new algorithm. Derivation and justification of the proposed algorithm are given in the next section.

2.1 Formulation of Pool-based AL in Regression

We address a regression problem of learning a real-valued function $f(\boldsymbol{x})$ defined on $\mathcal{D} \subset \mathbb{R}^d$. We are given a 'pool' of test *input* points $\{\boldsymbol{x}_j^{\mathrm{te}}\}_{j=1}^{n_{\mathrm{te}}}$, which are drawn

independently from an *unknown* test input distribution with strictly positive density $p_{\mathrm{te}}(\boldsymbol{x})$. From the pool, we are allowed to choose n_{tr} ($\ll n_{\mathrm{te}}$) input points for observing output values. Let $\{\boldsymbol{x}_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$ be input points selected from the pool and $\{y_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$ be corresponding output values, which we call *training samples*:

$$\{(\boldsymbol{x}_i^{\mathrm{tr}}, y_i^{\mathrm{tr}}) \mid y_i^{\mathrm{tr}} = f(\boldsymbol{x}_i^{\mathrm{tr}}) + \epsilon_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}},$$

where $\{\epsilon_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$ are i.i.d. noise with mean zero and unknown variance σ^2.

The goal of the regression task is to accurately predict the output values $\{f(\boldsymbol{x}_j^{\mathrm{te}})\}_{j=1}^{n_{\mathrm{te}}}$ at all test input points[3] $\{\boldsymbol{x}_j^{\mathrm{te}}\}_{j=1}^{n_{\mathrm{te}}}$. We adopt the squared loss as our error metric:

$$\frac{1}{n_{\mathrm{te}}} \sum_{j=1}^{n_{\mathrm{te}}} \left(\widehat{f}(\boldsymbol{x}_j^{\mathrm{te}}) - f(\boldsymbol{x}_j^{\mathrm{te}})\right)^2, \tag{1}$$

where $\widehat{f}(\boldsymbol{x})$ is a function learned from the training samples $\{(\boldsymbol{x}_i^{\mathrm{tr}}, y_i^{\mathrm{tr}})\}_{i=1}^{n_{\mathrm{tr}}}$.

2.2 Weighted Least-squares for Linear Regression Models

We use the following linear regression model for learning:

$$\widehat{f}(\boldsymbol{x}) = \sum_{\ell=1}^{t} \theta_\ell \varphi_\ell(\boldsymbol{x}), \tag{2}$$

where $\{\varphi_\ell(\boldsymbol{x})\}_{\ell=1}^{t}$ are fixed linearly independent basis functions. $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_t)^\top$ are parameters to be learned, where $^\top$ denotes the transpose of a vector or a matrix.

We learn the parameter $\boldsymbol{\theta}$ of the regression model by *Weighted Least-Squares (WLS)* with a weight function $w(\boldsymbol{x})$ (> 0 for all $\boldsymbol{x} \in \mathcal{D}$), i.e.,

$$\widehat{\boldsymbol{\theta}}_{\mathrm{W}} = \operatorname*{argmin}_{\boldsymbol{\theta}} \left[\sum_{i=1}^{n_{\mathrm{tr}}} w(\boldsymbol{x}_i^{\mathrm{tr}}) \left(\widehat{f}(\boldsymbol{x}_i^{\mathrm{tr}}) - y_i^{\mathrm{tr}}\right)^2\right], \tag{3}$$

where the subscript 'W' denotes 'Weighted'. Let $\boldsymbol{X}$ be the $n_{\mathrm{tr}} \times t$ matrix with $X_{i,\ell} = \varphi_\ell(\boldsymbol{x}_i^{\mathrm{tr}})$, and let $\boldsymbol{W}$ be the $n_{\mathrm{tr}} \times n_{\mathrm{tr}}$ diagonal matrix with $W_{i,i} = w(\boldsymbol{x}_i^{\mathrm{tr}})$. Then $\widehat{\boldsymbol{\theta}}_{\mathrm{W}}$ is given in a closed-form as

$$\widehat{\boldsymbol{\theta}}_{\mathrm{W}} = \boldsymbol{L}_{\mathrm{W}} \boldsymbol{y}^{\mathrm{tr}}, \tag{4}$$

where

$$\boldsymbol{L}_{\mathrm{W}} = (\boldsymbol{X}^\top \boldsymbol{W} \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{W},$$
$$\boldsymbol{y}^{\mathrm{tr}} = (y_1^{\mathrm{tr}}, y_2^{\mathrm{tr}}, \ldots, y_{n_{\mathrm{tr}}}^{\mathrm{tr}})^\top.$$

[3] Under the assumption that $n_{\mathrm{tr}} \ll n_{\mathrm{te}}$, the difference between the prediction error at all test input points $\{\boldsymbol{x}_j^{\mathrm{te}}\}_{j=1}^{n_{\mathrm{te}}}$ and the remaining test input points $\{\boldsymbol{x}_j^{\mathrm{te}}\}_{j=1}^{n_{\mathrm{te}}} \backslash \{\boldsymbol{x}_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$ is negligibly small. More specifically, if $n_{\mathrm{tr}} = o(\sqrt{n_{\mathrm{te}}})$, all the discussions in this paper is still valid even when the prediction error is evaluated only at the remaining test input points.

2.3 Proposed AL Algorithm: P-CV$_{\mathrm{W}}$

Here we describe our AL algorithm for choosing the training input points from the pool of test input points; its derivation and justification are provided in the next section.

First, we prepare a candidate set of training input points $\{\boldsymbol{x}_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$, which is a subset of $\{\boldsymbol{x}_j^{\mathrm{te}}\}_{j=1}^{n_{\mathrm{te}}}$. More specifically, we prepare a *resampling bias function* $b(\boldsymbol{x})$ (> 0 for all $\boldsymbol{x} \in \mathcal{D}$) and choose n_{tr} training input points from the pool of test input points $\{\boldsymbol{x}_j^{\mathrm{te}}\}_{j=1}^{n_{\mathrm{te}}}$ with probability proportional to

$$\{b(\boldsymbol{x}_j^{\mathrm{te}})\}_{j=1}^{n_{\mathrm{te}}}.$$

Later, we explain how we prepare a family of useful resampling bias functions. We evaluate the 'quality' of the candidate training input points $\{\boldsymbol{x}_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$ by

$$\text{P-CV}_{\mathrm{W}} = \mathrm{tr}(\widehat{\boldsymbol{U}}\boldsymbol{L}_{\mathrm{W}}\boldsymbol{L}_{\mathrm{W}}^{\top}), \tag{5}$$

where the weight function $w(\boldsymbol{x})$ included in $\boldsymbol{L}_{\mathrm{W}}$ is defined as

$$w(\boldsymbol{x}_j^{\mathrm{te}}) = b(\boldsymbol{x}_j^{\mathrm{te}})^{-1}.$$

$\widehat{\boldsymbol{U}}$ is the $t \times t$ matrix with

$$\widehat{U}_{\ell,\ell'} = \frac{1}{n_{\mathrm{te}}} \sum_{j=1}^{n_{\mathrm{te}}} \varphi_\ell(\boldsymbol{x}_j^{\mathrm{te}})\varphi_{\ell'}(\boldsymbol{x}_j^{\mathrm{te}}).$$

We call the above criterion *pool-based CV_W (P-CV_W)*, which is a pool-based extension of a population-based AL criterion *CV_W (Conditional Variance of WLS)* [4]; we will explain the meaning and derivation of P-CV$_{\mathrm{W}}$ in Section 3.

We repeat the above evaluation for each resampling bias function in our candidate set and choose the best one with the smallest P-CV$_{\mathrm{W}}$ score. Once the resampling bias function and the training input points are chosen, we gather training output values $\{y_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$ at the chosen location and train a linear regression model (2) using WLS with the chosen weight function.

In the above procedure, the choice of the candidates of the resampling bias function $b(\boldsymbol{x})$ is arbitrary. As a heuristic, we propose using the following family of resampling bias functions parameterized by a scalar γ:

$$b_\gamma(\boldsymbol{x}) = \left(\sum_{\ell,\ell'=1}^{t} [\widehat{\boldsymbol{U}}^{-1}]_{\ell,\ell'} \varphi_\ell(\boldsymbol{x})\varphi_{\ell'}(\boldsymbol{x}) \right)^{\gamma}. \tag{6}$$

The parameter γ controls the 'shape' of the training input distribution—when $\gamma = 0$, the resampling weight is uniform over all test input samples. Thus the above choice includes *passive learning* (the training and test distributions are equivalent) as a special case. We seek the best γ by simple multi-point search, i.e., we compute the value of P-CV$_{\mathrm{W}}$ for several different values of γ and choose the minimizer. In practice, we propose performing the search intensively around $\gamma = 1/2$, e.g., Eq.(13); the reason for this will be explained in the next section.

A pseudo code of the proposed pool-based AL algorithm is described in Figure 2.

Input: Test input points $\{\boldsymbol{x}_j^{\mathrm{te}}\}_{j=1}^{n_{\mathrm{te}}}$ and basis functions $\{\varphi_\ell(\boldsymbol{x})\}_{\ell=1}^{t}$
Output: Learned parameter $\widehat{\boldsymbol{\theta}}_{\mathrm{W}}$

Compute the $t \times t$ matrix $\widehat{\boldsymbol{U}}$ with $\widehat{U}_{\ell,\ell'} = \frac{1}{n_{\mathrm{te}}} \sum_{j=1}^{n_{\mathrm{te}}} \varphi_\ell(\boldsymbol{x}_j^{\mathrm{te}}) \varphi_{\ell'}(\boldsymbol{x}_j^{\mathrm{te}})$;
For several different values of γ (intensively around $\gamma = 1/2$)
  Compute $\{b_\gamma(\boldsymbol{x}_j^{\mathrm{te}})\}_{j=1}^{n_{\mathrm{te}}}$ with $b_\gamma(\boldsymbol{x}) = \left(\sum_{\ell,\ell'=1}^{t} [\widehat{\boldsymbol{U}}^{-1}]_{\ell,\ell'} \varphi_\ell(\boldsymbol{x}) \varphi_{\ell'}(\boldsymbol{x}) \right)^{\gamma}$;
  Choose $\mathcal{X}_\gamma^{\mathrm{tr}} = \{\boldsymbol{x}_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$ from $\{\boldsymbol{x}_j^{\mathrm{te}}\}_{j=1}^{n_{\mathrm{te}}}$ with probability proportional to $\{b_\gamma(\boldsymbol{x}_j^{\mathrm{te}})\}_{j=1}^{n_{\mathrm{te}}}$;
  Compute the $n_{\mathrm{tr}} \times t$ matrix $\boldsymbol{X}_\gamma$ with $[X_\gamma]_{i,\ell} = \varphi_\ell(\boldsymbol{x}_i^{\mathrm{tr}})$;
  Compute the $n_{\mathrm{tr}} \times n_{\mathrm{tr}}$ diagonal matrix $\boldsymbol{W}_\gamma$ with $[W_\gamma]_{i,i} = b_\gamma(\boldsymbol{x}_i^{\mathrm{tr}})^{-1}$;
  Compute $\boldsymbol{L}_\gamma = (\boldsymbol{X}_\gamma^\top \boldsymbol{W}_\gamma \boldsymbol{X}_\gamma)^{-1} \boldsymbol{X}_\gamma^\top \boldsymbol{W}_\gamma$;
  Compute $\text{P-CV}_{\mathrm{W}}(\gamma) = \mathrm{tr}(\widehat{\boldsymbol{U}} \boldsymbol{L}_\gamma \boldsymbol{L}_\gamma^\top)$;
End
Compute $\widehat{\gamma} = \mathrm{argmin}_\gamma \, \text{P-CV}_{\mathrm{W}}(\gamma)$;
Gather training output values $\{y_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$ at $\mathcal{X}_{\widehat{\gamma}}^{\mathrm{tr}}$;
Compute $\widehat{\boldsymbol{\theta}}_{\mathrm{W}} = \boldsymbol{L}_{\widehat{\gamma}} (y_1^{\mathrm{tr}}, y_2^{\mathrm{tr}}, \ldots, y_{n_{\mathrm{tr}}}^{\mathrm{tr}})^\top$;

Fig. 2. Pseudo code of proposed pool-based AL algorithm. In practice, the best γ may be intensively searched around $\gamma = 1/2$.

3 Derivation and Justification of Proposed AL Algorithm

The proposed P-CV_{W} criterion (5) and our choice of candidates of the training input distribution (6) are motivated by population-based AL criteria called CV_{W} (Conditional Variance of WLS; [4]) and FV_{W} (Full Variance of WLS; [2]). In this section, we explain how we came up with the pool-based AL algorithm given in Section 2.

3.1 Population-based AL Criterion: CV_{W}

Here we review a population-based AL criterion CV_{W}.

In the population-based framework, we are given the test input density $p_{\mathrm{te}}(\boldsymbol{x})$, and the goal is to determine the best training input density $p_{\mathrm{tr}}(\boldsymbol{x})$ from which we draw training input points $\{\boldsymbol{x}_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$ [8, 2–4].

The aim of the regression task in the population-based framework is to accurately predict the output values for all test input samples drawn from $p_{\mathrm{te}}(\boldsymbol{x})$. Thus the error metric (often called the *generalization error*) is

$$G' = \int \left(\widehat{f}(\boldsymbol{x}^{\mathrm{te}}) - f(\boldsymbol{x}^{\mathrm{te}}) \right)^2 p_{\mathrm{te}}(\boldsymbol{x}^{\mathrm{te}}) d\boldsymbol{x}^{\mathrm{te}} \equiv \|\widehat{f} - f\|_{p_{\mathrm{te}}}^2.$$

Suppose the regression model (2) *approximately* includes the learning target function $f(\boldsymbol{x})$, i.e., for a scalar δ such that $|\delta|$ is small, $f(\boldsymbol{x})$ is expressed as

$$f(\boldsymbol{x}) = g(\boldsymbol{x}) + \delta r(\boldsymbol{x}). \tag{7}$$

In the above, $g(\boldsymbol{x})$ is the optimal approximation to $f(\boldsymbol{x})$ by the model (2):

$$g(\boldsymbol{x}) = \textstyle\sum_{\ell=1}^{t} \theta_\ell^* \varphi_\ell(\boldsymbol{x}),$$

where $\boldsymbol{\theta}^* = (\theta_1^*, \theta_2^*, \ldots, \theta_t^*)^\top = \mathrm{argmin}_{\boldsymbol{\theta}} G'$ is the unknown optimal parameter. $\delta r(\boldsymbol{x})$ in Eq.(7) is the residual function, which is orthogonal to $\{\varphi_\ell(\boldsymbol{x})\}_{\ell=1}^t$ under $p_{\mathrm{te}}(\boldsymbol{x})$, i.e., $\langle r, \varphi_\ell \rangle_{p_{\mathrm{te}}} = 0$ for $\ell = 1, 2, \ldots, t$. The function $r(\boldsymbol{x})$ governs the nature of the model error, while δ is the possible magnitude of this error. In order to separate these two factors, we further impose $\|r\|_{p_{\mathrm{te}}} = 1$.

Let $\mathbb{E}_{\boldsymbol{\epsilon}}$ be the expectation over the noise $\{\epsilon_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$. Then, the generalization error expected over the training output noise can be decomposed into the (squared) *bias* term B, the *variance* term V, and the model error δ^2:

$$\mathbb{E}_{\boldsymbol{\epsilon}} G' = B + V + \delta^2,$$

where

$$B = \|\mathbb{E}_{\boldsymbol{\epsilon}} \widehat{f} - g\|_{p_{\mathrm{te}}}^2, \quad V = \mathbb{E}_{\boldsymbol{\epsilon}} \|\widehat{f} - \mathbb{E}_{\boldsymbol{\epsilon}} \widehat{f}\|_{p_{\mathrm{te}}}^2.$$

Since δ is constant which depends neither on $p_{\mathrm{tr}}(\boldsymbol{x})$ nor $\{\boldsymbol{x}_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$, we subtract δ^2 from G' and define it by G.

$$G = G' - \delta^2.$$

Here we use *Importance-Weighted Least-Squares (IWLS)* for parameter learning [9], i.e., Eq.(3) with weight function $w(\boldsymbol{x})$ being the ratio of densities called the *importance ratio*:

$$w(\boldsymbol{x}) = \frac{p_{\mathrm{te}}(\boldsymbol{x})}{p_{\mathrm{tr}}(\boldsymbol{x})}. \tag{8}$$

The solution $\widehat{\boldsymbol{\theta}}_{\mathrm{W}}$ is given by Eq.(4).

Let G_{W}, B_{W}, and V_{W} be G, B, and V for the learned function obtained by IWLS, respectively. Let $\boldsymbol{U}$ be the $t \times t$ matrix with

$$U_{\ell,\ell'} = \int \varphi_\ell(\boldsymbol{x}^{\mathrm{te}}) \varphi_{\ell'}(\boldsymbol{x}^{\mathrm{te}}) p_{\mathrm{te}}(\boldsymbol{x}^{\mathrm{te}}) d\boldsymbol{x}^{\mathrm{te}}.$$

Then, for IWLS with an approximately correct model, B_{W} and V_{W} are expressed as follows [4]:

$$B_{\mathrm{W}} = \mathcal{O}_p(\delta^2 n_{\mathrm{tr}}^{-1}), \quad V_{\mathrm{W}} = \sigma^2 \mathrm{tr}(\boldsymbol{U} \boldsymbol{L}_{\mathrm{W}} \boldsymbol{L}_{\mathrm{W}}^\top) = \mathcal{O}_p(n_{\mathrm{tr}}^{-1}).$$

The above equations imply that if $\delta = o_p(1)$,

$$\mathbb{E}_{\boldsymbol{\epsilon}} G_{\mathrm{W}} = \sigma^2 \mathrm{tr}(\boldsymbol{U} \boldsymbol{L}_{\mathrm{W}} \boldsymbol{L}_{\mathrm{W}}^\top) + o_p(n_{\mathrm{tr}}^{-1}).$$

The AL criterion CV_{W} is motivated by this asymptotic form, i.e., CV_{W} chooses the training input density $p_{\mathrm{tr}}(\boldsymbol{x})$ from the set $\mathcal{P}$ of all strictly positive probability densities as

$$p_{\mathrm{tr}}^{\mathrm{CV_W}} = \operatorname*{argmin}_{p_{\mathrm{tr}} \in \mathcal{P}} \mathrm{CV}_{\mathrm{W}}, \quad \mathrm{CV}_{\mathrm{W}} = \mathrm{tr}(\boldsymbol{U} \boldsymbol{L}_{\mathrm{W}} \boldsymbol{L}_{\mathrm{W}}^\top).$$

Practically, $\mathcal{P}$ may be replaced by a finite set $\widehat{\mathcal{P}}$ of strictly positive probability densities and choose the one that minimizes CV_{W} from the set $\widehat{\mathcal{P}}$.

3.2 Extension of $\mathrm{CV_W}$ to Pool-based Scenarios: P-$\mathrm{CV_W}$

Our basic idea of P-$\mathrm{CV_W}$ is to extend $\mathrm{CV_W}$ to the pool-based scenario, where we do not know $p_{\mathrm{te}}(\boldsymbol{x})$, but we are given a pool of test input samples $\{\boldsymbol{x}_i^{\mathrm{te}}\}_{i=1}^{n_{\mathrm{te}}}$ drawn independently from $p_{\mathrm{te}}(\boldsymbol{x})$. Under the pool-based setting, the following two quantities included in $\mathrm{CV_W}$ are not accessible:

(A) The expectation over $p_{\mathrm{te}}(\boldsymbol{x})$ in $\boldsymbol{U}$,
(B) The importance ratio $p_{\mathrm{te}}(\boldsymbol{x})/p_{\mathrm{tr}}(\boldsymbol{x})$ at training input points $\{\boldsymbol{x}_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$ in $\boldsymbol{L}_{\mathrm{W}}$.

Regarding (A), we may simply approximate the expectation over $p_{\mathrm{te}}(\boldsymbol{x})$ by the empirical average over the test input samples $\{\boldsymbol{x}_i^{\mathrm{te}}\}_{i=1}^{n_{\mathrm{te}}}$, which is known to be *consistent*.

On the other hand, approximation regarding (B) can be addressed as follows. In pool-based AL, we choose training input points from the pool of test input points following a resampling bias function $b(\boldsymbol{x})$. This implies that our training input distribution is defined *over* the test input distribution, i.e., the training input distribution is expressed as a product of the test input distribution and a resampling bias function $b(\boldsymbol{x})$:

$$p_{\mathrm{tr}}(\boldsymbol{x}_j^{\mathrm{te}}) \propto p_{\mathrm{te}}(\boldsymbol{x}_j^{\mathrm{te}}) b(\boldsymbol{x}_j^{\mathrm{te}}). \tag{9}$$

This immediately shows that the importance weight $w(\boldsymbol{x}_j^{\mathrm{te}})$ is given by

$$w(\boldsymbol{x}_j^{\mathrm{te}}) \propto b(\boldsymbol{x}_j^{\mathrm{te}})^{-1}. \tag{10}$$

Note that the scaling factor of $w(\boldsymbol{x})$ is irrelevant in IWLS (cf. Eq.(3)), so the above proportional form is sufficient here. By this, we can avoid density estimation which is known to be very hard.

Summarizing the above results, we obtain the P-$\mathrm{CV_W}$ criterion (5).

3.3 Population-based AL Criterion: $\mathrm{FV_W}$

Next, we show how we came up with the candidate set of resampling bias functions given in Eq.(6). Our choice is based on a population-based AL method proposed by [2]. First, we consider the population-based setting and briefly review this method.

For IWLS, [3] proved that the generalization error expected over training input points $\{\boldsymbol{x}_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$ and training output noise $\{\epsilon_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$ is asymptotically expressed as

$$\mathbb{E}_{\boldsymbol{x}}\mathbb{E}_{\boldsymbol{\epsilon}} G_{\mathrm{W}} = \frac{\mathrm{tr}(\boldsymbol{U}^{-1}(\boldsymbol{S} + \sigma^2 \boldsymbol{T}))}{n_{\mathrm{tr}}} + \mathcal{O}({n_{\mathrm{tr}}}^{-\frac{3}{2}}), \tag{11}$$

where $\mathbb{E}_{\boldsymbol{x}}$ is the expectation over training input points $\{\boldsymbol{x}_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$. $\boldsymbol{S}$ and $\boldsymbol{T}$ are the $t \times t$ matrices with

$$S_{\ell,\ell'} = \delta^2 \int \varphi_\ell(\boldsymbol{x}) \varphi_{\ell'}(\boldsymbol{x}) \left(r(\boldsymbol{x})\right)^2 \frac{p_{\mathrm{te}}(\boldsymbol{x})^2}{p_{\mathrm{tr}}(\boldsymbol{x})} d\boldsymbol{x},$$

$$T_{\ell,\ell'} = \int \varphi_\ell(\boldsymbol{x}) \varphi_{\ell'}(\boldsymbol{x}) \frac{p_{\mathrm{te}}(\boldsymbol{x})^2}{p_{\mathrm{tr}}(\boldsymbol{x})} d\boldsymbol{x}.$$

Note that $\frac{1}{n_{\mathrm{tr}}}\mathrm{tr}(\boldsymbol{U}^{-1}\boldsymbol{S})$ corresponds to the squared bias while $\frac{\sigma^2}{n_{\mathrm{tr}}}\mathrm{tr}(\boldsymbol{U}^{-1}\boldsymbol{T})$ corresponds to the variance.

It can be shown [3, 4] that if $\delta = o(1)$,

$$\mathbb{E}_{\boldsymbol{x}}\mathbb{E}_{\boldsymbol{\epsilon}}G_{\mathrm{W}} = \frac{\sigma^2}{n_{\mathrm{tr}}}\mathrm{tr}(\boldsymbol{U}^{-1}\boldsymbol{T}) + o({n_{\mathrm{tr}}}^{-1}).$$

Based on this asymptotic form, a population-based AL criterion, which we refer to as *FV_W (Full Variance of WLS)*, is given as follows [2]:

$$p_{\mathrm{tr}}^{\mathrm{FV_W}} = \operatorname*{argmin}_{p_{\mathrm{tr}}\in\mathcal{P}} \mathrm{FV}_{\mathrm{W}}, \quad \mathrm{FV}_{\mathrm{W}} = \frac{1}{n_{\mathrm{tr}}}\mathrm{tr}(\boldsymbol{U}^{-1}\boldsymbol{T}).$$

A notable feature of FV_{W} is that the optimal training input density $p_{\mathrm{tr}}^{\mathrm{FV_W}}(\boldsymbol{x})$ can be obtained in a closed-form [2]:

$$p_{\mathrm{tr}}^{\mathrm{FV_W}}(\boldsymbol{x}) \propto p_{\mathrm{te}}(\boldsymbol{x})b_{\mathrm{FV_W}}(\boldsymbol{x}), \quad b_{\mathrm{FV_W}}(\boldsymbol{x}) = \sqrt{\sum_{\ell,\ell'=1}^{t}[\boldsymbol{U}^{-1}]_{\ell,\ell'}\varphi_\ell(\boldsymbol{x})\varphi_{\ell'}(\boldsymbol{x})}.$$

Note that the importance ratio for the optimal training input density $p_{\mathrm{tr}}^{\mathrm{FV_W}}(\boldsymbol{x})$ is given by

$$w_{\mathrm{FV_W}}(\boldsymbol{x}) \propto b_{\mathrm{FV_W}}(\boldsymbol{x})^{-1}.$$

3.4 Extension of $\mathrm{FV_W}$ to Pool-based Scenarios: P-$\mathrm{FV_W}$

If the values of the function $b_{\mathrm{FV_W}}(\boldsymbol{x})$ at the test input points $\{\boldsymbol{x}_j^{\mathrm{te}}\}_{j=1}^{n_{\mathrm{te}}}$ are available, they can be used as a resampling bias function in pool-based AL. However, since $\boldsymbol{U}$ is unknown in the pool-based scenario, it is not possible to directly compute the values of $b_{\mathrm{FV_W}}(\boldsymbol{x})$ at the test input points $\{\boldsymbol{x}_j^{\mathrm{te}}\}_{j=1}^{n_{\mathrm{te}}}$. To cope with this problem, we propose simply replacing $\boldsymbol{U}$ with an empirical estimate $\widehat{\boldsymbol{U}}$. Then, the resampling bias function $\{b_{\text{P-FV}_{\mathrm{W}}}(\boldsymbol{x}_j^{\mathrm{te}})\}_{j=1}^{n_{\mathrm{te}}}$ is given by

$$b_{\text{P-FV}_{\mathrm{W}}}(\boldsymbol{x}_j^{\mathrm{te}}) = \sqrt{\sum_{\ell,\ell'=1}^{t}[\widehat{\boldsymbol{U}}^{-1}]_{\ell,\ell'}\varphi_\ell(\boldsymbol{x}_j^{\mathrm{te}})\varphi_{\ell'}(\boldsymbol{x}_j^{\mathrm{te}})}. \tag{12}$$

The importance weight is simply given by

$$w_{\text{P-FV}_{\mathrm{W}}}(\boldsymbol{x}_j^{\mathrm{te}}) \propto b_{\text{P-FV}_{\mathrm{W}}}(\boldsymbol{x}_j^{\mathrm{te}})^{-1}.$$

3.5 Combining P-$\mathrm{CV_W}$ and P-$\mathrm{FV_W}$

It was shown that P-$\mathrm{FV_W}$ has a closed-form solution of the optimal resampling bias function. This simply suggests using $b_{\text{P-FV}_{\mathrm{W}}}(\boldsymbol{x}_j^{\mathrm{te}})$ for AL. Nevertheless, we argue that it is possible to further improve the solution.

The point of our argument is the way the generalization error is analyzed—the optimality of $\mathrm{FV_W}$ is in terms of the expectation over *both* training input points $\{\boldsymbol{x}_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$ and training output noise $\{\epsilon_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$, while $\mathrm{CV_W}$ is optimal in terms of the *conditional* expectation over training output noise $\{\epsilon_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$ given $\{\boldsymbol{x}_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$. However, in reality, what we really want to evaluate is the *single-trial* generalization error (i.e., without any expectation; both $\{\boldsymbol{x}_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$ and $\{\epsilon_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$ are given and fixed). Unfortunately, it is not possible to directly evaluate the single-trial generalization error since the training output noise $\{\epsilon_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$ cannot be observed directly; on the other hand, the training input points $\{\boldsymbol{x}_i^{\mathrm{tr}}\}_{i=1}^{n_{\mathrm{tr}}}$ are available. It was shown that the conditional expectation approach is provably more accurate in the single-trial analysis than the full expectation approach: if $\delta = o_p(n_{\mathrm{tr}}^{-1/4})$ and terms of $o_p(n_{\mathrm{tr}}^{-3})$ are ignored, the following inequality holds [4]:

$$\mathbb{E}_{\boldsymbol{\epsilon}}(\sigma^2\mathrm{FV_W} - G_{\mathrm{W}})^2 \geq \mathbb{E}_{\boldsymbol{\epsilon}}(\sigma^2\mathrm{CV_W} - G_{\mathrm{W}})^2.$$

This implies that $\sigma^2\mathrm{CV_W}$ is asymptotically a more accurate estimator of the single-trial generalization error G_{W} than $\sigma^2\mathrm{FV_W}$.

This analysis suggests that using $\mathrm{P\text{-}CV_W}$ is more suitable than $\mathrm{P\text{-}FV_W}$. However, a drawback of $\mathrm{P\text{-}CV_W}$ is that a closed-form solution is not available—thus, we may practically need to prepare candidates of training input samples and search for the best solution from the candidates. To ease this problem, our heuristic is to use the closed-form solution of $\mathrm{P\text{-}FV_W}$ as a 'base' candidate and search around its vicinity. More specifically, we consider a family of resampling bias functions (6), which is parameterized by γ. This family consists of the optimal solution of $\mathrm{P\text{-}FV_W}$ ($\gamma = 1/2$) and its variants ($\gamma \neq 1/2$); passive learning is also included as a special case ($\gamma = 0$) in this family.

The experimental results in Section 4 show that an additional search using $\mathrm{P\text{-}CV_W}$ tends to significantly improve the AL performance over $\mathrm{P\text{-}FV_W}$.

4 Simulations

In this section, we quantitatively compare the proposed and existing AL methods through numerical experiments.

4.1 Toy Dataset

We first illustrate how the proposed and existing AL methods behave under a controlled setting.

Let the input dimension be $d = 1$ and let the learning target function be

$$f(x) = 1 - x + x^2 + \delta r(x),$$

where $r(x) = (z^3 - 3z)/\sqrt{6}$ with $z = (x - 0.2)/0.4$. $r(x)$ defined here is a third order polynomial and is chosen to satisfy $\langle r, \varphi_\ell \rangle_{p_{\mathrm{te}}} = 0$ and $\|r\|_{p_{\mathrm{te}}} = 1$. Let us consider three cases $\delta = 0, 0.03, 0.06$.

Let the number of training examples to gather be $n_{\rm tr} = 100$ and let $\{\epsilon_i^{\rm tr}\}_{i=1}^{n_{\rm tr}}$ be i.i.d. Gaussian noise with mean zero and standard deviation $\sigma = 0.3$, where σ is treated as unknown here. Let the test input density $p_{\rm te}(x)$ be Gaussian with mean 0.2 and standard deviation 0.4; $p_{\rm te}(x)$ is also treated as unknown here. We draw $n_{\rm te} = 1000$ test input points independently from the test input distribution.

We use a polynomial model of order 2 for learning:

$$\widehat{f}(x) = \theta_1 + \theta_2 x + \theta_3 x^2.$$

We compare the performance of the following sampling strategies:

(A) P-CV$_{\rm W}$: We draw training input points following the resampling bias function (6) with

$$\gamma \in \{0, 0.1, 0.2, \ldots, 1\} \cup \{0.4, 0.41, 0.42, \ldots, 0.6\}. \tag{13}$$

Then we choose the best γ from the above candidates based on P-CV$_{\rm W}$ (5). IWLS is used for parameter learning.

(B) P-FV$_{\rm W}$: We draw training input points following the resampling bias function (12). IWLS is used for parameter learning.

(C) Q-OPT [1, 7, 8]: We draw training input points following the resampling bias function (6) with Eq.(13), and choose the best γ based on

$$\text{Q-OPT} = \mathrm{tr}(\widehat{\boldsymbol{U}}\boldsymbol{L}_{\rm O}\boldsymbol{L}_{\rm O}^\top),$$

where $\boldsymbol{L}_{\rm O} = (\boldsymbol{X}^\top\boldsymbol{X})^{-1}\boldsymbol{X}^\top$. OLS is used for parameter learning.

(D) Passive: We draw training input points uniformly from the pool of test input samples. OLS is used for parameter learning.

In Table 1, the mean squared test error (1) obtained by each method is described. The numbers in the table are means and standard deviations over 100 trials.

When $\delta = 0$, Q-OPT and P-CV$_{\rm W}$ are comparable to each other and are better than P-FV$_{\rm W}$ and Passive. When $\delta = 0.03$, the performance of P-CV$_{\rm W}$ and P-FV$_{\rm W}$ is almost unchanged, while the performance of Q-OPT is degraded significantly. Consequently, P-CV$_{\rm W}$ gives the best performance among all. When $\delta = 0.06$, the performance of P-CV$_{\rm W}$ and P-FV$_{\rm W}$ are still almost unchanged, while Q-OPT performs very poorly and is outperformed even by the baseline Passive method.

The above results show that P-CV$_{\rm W}$ and P-FV$_{\rm W}$ are highly robust against model misspecification, while Q-OPT is very sensitive to the violation of the model correctness assumption. P-CV$_{\rm W}$ tends to outperform P-FV$_{\rm W}$, which would be caused by the fact that CV$_{\rm W}$ is a more accurate estimator of the single-trial generalization error than FV$_{\rm W}$.

4.2 Benchmark Datasets

Here we use the *Bank*, *Kin*, and *Pumadyn* regression benchmark data families provided by DELVE [14]. Each data family consists of 8 different datasets: The

input dimension is either $d = 8$ or 32, the target function is either 'fairly linear' or 'non-linear' ('f' or 'n'), and the unpredictability/noise level is either 'medium' or 'high' ('m' or 'h'). Thus we use 24 datasets in total. Each dataset includes 8192 samples, consisting of d-dimensional input and 1-dimensional output data. For convenience, we normalize every attribute into $[0, 1]$.

We use all 8192 input samples as the pool of test input points (i.e., $n_{\mathrm{te}} = 8192$), and choose $n_{\mathrm{tr}} = 100$ training input points from the pool when $d = 8$ and $n_{\mathrm{tr}} = 300$ training input points when $d = 32$. We use the following linear regression model:

$$\widehat{f}(\boldsymbol{x}) = \sum_{\ell=1}^{50} \theta_\ell \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{c}_\ell\|^2}{2}\right),$$

where $\{\boldsymbol{c}_\ell\}_{\ell=1}^{50}$ are template points randomly chosen from the pool of test input points. Other settings are the same as the toy experiments in Section 4.1.

Table 2 summarizes the mean squared test error (1) over 1000 trials, where all the values are normalized by the mean error of the Passive method.

When $d = 8$, all 3 AL methods tend to be better than the Passive method. Among them, P-CV_{W} significantly outperforms P-FV_{W} and Q-OPT. When $d = 32$, Q-OPT outperforms P-CV_{W} and P-FV_{W} for several datasets. However, the performance of Q-OPT is highly unstable and is very poor for the *kin-32fm*, *kin-32fh*, and *pumadyn-32fm* datasets. Consequently, the average error of Q-OPT over all 12 datasets is worse than the baseline Passive method. On the other hand, P-CV_{W} and P-FV_{W} are still stable and consistently outperform the Passive method. Among these two methods, P-CV_{W} significantly outperforms P-FV_{W}.

From the above experiments, we conclude that P-CV_{W} and P-FV_{W} are more reliable than Q-OPT, and P-CV_{W} tends to outperform P-FV_{W}.

5 Real-World Applications

Finally, we apply the proposed AL method to a wafer alignment problem in semiconductor exposure apparatus (see the left picture of Figure 3).

Recent semiconductors have the layered circuit structure, which are built by exposing circuit patterns multiple times. In this process, it is extremely important to align the wafer at the same position with very high accuracy. To this end, the location of markers are measured to adjust the shift and rotation of wafers. However, measuring the location of markers is time-consuming and therefore there is a strong need to reduce the number of markers to measure for speeding up the semiconductor production process.

The right picture of Figure 3 illustrates a wafer, where markers are printed uniformly over the wafer. Our goal here is to choose the most 'informative' markers to measure for better alignment of the wafer. A conventional choice is to measure markers far from the center in a symmetric way (see the right picture of Figure 3 again), which would provide robust estimation of the rotation angle. However, this naive approach is not necessarily the best since misalignment is

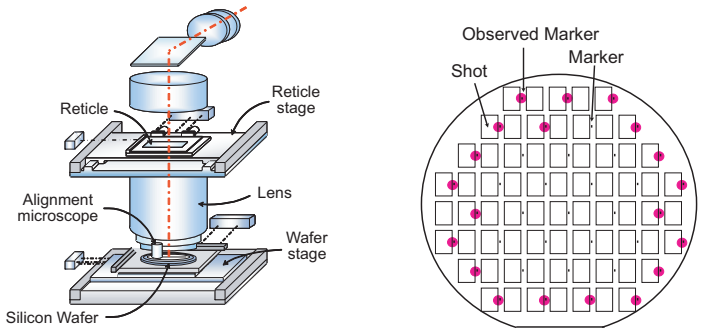


Fig. 3. Exposure apparatus (left) and a wafer (right).

not only caused by affine transformation, but also by several other non-linear factors such as a warp, a biased characteristic of measurement apparatus, and different temperature conditions. In practice, it is not easy to model such non-linear factors accurately, so the linear affine model or the second-order model is often used in wafer alignment. However, this causes model misspecification and therefore our proposed AL method would be useful in this application.

Let us consider the functions whose input $\boldsymbol{x} = (u, v)^\top$ is the location on the wafer and whose output is the horizontal discrepancy Δu or the vertical discrepancy Δv. We learn these functions by the following second-order model.

$$\Delta u \text{ or } \Delta v = \theta_0 + \theta_1 u + \theta_2 v + \theta_3 uv + \theta_4 u^2 + \theta_5 v^2.$$

We totally have 220 wafer samples and our experiment is carried out as follows. For each wafer, we choose $n_{\mathrm{tr}} = 20$ points from $n_{\mathrm{te}} = 38$ markers and observe the horizontal and the vertical discrepancies. Then the above model is trained and its prediction performance is tested using all 38 markers in the 220 wafers. This process is repeated for all 220 wafers. Since the choice of the sampling location by AL methods is stochastic, we repeat the above experiment for 100 times with different random seeds and take the mean value.

The mean and standard deviation of the squared test error over 220 wafers are summarized in Table 3. This shows that the proposed P-CV_{W} works significantly better than other sampling strategies and it provides about 10-percent reduction in the squared error from the conventional heuristic of choosing the outer markers. We also conducted similar experiments with the first-order or the third-order models and confirmed that P-CV_{W} still works the best. However, the errors were larger than the second-order model and therefore we omit the detail.

6 Conclusions

We extended a population-based AL method (FV_{W}) to a pool-based scenario (P-FV_{W}) and derived a closed-form 'optimal' resampling bias function. This closed-form solution is optimal within the full-expectation framework, but is not

Table 1. The mean squared test error for the toy dataset (means and standard deviations over 100 trials). For better comparison, we subtracted the model error δ^2 from the error and multiplied all values by 10^3. For each δ, the best method and comparable ones by the Wilcoxon signed-rank test at the significance level 5% are indicated with '$\circ$'.

	P-CV$_{\mathrm{W}}$	P-FV$_{\mathrm{W}}$	Q-OPT	Passive
$\delta = 0$	$^\circ$2.03±1.81	2.59±1.83	$^\circ$1.82±1.69	3.10±3.09
$\delta = 0.03$	$^\circ$2.17±2.04	2.81±2.01	2.62±2.05	3.40±3.55
$\delta = 0.06$	$^\circ$2.42±2.65	3.19±2.59	4.85±3.37	4.12±4.71
Average	$^\circ$2.21±2.19	2.86±2.18	3.10±2.78	3.54±3.85

Table 2. The mean squared test error for the DELVE datasets (means and standard deviations over 1000 trials). For better comparison, all the values are normalized by the mean error of the Passive method.

	P-CV$_{\mathrm{W}}$	P-FV$_{\mathrm{W}}$	Q-OPT	Passive
bank-8fm	$^\circ$0.89±0.14	0.95±0.16	0.91±0.14	1.00±0.19
bank-8fh	0.86±0.14	0.94±0.17	$^\circ$0.85±0.14	1.00±0.20
bank-8nm	$^\circ$0.89±0.16	0.95±0.20	0.91±0.18	1.00±0.21
bank-8nh	0.88±0.16	0.95±0.20	$^\circ$0.87±0.16	1.00±0.21
kin-8fm	$^\circ$0.78±0.22	0.87±0.24	0.87±0.22	1.00±0.25
kin-8fh	$^\circ$0.80±0.17	0.88±0.21	0.85±0.17	1.00±0.23
kin-8nm	$^\circ$0.91±0.14	0.97±0.16	0.92±0.14	1.00±0.17
kin-8nh	$^\circ$0.90±0.13	0.96±0.16	0.90±0.13	1.00±0.17
pumadyn-8fm	$^\circ$0.89±0.13	0.95±0.16	$^\circ$0.89±0.12	1.00±0.18
pumadyn-8fh	0.89±0.13	0.98±0.16	$^\circ$0.88±0.12	1.00±0.17
pumadyn-8nm	$^\circ$0.91±0.13	0.98±0.17	0.92±0.13	1.00±0.18
pumadyn-8nh	$^\circ$0.91±0.13	0.97±0.14	0.91±0.13	1.00±0.17
Average	$^\circ$0.87±0.16	0.95±0.18	0.89±0.15	1.00±0.20

	P-CV$_{\mathrm{W}}$	P-FV$_{\mathrm{W}}$	Q-OPT	Passive
bank-32fm	0.97±0.05	0.99±0.05	$^\circ$0.96±0.04	1.00±0.06
bank-32fh	0.98±0.05	0.99±0.05	$^\circ$0.96±0.04	1.00±0.05
bank-32nm	0.98±0.06	0.99±0.07	$^\circ$0.96±0.06	1.00±0.07
bank-32nh	0.97±0.05	0.99±0.06	$^\circ$0.96±0.05	1.00±0.06
kin-32fm	$^\circ$0.79±0.07	0.93±0.09	1.53±0.14	1.00±0.11
kin-32fh	$^\circ$0.79±0.07	0.92±0.08	1.40±0.12	1.00±0.10
kin-32nm	0.95±0.04	0.97±0.04	$^\circ$0.93±0.04	1.00±0.05
kin-32nh	0.95±0.04	0.97±0.04	$^\circ$0.92±0.03	1.00±0.05
pumadyn-32fm	$^\circ$0.98±0.12	0.99±0.13	1.15±0.15	1.00±0.13
pumadyn-32fh	0.96±0.04	0.98±0.05	$^\circ$0.95±0.04	1.00±0.05
pumadyn-32nm	0.96±0.04	0.98±0.04	$^\circ$0.93±0.03	1.00±0.05
pumadyn-32nh	0.96±0.03	0.98±0.04	$^\circ$0.92±0.03	1.00±0.04
Average	$^\circ$0.94±0.09	0.97±0.07	1.05±0.21	1.00±0.07

Table 3. The mean squared test error for the wafer alignment problem (means and standard deviations over 220 wafers). 'Conv.' indicates the conventional heuristic of choosing the outer markers.

P-CV$_{\mathrm{W}}$	P-FV$_{\mathrm{W}}$	Q-OPT	Passive	Conv.
$^\circ$1.93±0.89	2.09±0.98	1.96±0.91	2.32±1.15	2.13±1.08

necessarily optimal in the single-trial analysis. To further improve the performance, we extended another population-based method ($\mathrm{CV_W}$) to a pool-based scenario (P-$\mathrm{CV_W}$), which is input-dependent and therefore more accurate. However, P-$\mathrm{CV_W}$ does not allow us to obtain a closed-form solution. To cope with this problem, we proposed a practical procedure which efficiently searches for a better solution around the P-$\mathrm{FV_W}$ optimal solution. Simulations showed that the proposed method consistently outperforms the baseline passive learning scheme and compares favorably with other AL methods.

References

1. Fedorov, V.V.: Theory of Optimal Experiments. Academic Press, New York (1972)
2. Wiens, D.P.: Robust weights and designs for biased regression models: Least squares and generalized M-estimation. Journal of Statistical Planning and Inference **83**(2) (2000) 395–412
3. Kanamori, T., Shimodaira, H.: Active learning algorithm using the maximum weighted log-likelihood estimator. Journal of Statistical Planning and Inference **116**(1) (2003) 149–162
4. Sugiyama, M.: Active learning in approximately linear regression based on conditional expectation of generalization error. Journal of Machine Learning Research **7** (Jan. 2006) 141–166
5. McCallum, A., Nigam, K.: Employing EM in pool-based active learning for text classification. In: Proceedings of the 15th International Conference on Machine Learning. (1998)
6. Bach, F.: Active learning for misspecified generalized linear models. In Schölkopf, B., Platt, J., Hoffman, T., eds.: Advances in Neural Information Processing Systems 19. MIT Press, Cambridge, MA (2007)
7. Cohn, D.A., Ghahramani, Z., Jordan, M.I.: Active learning with statistical models. Journal of Artificial Intelligence Research **4** (1996) 129–145
8. Fukumizu, K.: Statistical active learning in multilayer perceptrons. IEEE Transactions on Neural Networks **11**(1) (2000) 17–26
9. Shimodaira, H.: Improving predictive inference under covariate shift by weighting the log-likelihood function. Journal of Statistical Planning and Inference **90**(2) (2000) 227–244
10. Fishman, G.S.: Monte Carlo: Concepts, Algorithms, and Applications. Springer-Verlag, Berlin (1996)
11. Huang, J., Smola, A., Gretton, A., Borgwardt, K.M., Schölkopf, B.: Correcting sample selection bias by unlabeled data. In Schölkopf, B., Platt, J., Hoffman, T., eds.: Advances in Neural Information Processing Systems 19. MIT Press, Cambridge, MA (2007) 601–608
12. Bickel, S., Brückner, M., Scheffer, T.: Discriminative learning for differing training and test distributions. In: Proceedings of the 24th International Conference on Machine Learning. (2007)
13. Sugiyama, M., Nakajima, S., Kashima, H., von Bünau, P., Kawanabe, M.: Direct importance estimation with model selection and its application to covariate shift adaptation. In: Advances in Neural Information Processing Systems 20, Cambridge, MA, MIT Press (2008)
14. Rasmussen, C.E., Neal, R.M., Hinton, G.E., van Camp, D., Revow, M., Ghahramani, Z., Kustra, R., Tibshirani, R.: The DELVE manual (1996)