Batch Mode Active Learning for Regression With Expected Model Change

Wenbin Cai, *Student Member, IEEE*, Muhan Zhang, and Ya Zhang, *Member, IEEE*

***Abstract*— While active learning (AL) has been widely studied for classification problems, limited efforts have been done on AL for regression. In this paper, we introduce a new AL framework for regression, expected model change maximization (EMCM), which aims at choosing the unlabeled data instances that result in the maximum change of the current model once labeled. The model change is quantified as the difference between the current model parameters and the updated parameters after the inclusion of the newly selected examples. In light of the stochastic gradient descent learning rule, we approximate the change as the gradient of the loss function with respect to each single candidate instance. Under the EMCM framework, we propose novel AL algorithms for the linear and nonlinear regression models. In addition, by simulating the behavior of the sequential AL policy when applied for *k* iterations, we further extend the algorithms to batch mode AL to simultaneously choose a set of *k* most informative instances at each query time. Extensive experimental results on both UCI and StatLib benchmark data sets have demonstrated that the proposed algorithms are highly effective and efficient.**

***Index Terms*— Active learning (AL), batch mode, expected model change, linear regression, nonlinear regression.**

I. Introduction

DATA collection, feature engineering, and algorithm design are the three essential components in building any machine learned model. However, in many learning tasks, the importance of data collection is often under weighted. Passive learning, which randomly selects training examples according to a certain underlying distribution and sends them to editors for manual annotation, is the most widely used approach for data collection. The manual annotation process could be both time consuming and labor consuming. Very often, a learning task only has a limited budget, so that one has to make a tradeoff between data quality and model performance. On the other hand, not all examples selected by passive learning positively contribute to the model training. Given a fixed budget, active learning (AL) selectively chooses the most helpful instances, maximizing the performance of the model learned. A typical AL process iterates through the following three steps: 1) build a base model from a given small initial training seed; 2) choose the informative examples with a predefined sampling function and query their labels; and 3) add those newly labeled examples to the training set and update the model. This active sampling process is usually iterated until a given performance threshold is reached. In recent years, AL has been applied in many machine learning applications [6], [21], [30].

So far, AL has been widely studied for classification. Uncertainty sampling, a widely adopted AL strategy, chooses the example whose label the current classifier is most uncertain about [11], [20], [25]. Another classical AL framework is query-by-committee (QBC), which constructs a committee of member models and queries the unlabeled example having the largest disagreement among the members [23]. Recently, the capability of examples to change the model has been investigated for AL. In the literature, limited efforts have been done on AL for regression. Consequently, a general AL framework concentrating on regression is of great need. In this paper, we attempt to employ the model change-based strategy for regression. The major challenge lies in the measurement of model change, especially for nonparametric models, such as tree-based models.

Most existing AL methods have focused on selecting a single most informative example in each data sampling iteration. This AL framework is usually regarded as sequential AL (SAL). But, this SAL scheme is not applicable in real applications due to the following two reasons: 1) the model has to retrain after each new example is queried, resulting in an extremely high time cost and 2) SAL could make wasteful use of the practical resource if a parallel labeling system is available (we often have multiple editors in practice).

To address those limitations, batch mode AL (BMAL), which selects a batch of examples in each iteration, has recently drawn a great deal of attention among machine learning researchers. One naive approach toward BMAL is to simply apply an existing SAL algorithm *k* times to generate a batch, e.g., selecting the *k* minimum margin examples when employing the margin-based strategy. The main problem with such naive BMAL approach is that the similarity or correlation among selected examples is totally ignored. To reduce the data redundancy among a selected batch, several BMAL algorithms are recently proposed for classification, mainly based

Manuscript received May 25, 2014; revised January 23, 2016 and February 26, 2016; accepted March 10, 2016. Date of publication April 20, 2016; date of current version June 15, 2017. This work was supported in part by the High Technology Research and Development Program of China under Grant 2015AA015801, in part by the National Natural Science Foundation of China under Grant 61003107 and Grant 61521062, and in part by the Science and Technology Commission of Shanghai Municipality under Grant 14511107500 and Grant 12DZ2272600. *(Corresponding author: Ya Zhang.)*

W. Cai and Y. Zhang are with the Cooperative Medianet Innovation Center, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: cai-wenbin@sjtu.edu.cn; ya_zhang@sjtu.edu.cn).

M. Zhang is with Washington University in St. Louis, St. Louis, MO 63130 USA (e-mail: muhan@wustl.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2016.2542184

on optimizing certain information measures [2]–[5], [7]–[9]. However, the information measures adopted (e.g., fisher information matrix [2]–[4], expected log likelihood [5], and mutual information [7]) are derived for classification models. As a result, the existing BMAL methods are not directly applicable to regression.

In this paper, we consider the ability of the data instances to change the current model, and accordingly propose a novel AL framework, named expected model change maximization (EMCM), in the context of regression. EMCM quantifies the change as the difference between the current model parameters and the new model parameters learned from enlarged training data, and chooses the data examples that result in the greatest change. In light of the stochastic gradient descent (SGD) rule, which repeatedly updates the model parameters according to the negative gradient of the loss at each training example, we use the gradient of the loss at a candidate example to estimate its ability to change the model.

Under the EMCM framework, we first develop a novel AL algorithm for linear regression, where the change is calculated as the norm of gradient at a single candidate example. For nonlinear regression, gradient boosting decision tree (GBDT) is used as the base learner in this paper. In comparison to linear regression, GBDT has a discrete model form, which makes it infeasible to directly measure the model change as gradient. To address this issue, we generate super features from trees and approximate GBDT as a linear regression model with feature mapping. We then propose an AL algorithm for GBDT-based nonlinear regression.

Besides, motivated by recent work of matching the behavior of an SAL method [9], we further extend the proposed EMCM algorithms to BMAL, and the key idea is that BMAL could capture the per-example accuracy improvement of SAL with fewer iterations via batch selection. Hence, we extend sequential EMCM algorithms to batch mode EMCM (B-EMCM) algorithms by approximating the SAL behavior without model retraining to simultaneously choose a batch of examples, i.e., selecting a batch of k examples that best matches the outputs of the sequential EMCM counterparts when performed for k iterations.

We validate our AL algorithms on various benchmark data sets from UCI and StatLib. Substantial experimental results have demonstrated that the proposed AL algorithms are highly effective and efficient.

The proposed EMCM is generic, and is naturally suitable to the SGD-based models. We also discuss its potential usability for non-SGD learning models [we extend EMCM to Gaussian process (GP) regression in this paper] to show its applicability in a wide spectrum of applications.

The main contributions of this paper can be summarized as follows.

1) We introduce a novel AL framework for regression, called EMCM, which queries the examples, maximizing the model change once added to the training data.
2) Under this framework, we design new AL algorithms for both linear and nonlinear regression models, which are effective in selecting useful instances for model training.
3) By imitating the behavior of the sequential counterparts, we further extend sequential EMCM algorithms to B-EMCM algorithms.

The rest of this paper is structured as follows. Section II summarizes the existing AL approaches. Section III presents the general framework of EMCM. Our proposed AL algorithms are presented in Section IV. Section V details our B-EMCM algorithms. Section VI presents the experiments and interprets the results. Possible extensions are discussed in Section VII. Finally, we conclude this paper in Section VIII.

II. Related Work

AL has gained a great deal of attentions in the machine learning community in recent years. In this section, we summarize several classic AL strategies. A comprehensive AL survey can be found in [32].

A. Active Learning for Classification

So far, various types of AL strategies have been investigated for classification tasks [16], [24].

The extensively used strategy is uncertainty sampling [11], [20], [25], which queries the instance whose label the current classifier is most uncertain about. This data sampling approach is straightforward for probabilistic models using entropy-based measurement [20]. This strategy could also be applied to non-probabilistic models, such as support vector machines (SVMs), and aims to choose the instance located in the margin [25].

QBC [23] is another typical AL framework, which constructs a committee of model members and queries the instance the members disagree the most. The widely used function for disagreement measure is called vote entropy [27]. To effectively construct the committee, various types of ensemble learning techniques have been employed, such as boosting and bagging [28], [29], [31].

Another decision-theoretic AL strategy aims at minimizing the generalization error of the model. Roy and McCallum [33] proposed to choose the instance that results in the lowest generalization error on the future unseen data after labeled and incorporated into the training set. The major drawback of this AL approach lies in its high computational complexity. Instead of choosing the instance yielding the smallest generalization error, Nguyen and Smeulders [36] suggested to select the data instance that contributes the most to the current generalization error.

Model change-based idea has been studied in classification. Settles *et al.* [19] introduced a data selection algorithm, which chooses the instance that would greatly change to the classifier, and the change was quantified as the length of the new gradient of the loss function with respect to model parameters.

B. Active Learning for Regression

In the literature, AL targeting on regression is still less well-researched. We summarize the existing methods as follows.

Castro *et al.* [10] theoretically analyzed AL in the context of regression with a certain noise ratio. Sugiyama [12] proposed a population-based AL method, where the input data examples can be arbitrarily generated in the space.

Sugiyama and Nakajima [13] introduced a theoretically optimal AL algorithm that attempts to directly minimize the generalization error when employing an additive regression model. Cohn *et al.* [14] presented an AL strategy with a similar motivation, which chooses the example, minimizing the output variance to reduce generalization error. Freund *et al.* [23] suggested that the QBC framework could be applied to regression cases where the outputs are continuous, which is related to the variance-based QBC [26]. Yu and Kim [17] provided passive sampling heuristics based on the geometric characteristics of data. Cai *et al.* [18] presented a novel data sampling solution in the context of regression, which queries the example leading to the largest model change.

As listed before, all the regression-oriented AL techniques are designed under the sequential mode. Due to the important role of BMAL in practical cases, it is highly desirable to derive BMAL methods in the context of regression.

C. Batch Mode Active Learning

In the past, there is a limited amount of work on BMAL for classification. In general, the key idea of BMAL is to reduce the redundancy among the selected examples in the batch, and thus each example could provide unique information for model updating.

Brinker [1] introduced an SVM-based batch selection method by incorporating the diversity of data examples. Hoi *et al.* [2]–[4] presented a BMAL framework, employing the fisher information matrix to measure the overall information for a set of examples. Guo and Schuurmans [5] proposed a discriminative BMAL approach that formulates the batch selection as a continuous optimization problem. In recent years, Guo [7] introduced a novel BMAL method that selects a set of examples, maximizing the mutual information between labeled and unlabeled instances. Chattopadhyay *et al.* [8] selected a batch of examples, which aims to minimize the difference in data distribution between the expanded training data and the large unlabeled data after annotation. Azimi *et al.* [9] proposed a novel BMAL approach with the idea of approximating the behavior of SAL methods when applied for k iterations. Unlike previous BMAL strategies that require the batch size as an input, Chakraborty *et al.* [38] recently proposed new dynamic BMAL frameworks, which can adaptively determine the batch size by integrating the batch size and selection criteria into a single optimization formulation.

As summarized in the above, the existing BMAL algorithms are mainly derived with classification models, e.g., SVM-based BMAL [1], which cannot be directly generalized to regression. In this paper, we extend our previous work [18] to BMAL by simulating the sequential mode AL behavior to simultaneously choose a set of examples without retraining, which is the new contributions of this paper. Now, we present the formulations of our AL methods targeting on regression, and then extend them to BMAL algorithms.

III. Expected Model Change Maximization

In this section, we present the general framework of EMCM. After that, we give an empirical interpretation of EMCM to motivate this idea.

A. Framework of EMCM

The objective of supervised learning can be represented as learning a function $f(.)$ that minimizes the generalization error on the future test data

$$\epsilon = \int_{\mathcal{X}\times\mathcal{Y}} \mathcal{L}[f(x), y(x)]dP(x, y) \tag{1}$$

where $y(x)$ and $f(x)$ stand for the true label and the predicted label of the example x, respectively. $\mathcal{L}[f(x), y(x)]$ is a given loss function. Since the joint distribution $P(x, y)$ is unknown, we cannot directly solve formula (1). In practice, we are given a training set $\mathcal{D} = \{(x_i, y_i), x_i \in \mathcal{X}, y_i \in \mathcal{Y}\}_{i=1}^n$ drawn independent identically distributed from $P(x, y)$, i.e., $\mathcal{D} \sim P(x, y)$, and then build the model by minimizing the empirical error on $\mathcal{D}$

$$\hat{\epsilon}_{\mathcal{D}} = \sum_{i=1}^{n} \mathcal{L}[f(x_i), y_i]. \tag{2}$$

This model fitting process is well known as the empirical risk minimization principle.

Suppose that the model is parameterized by Θ. To search Θ minimizing the empirical error, a widely used search approach is SGD rule, which updates Θ iteratively according to the negative gradient of the loss $\mathcal{L}(\Theta)$ with respect to each training example (x_i, y_i)

$$\Theta_{\text{new}} \leftarrow \Theta - \alpha\frac{\partial \mathcal{L}_{x_i}(\Theta)}{\partial\Theta}, \quad i = 1, 2, \ldots, n \tag{3}$$

where α represents the learning rate.

Now, we investigate the SGD rule in AL problems. Suppose a candidate instance (x^+, y^+) is incorporated into the training set. The empirical error on the accumulated training set $\mathcal{D}^+ = \mathcal{D} \cup (x^+, y^+)$ becomes

$$\hat{\epsilon}_{\mathcal{D}^+} = \sum_{i=1}^{n} \mathcal{L}[f(x_i), y_i] + \underbrace{\mathcal{L}[f(x^+), y^+]}_{:=\mathcal{L}_{x^+}(\Theta)}. \tag{4}$$

Consequently, the parameter Θ is changed due to training set is changed, i.e., the instance (x^+, y^+) is added to the training set. According to the SGD update rule, the model change $\mathbf{C}_\Theta(x^+)$, i.e., the parameter change, can be approximated as the gradient of the loss function at the candidate instance. It can be written as

$$\mathbf{C}_\Theta(x^+) = \triangle\Theta \approx \alpha\frac{\partial \mathcal{L}_{x^+}(\Theta)}{\partial\Theta}. \tag{5}$$

Since the objective of our AL strategy is to choose the sample x^* that leads to the maximum model change, we can formulate the selection function as follows:

$$x^* = \underset{x\in\mathcal{U}}{\arg\max}\, ||\mathbf{C}_\Theta(x)|| \tag{6}$$

where $\mathcal{U}$ denotes the unlabeled pool set and x^* is the selected example.

In practical settings, we cannot directly calculate the model change in (6), since the true label y^+ of the candidate example x^+ is unknown before querying. Instead, for regression tasks, we calculate expected model change over the prediction distribution $y^+ \in Y$, estimated by the current regression model to approximate the true change. We assume that the learning rate α is identical for each candidate data sample, and our EMCM criteria for AL are formulated as

$$x^* = \arg\max_{x \in \mathcal{U}} \int_Y \left\| \frac{\partial \mathcal{L}_x(\Theta)}{\partial \Theta} \right\| P(y|x) dy \tag{7}$$

where $P(y|x)$ is the conditional probability of the label y given the instance x estimated by the current model.

B. Empirical Interpretation

In this section, we introduce an empirical interpretation for our EMCM framework by linking the model change to the model's generalization performance. We believe that the model change is a reasonable indicator for reducing the generalization error with the following two major reasons.

1) The generalization capability can be changed if and only if the current model is changed, As a result, it is useless to query the instance that cannot update the current model in AL.
2) The data points significantly changing the current model are expected to produce a faster convergence speed to the true model, and this is the underlying motivation behind our EMCM framework.

We here note that a big change in the current model not always leads to better generalization performance, since an outlier also results in a big model change. However, in AL tasks, unlabeled examples are repeatedly selected from a given pool set. Once the model has been changed by an outlier, the EMCM strategy will certainly query a good example in the next data selection iteration that maximizes the change again, which immediately relieves the negative effect of the outlier. In practice, because the amount of outliers is usually very restricted in the data, it is reasonable to believe that the proposed EMCM framework will result in very good generalization performance with more data instances queried.

IV. EMCM FOR REGRESSION

In this section, we first apply the proposed EMCM framework to linear regression and accordingly derive the algorithm. Then, the EMCM algorithm for GBDT-based nonlinear regression is presented, i.e., GBDT is used as the base learner.

A. EMCM for Linear Regression

Linear regression consists of finding the best-fitting straight line through the training data, which can be formulated as

$$f(x;\theta) = \sum_{i=0}^{p} \theta_i x_{(i)} = \theta^T x \tag{8}$$

where $x_{(0)} = 1$ is the intercept term and the values of $x_{(i)}$ are the features of the example x. The linear regression model is parameterized by the weight vector θ. Below, focusing on the change in the parameter θ, we apply the proposed EMCM framework to the linear regression model to derive our AL algorithm.

Algorithm 1 EMCM for Linear Regression

```
Input: the small initial labeled data set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$, the unlabeled pool set $\mathcal{U}$, the size of the ensemble $Z$, the linear regression model $f(x;\theta)$ trained on $\mathcal{D}$.
1: $\mathcal{B}(Z) = \{f_1, f_2, ..., f_Z\}$
2: for each $x$ in $\mathcal{U}$ do
3:     $\{y_1, y_2, ..., y_Z\} \leftarrow \mathcal{B}(Z)$
4:     Calculate the derivative via Eq. (11).
5:     Estimate the true model change via Eq. (12).
6: end for
Output: the example $x^*$ having the greatest model change.
```

Training linear regression is to minimize the squared-error loss on the current training set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$

$$\hat{\epsilon}_{\mathcal{D}} = \frac{1}{2} \sum_{i=1}^{n} (f(x_i) - y_i)^2. \tag{9}$$

Suppose a candidate point x^+ with a given label y^+ is added to the training set $\mathcal{D}$. The empirical error on the accumulated training set $\mathcal{D}^+ = \mathcal{D} \cup (x^+, y^+)$, then becomes

$$\hat{\epsilon}_{\mathcal{D}^+} = \frac{1}{2} \sum_{i=1}^{n} (f(x_i) - y_i)^2 + \underbrace{\frac{1}{2}(f(x^+) - y^+)^2}_{:=\mathcal{L}_{x^+}(\theta)}. \tag{10}$$

Therefore, for linear regression, the derivative of the squared-error loss $\mathcal{L}_{x^+}(\theta)$ with respect to the parameters θ at x^+ is formulated as

$$\begin{aligned} \frac{\partial \mathcal{L}_{x^+}(\theta)}{\partial \theta} &= (f(x^+) - y^+) \frac{\partial f(x^+)}{\partial \theta} \\ &= (f(x^+) - y^+) \frac{\partial \theta^T x^+}{\partial \theta} \\ &= (f(x^+) - y^+) x^+. \end{aligned} \tag{11}$$

As discussed previously, because the true label y^+ is unknown in advance, we here apply bootstrap to generate an ensemble $\mathcal{B}(Z) = \{f_1, f_2, \ldots, f_Z\}$ to estimate the prediction distribution $y^+ \in \{y_1, y_2, \ldots, y_Z\}$, and use the expected model change to approximate the true model change. The use of bootstrap to estimate prediction distribution has been well investigated [35]. Thus, the final EMCM sampling function for linear regression is expressed as

$$x^* = \arg\max_{x \in \mathcal{U}} \frac{1}{Z} \sum_{z=1}^{Z} ||(f(x) - y_z(x))x|| \tag{12}$$

where $f(x)$ is the model's prediction, and $y_z(x)$, $z = 1, \ldots, Z$ are calculated from the Z bootstrap models, which are learned with the bootstrap examples. The pseudocode for EMCM for linear regression is summarized in Algorithm 1.

B. EMCM for Nonlinear Regression

We here employ GBDT as the nonlinear regression base model, which can be formulated as an additive model

$$f\left(x; \{\lambda, \Theta\}_1^M\right) = \sum_{m=1}^{M} \lambda_m h_m(x; \Theta_m) \tag{13}$$

where M denotes the number of individual trees in GBDT and the values of $\{\lambda, \Theta\}_1^M$ parameterize the model. Each single tree $h_m(x; \Theta_m)$ is a J-terminal node regression tree

$$h\big(x; \{\gamma, R\}_1^J\big) = \sum_{j=1}^{J} \gamma_j \mathbf{1}(x \in R_j) \tag{14}$$

where the values of $\{\gamma\}_1^J$ are coefficients, the values of $\{R\}_1^J$ are the regions partitioned by the individual tree, and $\mathbf{1}(\cdot)$ denotes the indicator function. In recent research, GBDT has been extensively employed as the state-of-art learning algorithm in many machine learning problems [15]. More details about GBDT can be found in [34] and [37].

Because the parameters in decision tree-based models are not differentiable, we are not able to directly measure the model change using the derivative. To address this issue, we assume that adding a single example to training data does not influence the tree structure. The prediction rule of a single decision tree is: $x \in R_j \Rightarrow h(x) = \gamma_j$. Because the number of examples selected at each query time is quite small in comparison of the size of existing training data, in most cases, the region where x is located remains unchanged. Therefore, this assumption is reasonable, especially for the SAL cases.

Under this assumption, by employing the concept of super features, we map each unlabeled instance to the super features based on each individual tree

$$\phi(x) = [h_1(x), h_2(x), \ldots, h_M(x)]^T \tag{15}$$

As a result, the GBDT model can be approximated as a linear regression model

$$f\big(x; \{\lambda\}_1^M\big) = \sum_{m=1}^{M} \lambda_m h_m(x) = \lambda^T \phi(x). \tag{16}$$

Hence, we concentrate on the change in parameters $\{\lambda\}_1^M$ for AL. The derivative of the squared-error loss $\mathcal{L}_{x^+}(\lambda)$ with respect to the parameters $\lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_M\}$ at the candidate example x^+ is

$$\begin{aligned}\frac{\partial \mathcal{L}_{x^+}(\lambda)}{\partial \lambda} &= (f(x^+) - y^+)\frac{\partial f(x^+)}{\partial \lambda}\\ &= (f(x^+) - y^+)\frac{\partial \lambda^T \phi(x^+)}{\partial \lambda}\\ &= (f(x^+) - y^+)\phi(x^+). \end{aligned}\tag{17}$$

Identically, we create an ensemble with the bootstrap examples $\mathcal{B}(Z) = \{f_1, f_2, \ldots, f_Z\}$ to calculate the prediction distribution $y^+ \in \{y_1, y_2, \ldots, y_Z\}$, and estimate the true model change using expectation calculation. Our final EMCM sample selection criteria for GBDT can be formulated as

$$x^* = \arg\max_{x \in \mathcal{U}} \frac{1}{Z} \sum_{z=1}^{Z} ||(f(x) - y_z(x))\phi(x)||. \tag{18}$$

The corresponding pseudocode for EMCM for GBDT regression is presented in Algorithm 2.

Algorithm 2 EMCM for Nonlinear Regression

```
Input: the small initial labeled data set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$, the unlabeled pool set $\mathcal{U}$, the size of the ensemble $Z$, the GBDT model $f(x;\lambda)$ trained on $\mathcal{D}$.
1: $\mathcal{B}(Z) = \{f_1, f_2, ..., f_Z\}$
2: for each $x$ in $\mathcal{U}$ do
3:     Generate super features via Eq. (15).
4:     $\{y_1, y_2, ..., y_Z\} \leftarrow \mathcal{B}(Z)$
5:     Calculate the derivative via Eq. (17).
6:     Estimate the true model change via Eq. (18).
7: end for
Output: the example $x^*$ having the greatest model change.
```

V. Batch Mode EMCM for Regression

To select a total of k data points, the SAL method chooses a single example in each sampling iteration and has to be retrained every time the new example is labeled and added to the training set. The main drawback of SAL lies in its high computational complexity, making it unacceptable in practical applications.

Motivated by recent research on BMAL [9], our goal for BMAL is to select a batch of k examples $b = \{x_1^*, \ldots, x_j^*, \ldots, x_k^*\}$ that closely matches the outputs of the sequential counterparts without retraining. Clearly, the key challenge is how to choose the jth example x_j^* $(2 \le j \le k)$ after the first $j-1$ ones are selected without any retraining procedure.

In this section, we extend the sequential EMCM algorithms to the B-EMCM algorithms by simulating the behavior of the SAL counterpart to select a batch of k examples. As discussed previously, the objective is to catch the per-example performance improvement of sequential methods with fewer sampling iterations via batch selection. The details of our B-EMCM algorithms are provided in the following.

A. B-EMCM for Linear Regression

Under our EMCM framework, we here consider the model change, i.e., the gradient of the error, with respect to the jth candidate example $(2 \le j \le k)$ after selecting the first $j-1$ ones without retraining the model.

For linear regression, if the first selected data point (x_1^*, y_1^*) were incorporated into the training set $\mathcal{D}$, the parameter θ of the regression model is adjusted according to the gradient of the loss. Given the derivative at x_1^* calculated with (11), the new parameter θ_* can be approximated as follows:

$$\theta_* \approx \theta - \alpha \frac{\partial \mathcal{L}_{x_1^*}(\theta)}{\partial \theta}. \tag{19}$$

Consequently, the new linear regression model can be written as follows:

$$f_*(x) = \theta_*^T x \approx \left(\theta - \alpha \frac{\partial \mathcal{L}_{x_1^*}(\theta)}{\partial \theta}\right)^T x \tag{20}$$

and the derivative of the loss at the second candidate instance x_2^+ is calculated as

$$\begin{aligned}\frac{\partial \mathcal{L}_{x_2^+}\left(\theta_* | \left(x_1^*, y_1^*\right)\right)}{\partial \theta_*} &= \left(f_*\left(x_2^+\right) - y_2^+\right) x_2^+ \\ &\approx \left(\left(\theta - \alpha \frac{\partial \mathcal{L}_{x_1^*}(\theta)}{\partial \theta}\right)^T x_2^+ - y_2^+\right) x_2^+ \\ &= \left(\theta^T x_2^+ - y_2^+ - \alpha \left(\frac{\partial \mathcal{L}_{x_1^*}(\theta)}{\partial \theta}\right)^T x_2^+\right) x_2^+ \\ &= \left(f\left(x_2^+\right) - y_2^+\right) x_2^+ - \alpha \left(\frac{\partial \mathcal{L}_{x_1^*}(\theta)}{\partial \theta}\right)^T x_2^+ x_2^+ \\ &= \frac{\partial \mathcal{L}_{x_2^+}(\theta)}{\partial \theta} - \alpha \left(\frac{\partial \mathcal{L}_{x_1^*}(\theta)}{\partial \theta}\right)^T x_2^+ x_2^+. \end{aligned} \quad (21)$$

It formally shows that the derivative of the loss at the second candidate example x_2^+ can be directly obtained with the current model $f(.)$ without retraining.

In general, given the derivative at previous selected examples x_i^* ($i = 1, 2, \ldots, j-1$), the derivative with respect to the jth candidate example x_j^+ ($2 \le j \le k$) after selecting first $j-1$ ones can be approximated as

$$\frac{\partial \mathcal{L}_{x_j^+}\left(\theta_* | b_{j-1}^*\right)}{\partial \theta_*} \approx \frac{\partial \mathcal{L}_{x_j^+}(\theta)}{\partial \theta} - \alpha \sum_{i=1}^{j-1} \left(\frac{\partial \mathcal{L}_{x_i^*}\left(\theta_* | b_{i-1}^*\right)}{\partial \theta_*}\right)^T x_j^+ x_j^+ \quad (22)$$

where

$$\begin{aligned} b_{j-1}^* &= \left\{\left(x_1^*, y_1^*\right), \ldots, \left(x_{j-1}^*, y_{j-1}^*\right)\right\}, \quad j = 2, 3, \ldots, k \\ b_{i-1}^* &= \left\{\left(x_1^*, y_1^*\right), \ldots, \left(x_{i-1}^*, y_{i-1}^*\right)\right\}, \quad 1 \le i \le j-1. \end{aligned}$$

Note that $b_{i-1}^* = \emptyset$ if $i = 1$. For convenient use of the notation in the following, we define:

$$C\left(x_j^+, b_{j-1}^*\right) = \alpha \sum_{i=1}^{j-1} \left(\frac{\partial \mathcal{L}_{x_i^*}\left(\theta_* | b_{i-1}^*\right)}{\partial \theta_*}\right)^T x_j^+ x_j^+. \quad (23)$$

Furthermore, combined with (11), it is easy to see that the derivative at each candidate data point x_j^+ (including the first one) can be calculated in a unified manner. Thus, we have

$$\frac{\partial \mathcal{L}_{x_j^+}\left(\theta_* | b_{j-1}^*\right)}{\partial \theta_*} \approx \left(f\left(x_j^+\right) - y_j^+\right) x_j^+ - C\left(x_j^+, b_{j-1}^*\right)$$
$$j = 1, 2, \ldots, k, \quad 1 \le i \le j-1. \quad (24)$$

The added term compared with the EMCM, i.e., the second term of the above, $C(x_j^+, b_{j-1}^*)$, measures the relationship between the candidate example x_j^+ and the previous selected examples b_{j-1}^*, and it equals zero when $j = 1$.

Similarly, by taking advantage of the prediction distribution $y^+ \in \{y_1, y_2, \ldots, y_Z\}$ estimated using the ensemble $\mathcal{B}(Z)$, we approximate the true model change by averaging the estimated model change over Z possible labels. Our B-EMCM sampling function for linear regression can be formulated as

$$x_j^* = \underset{x \in \mathcal{U} \setminus b_{j-1}^*}{\arg\max} \frac{1}{Z} \sum_{z=1}^{Z} \left|\left|(f(x) - y_z(x))x - C\left(x, b_{j-1}^*\right)\right|\right|$$
$$j = 1, 2, \ldots, k, \quad 1 \le i \le j-1. \quad (25)$$

Note that if the learning rate α is set to be zero, i.e., the model is not updated during AL iterations, the B-EMCM algorithm is equivalent to the naive batch EMCM method, i.e., selecting top-k maximum model change examples without considering the data correlation. The pseudocode of B-EMCM for linear regression is presented in Algorithm 3.

Algorithm 3 B-EMCM for Linear Regression

```
Input: the small initial labeled data set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$, the unlabeled pool set $\mathcal{U}$, the size of the ensemble $Z$, the size of the batch $k$, the linear regression model $f(x;\theta)$ trained on $\mathcal{D}$.
1: initialize: $b = \emptyset$
2: $\mathcal{B}(Z) = \{f_1, f_2, ..., f_Z\}$
3: while $| b | < k$ do
4:   for each $x$ in $\mathcal{U}$ do
5:     $\{y_1, y_2, ..., y_Z\} \leftarrow \mathcal{B}(Z)$
6:     Calculate the derivative via Eq. (24).
7:     Estimate the true model change via Eq. (25).
8:   end for
9:   Select $x^*$ having the greatest model change.
10:  $\mathcal{U} \leftarrow \mathcal{U} \setminus x^*$, $b \leftarrow b \cup x^*$
11: end while
Output: the batch $b = \{x_1^*, x_2^*, ..., x_k^*\}$.
```

B. B-EMCM for Nonlinear Regression

For BMAL with the GBDT model, since we simultaneously choose a batch of data examples in each query time, the above assumption is that the structure of tree remains unchanged during one single AL iteration might not be valid if the size of batch is relatively large. One possible solution is to incorporate the change of the tree structure, which could be measured by the change in decision regions, into our EMCM framework. We leave this as our future work. In this BMAL study, we still use the change in the weights $\lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_M\}$ to approximate the change in GBDT and ignore the tree structure change.

Similar to the above, we first approximate GBDT as a linear regression model through feature mapping $\phi(x)$. Then, the B-EMCM algorithm for GBDT is proceeded in a similar fashion as above. The derivative of the error with respect to the jth candidate example x_j^+ ($1 \le j \le k$) after choosing the first $j-1$ ones is calculated as

$$\frac{\partial \mathcal{L}_{x_j^+}\left(\lambda_* | b_{j-1}^*\right)}{\partial \lambda_*} \approx \left(f\left(x_j^+\right) - y_j^+\right) \phi\left(x_j^+\right) - C\left(\phi\left(x_j^+\right), b_{j-1}^*\right)$$
$$j = 1, 2, \ldots, k, \quad 1 \le i \le j-1. \quad (26)$$

Similarly, the true model change is approximated with the expectation calculation over Z possible labels that are estimated through the ensemble $\mathcal{B}(Z)$, and the final B-EMCM

```
Algorithm 4 B-EMCM for Nonlinear GBDT Regression
Input: the small initial labeled data set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$, the unlabeled pool set $\mathcal{U}$, the size of the ensemble $Z$, the size of the batch $k$, the GBDT model $f(x;\lambda)$ trained on $\mathcal{D}$.
1: initialize: $b = \emptyset$
2: $\mathcal{B}(Z) = \{f_1, f_2, ..., f_Z\}$
3: while $|\, b \,| < k$ do
4:   for each $x$ in $\mathcal{U}$ do
5:     Generate super features via Eq. (15).
6:     $\{y_1, y_2, ..., y_Z\} \leftarrow \mathcal{B}(Z)$
7:     Calculate the derivative via Eq. (26).
8:     Estimate the true model change via Eq. (27).
9:   end for
10:  Select $x^*$ having the greatest model change.
11:  $\mathcal{U} \leftarrow \mathcal{U} \setminus x^*$, $b \leftarrow b \cup x^*$
12: end while
Output: the batch $b = \{x_1^*, x_2^*, ..., x_k^*\}$.
```

selection rule can be expressed as

$$x_j^* = \underset{x \in \mathcal{U} \setminus b_{j-1}^*}{\arg\max} \frac{1}{Z} \sum_{z=1}^{Z} \left|\left|(f(x) - y_z(x))\phi(x) - C\big(\phi(x), b_{j-1}^*\big)\right|\right|$$
$$j = 1, 2, \ldots, k, \quad 1 \le i \le j-1. \quad (27)$$

Algorithm 4 gives the pseudocode of B-EMCM for nonlinear GBDT regression.

Remark: Based on the above derivation, we can get a uniform view to our algorithms, i.e., EMCM and B-EMCM are related to each other by

$$\text{B-EMCM} = \text{EMCM} - \text{correlation}\big(x_j^+, b_{j-1}^*\big) \quad (28)$$

implying that B-EMCM is the extension of EMCM by taking particular consideration of the correlation between the current candidate example x_j^+ and the previous selected examples b_{j-1}^*.

VI. Experiments

In this section, we present extensive experimental studies to demonstrate the effectiveness and efficiency of our algorithms.

A. Data Sets and Experimental Setups

We use eight benchmark data sets from the UCI machine learning repository[1] and the StatLib from CMU.[2] These data sets are collected from various domains and have been extensively used for testing regression models [17], [41]. Their statistics are presented in Table I. We randomly split each data set into three disjoint parts to construct the initial training seed $\mathcal{D}$, the unlabeled set $\mathcal{U}$, and the test set $\mathcal{T}$. In this paper, we consider a hard AL scenario, where the size of initial training set is small: $\mathcal{D}(10\%) + \mathcal{U}(70\%) + \mathcal{T}(20\%)$. For the features, we normalize the features as [18].

Two regression models are adopted as the basis learners: the linear regression model and the GBDT model. The learning rate α is empirically set to be 0.05, which is commonly used in SGD. We use the standard bootstrap method [37] to estimate the predictive distribution. That is, we randomly draw data sets with replacement from the training set $\mathcal{D}$, and each sample the same size as the training set. We then refit Z bootstrap models on these bootstrap data sets. To decide the optimal parameter, i.e., the size of the ensemble Z, in bootstrap, we experiment with six values of Z : $Z = \{3, 4, 5, 6, 7, 8\}$, and choose the one having the best averaged performance. Here, we set $Z = 4$ in our experiments based on our empirical studies.

In our implementation, the AL process repeats ten iterations. In each round of data sampling, 3% of the entire instances are selected from $\mathcal{U}$ and labeled (i.e., $k = 3\%$). In practice, the choice of k highly relies on the budget available.

[1]http://archive.ics.uci.edu/ml/
[2]http://lib.stat.cmu.edu/

TABLE I
Statistics of the Eight Regression Data Sets

Data sets		# Examples	# Features	Source
PM10		500	7	StatLib
Housing		506	13	UCI
Forest		517	10	UCI
CPS		537	8	StatLib
Concrete		1,030	8	UCI
Wine	**Redwine**	1,599	11	UCI
	Whitewine	4,898	11	UCI
Bike		17,379	14	UCI

B. Comparison Methods and Evaluation Metrics

To test the effectiveness of our proposed methods, we compare our algorithms with several state-of-the-art AL competitors. Five baselines are included in our experiments. Note that all the competitors are performed in the naive batch mode, i.e., choosing the top k examples in each selection round based on their usefulness measure, as the existing BMAL methods are not readily applicable to regression.

1) *P-ALICE:* P-ALICE is a statistically optimal AL strategy using the bias-variance decomposition, which aims to minimize the conditional variance expected over training output noise given training input points [13].
2) *P-FV$_W$:* P-FV$_W$ is another variance reduction technique, which is to minimize the full variance expected over both training output noise and training examples [13].
3) *QBC:* The QBC for regression approach chooses the data point, which has the largest variance among the members' prediction [23], [26].
4) *Greedy:* The greedy approach aims to select the new example having the largest minimum distance from labeled data [17].
5) *RAND:* The random data selection, which is widely used in practical applications, represents a baseline.

Furthermore, our model change-based AL algorithms are conducted in the following three versions.

1) *Naive Batch EMCM:* Our proposed EMCM algorithms to choose the top k examples in each AL iteration and ignore the information overlap among these selected data points (denoted by N$_k$-EMCM).

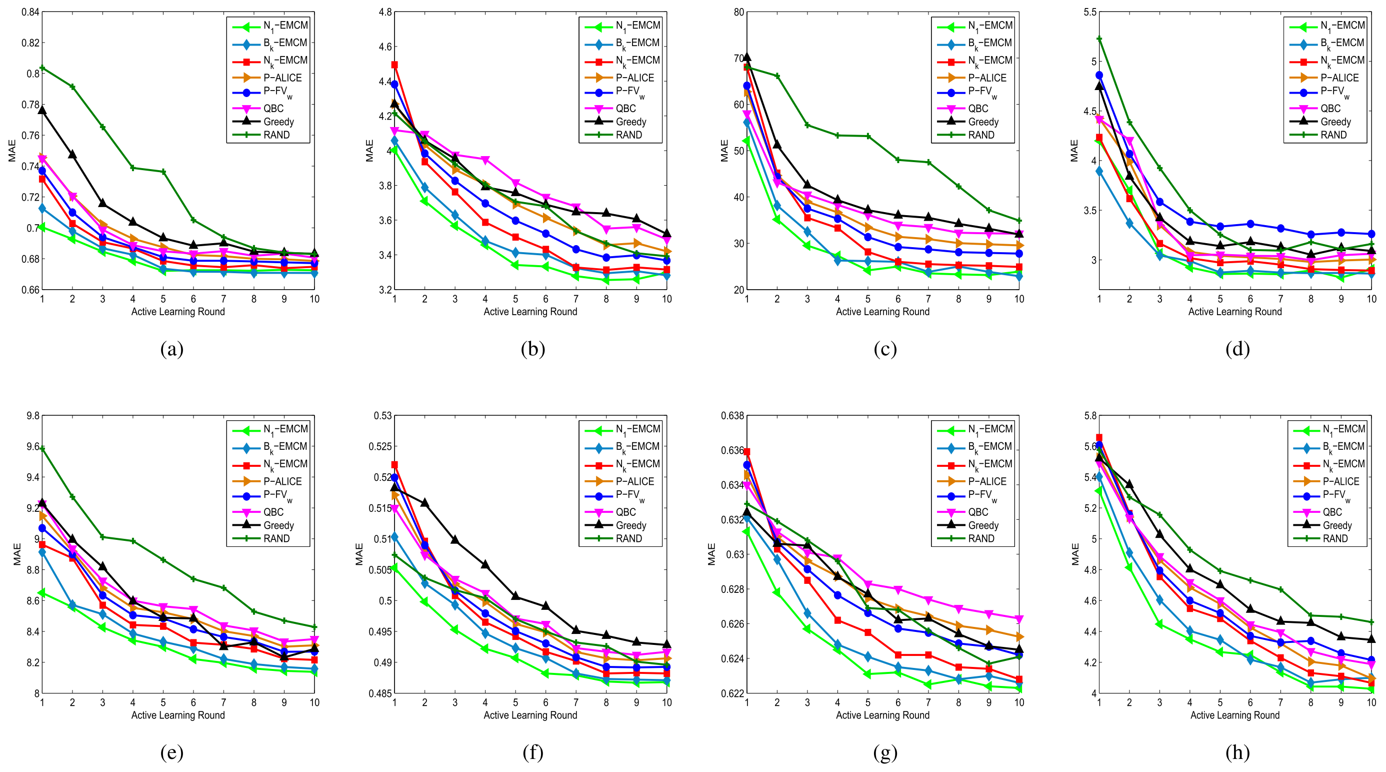


Fig. 1. Experimental results for linear regression on the UCI and StatLib data sets. The evaluation metric used is MAE. (a) PM10. (b) Housing. (c) Forest. (d) CPS. (e) Concrete. (f) Redwine. (g) Whitewine. (h) Bike.

2) *B-EMCM:* Our proposed B-EMCM algorithms to consider the correlation among the selected k examples in the batch (denoted by B$_k$-EMCM).

3) *Sequential EMCM:* The sequential type of EMCM, which our batch mode B$_k$-EMCM algorithms attempt to match, is to retrain the model after every new example is chosen and added to the training set. Hence, sequential EMCM (denoted by N$_1$-EMCM) can be viewed as a special case of N$_k$-EMCM when $k = 1$. The SAL algorithm, which is generally more example-efficient than their batch counterparts, since each data example is chosen with more information, has been regarded as a gold standard in the prior BMAL work [4], [9].

For evaluation, we here use two popular error-based metrics: mean absolute error (MAE) and root mean squared error (RMSE), to verify the performance of each method on the test set

$$\text{MAE} = \frac{1}{|T|}\sum_{i=1}^{|T|} |f(x_i) - y_i| \tag{29}$$

$$\text{RMSE} = \sqrt{\frac{1}{|T|}\sum_{i=1}^{|T|} (f(x_i) - y_i)^2} \tag{30}$$

where $|T|$ denotes the size of the test set, and y_i and $f(x_i)$ are the ground truth and the prediction of the example x_i, respectively. In addition to these error-based metrics, we further employ a correlation-based metric, R-square, to validate the performance of each algorithm

$$R\text{-square} = 1 - \frac{\sum_{i=1}^{|T|}(y_i - f(x_i))^2}{\sum_{i=1}^{|T|}(y_i - \bar{y})^2} \tag{31}$$

where $\bar{y}$ denotes the mean of y_i. To avoid random fluctuation, each experiment is repeated ten times by varying the training-pool-test sets, and the averaged results are reported.

C. Error-Based Experimental Results

The comparisons of the eight sample selection algorithms on these benchmark data sets are presented in Figs. 1–4. The x-axis represents the number of iterations for the AL process, and the y-axis denotes the value of MAE and RMSE. Several general observations, as shown in Figs. 1–4, are explained as follows.

1) In general, we see that both MAE and RMSE decrease when the number of training points increases for all of eight algorithms, which matches the intuition that model performance is positively correlated with the size of training sets.

2) Our derived model change-based methods (N$_1$-EMCM, B$_k$-EMCM, and N$_k$-EMCM) obviously perform better than the five baseline algorithms (i.e., P-ALICE, P-FV$_W$, QBC, greedy, and RAND) in most evaluation points. A possible explanation is that our proposed strategies estimate the model change as the gradient of the squared-error loss, which is directly related to the objective function RMSE and MAE used to evaluate the models. Thus, the examples chosen by these three

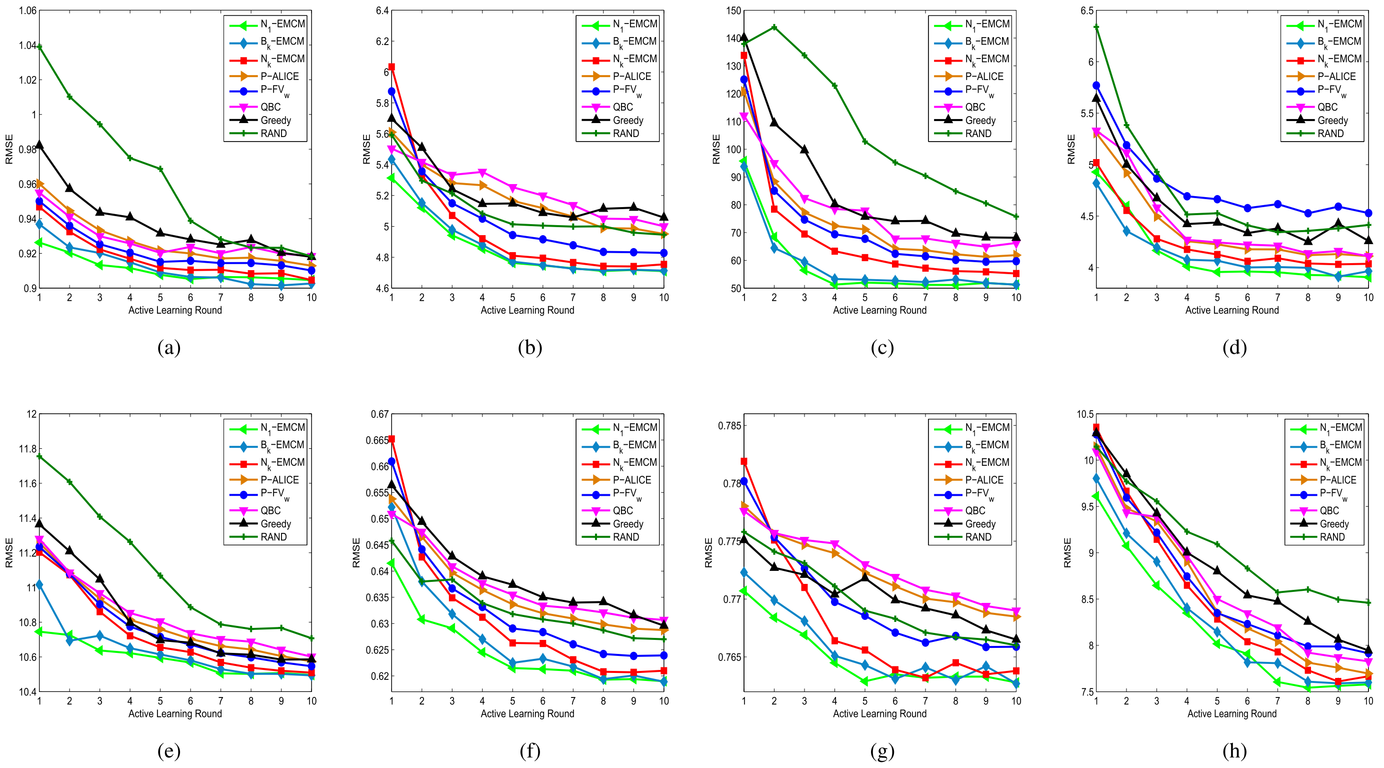


Fig. 2. Experimental results for linear regression on the UCI and StatLib benchmark data sets. The evaluation metric used is RMSE. (a) PM10. (b) Housing. (c) Forest. (d) current population survey. (e) Concrete. (f) Redwine. (g) Whitewine. (h) Bike.

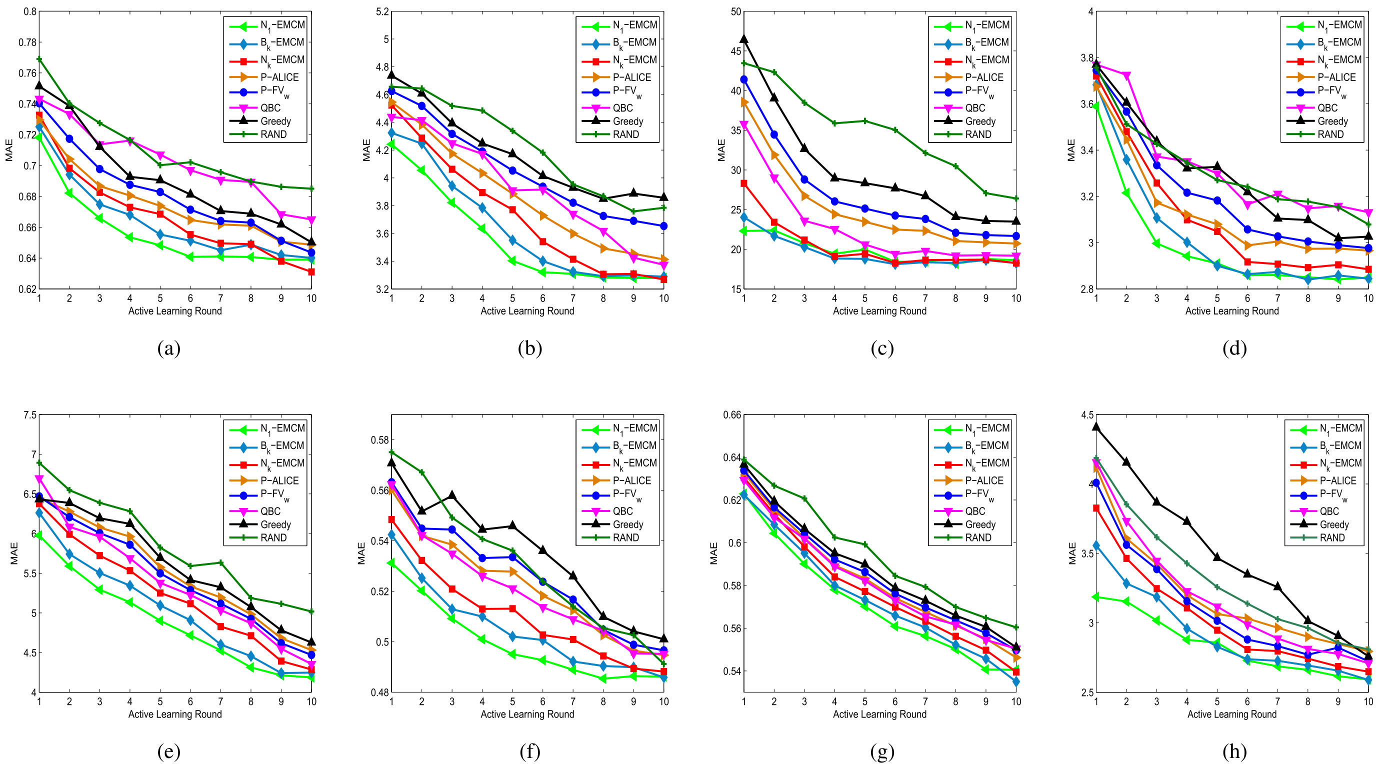


Fig. 3. Experimental results for GBDT regression on the UCI and StatLib benchmark data sets. The evaluation metric used is MAE. (a) PM10. (b) Housing. (c) Forest. (d) CPS. (e) Concrete. (f) Redwine. (g) Whitewine. (h) Bike.

methods are more likely to contribute positively to improve the regression model.

3) The sequential type of EMCM algorithm, N_1-EMCM, is observed to perform the best among the eight methods in most cases during the entire sampling process, demonstrating that the SAL method is more effective in choosing the most informative examples to improve the model quality. This is due to the reason

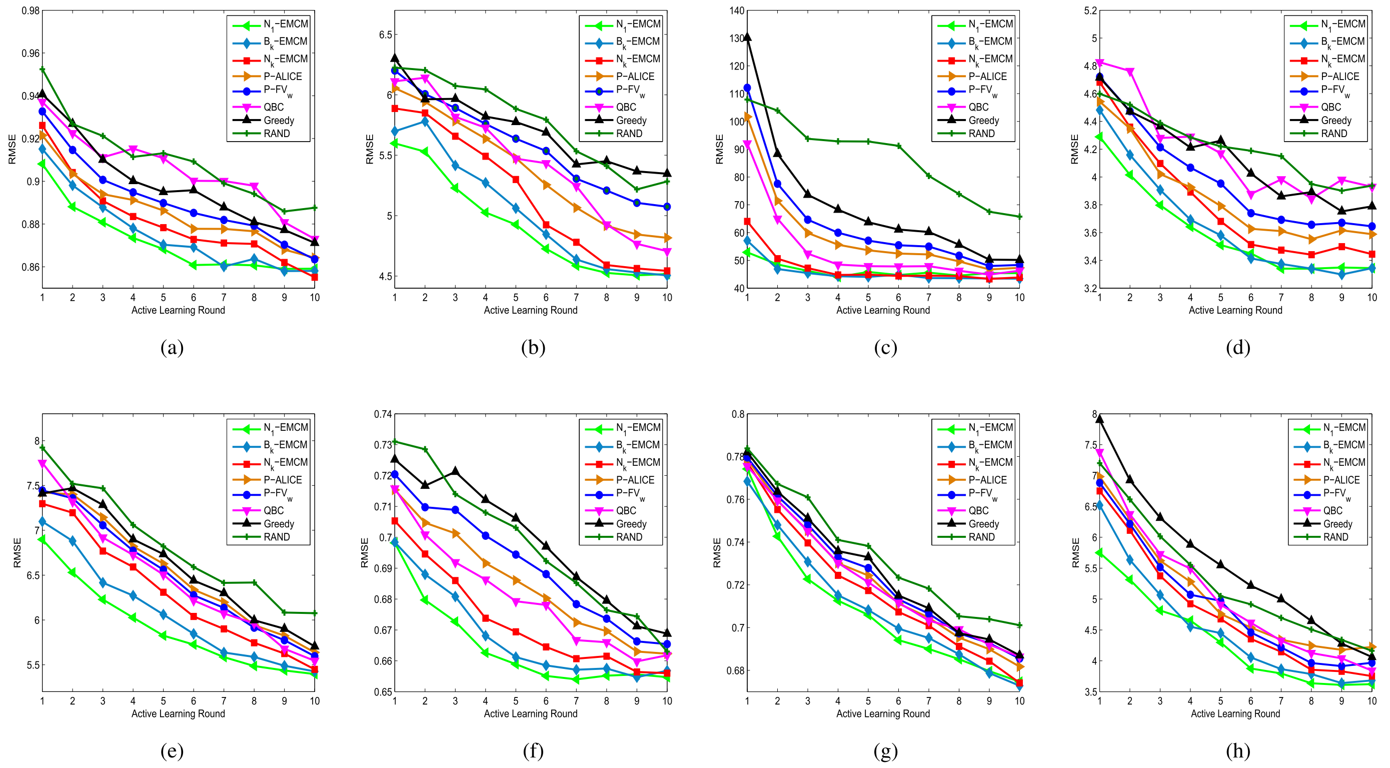


Fig. 4. Experimental results for GBDT regression on the UCI and StatLib benchmark data sets. The evaluation metric used is RMSE. (a) PM10. (b) Housing. (c) Forest. (d) CPS. (e) Concrete. (f) Redwine. (g) Whitewine. (h) Bike.

that the model is updated every new example is chosen and added to the training set, so that each example is selected with more information.

4) For linear regression, we see that N$_k$-EMCM performs slightly poor at the very beginning of AL process on several data sets, e.g., Housing, Redwine, Whitewine, and Bike. After two or three active sampling iterations, the performance of the N$_k$-EMCM method increases quickly and starts to outperform baseline methods. This phenomenon is in fact expected as we discussed in Section III-B, that is, the model performance may be hurt by selecting some outliers at the initial stages of data selection. However, to maximize the change again, N$_k$-EMCM will certainly select good examples in the latter AL round, and hence, the negative influence of the outliers is offset. Since the amount of outliers is usually very restricted, the proposed N$_k$-EMCM approach could perform well with more data queried, which is still very promising in practice.

5) In contrast, the proposed B$_k$-EMCM strategy performs much better than N$_k$-EMCM at the initial stage of AL in most cases. The results may be based on the possible reason that the B$_k$-EMCM takes particular consideration of the relationship among the selected data examples at the batch, and therefore, the negative effect of the outliers could be relieved effectively. Furthermore, we can observe that our proposed B$_k$-EMCM algorithm is able to closely match the performance of N$_1$-EMCM, especially at the latter cycles, indicating that the proposed B$_k$-EMCM method accurately simulates the behavior of the sequential method.

6) Furthermore, unlike the results for linear regression where the performance of the model is decreased greatly at the initial stage of the sampling process [e.g., as presented in Fig. 2(b)], possibly due to the inclusion of the outliers, we can observe that N$_k$-EMCM consistently achieves lower MAE and RMSE scores than the other five sampling methods during the whole data selection process for GBDT-based regression (as shown in Figs. 3 and 4). This is likely due to the reason that GBDT is very robust to outliers.

7) To better verify the effectiveness of our AL algorithms, we carry out significance test on the comparisons. Due to the space constraint, we only present the statistical test in terms of RMSE. Similar results could be obtained for MAE based on our empirical studies. Tables II and III detail the comparisons of one-tailed paired T-test of B$_k$-EMCM and N$_k$-EMCM versus the compared algorithms. Here, win% denotes the percentage of evaluation points where our algorithms (B$_k$-EMCM and N$_k$-EMCM) statistically outperform the baselines at 95% significance level ($p < 0.05$), which is a standard method of using the paired T-test [7]. We can observe that our proposed AL algorithms perform significantly better than the compared methods in most cases.

D. Correlation-Based Experimental Results

Due to the space limit, here we only report the experimental results on four data sets, i.e., Concrete, Redwine, Whitewine, and Bike for the linear regression model in terms of R-square.

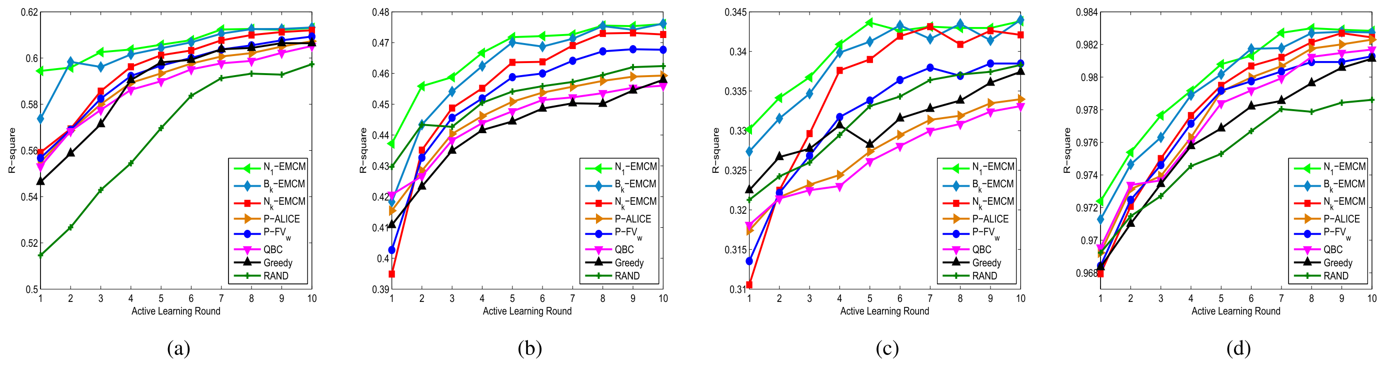


Fig. 5. Experimental results on (a) Concrete, (b) Redwine, (c) Whitewine, and (d) Bike with linear regression in terms of R-square.

TABLE II
WIN% OF B$_k$-EMCM AND N$_k$-EMCM VERSUS THE OTHER ALGORITHMS IN ONE-TAILED PAIRED T-TEST AT 95% SIGNIFICANCE LEVEL ($p < 0.05$) FOR LINEAR REGRESSION IN TERMS OF RMSE

B$_k$-EMCM vs.	P-ALICE	P-FV$_W$	QBC	Greedy	RAND
PM10	90%	60%	90%	100%	100%
Housing	90%	80%	90%	100%	90%
Forest	100%	70%	100%	100%	100%
CPS	60%	100%	50%	90%	100%
Concrete	90%	80%	100%	100%	100%
Redwine	90%	90%	90%	90%	80%
Whitewine	100%	100%	100%	100%	100%
Bike	90%	100%	90%	100%	100%
N$_k$-EMCM vs.	P-ALICE	P-FV$_W$	QBC	Greedy	RAND
PM10	50%	0%	50%	100%	100%
Housing	80%	40%	80%	70%	70%
Forest	40%	10%	60%	90%	90%
CPS	30%	100%	30%	90%	100%
Concrete	50%	30%	50%	30%	100%
Redwine	70%	20%	80%	90%	60%
Whitewine	80%	70%	80%	70%	70%
Bike	40%	50%	70%	90%	80%

TABLE III
WIN% OF B$_k$-EMCM AND N$_k$-EMCM VERSUS THE OTHER ALGORITHMS IN ONE-TAILED PAIRED T-TEST AT 95% SIGNIFICANCE LEVEL ($p < 0.05$) FOR GBDT REGRESSION IN TERMS OF RMSE

B$_k$-EMCM vs.	P-ALICE	P-FV$_W$	QBC	Greedy	RAND
PM10	60%	90%	100%	100%	100%
Housing	90%	100%	100%	90%	100%
Forest	70%	80%	30%	100%	100%
CPS	70%	100%	100%	100%	90%
Concrete	100%	90%	90%	100%	100%
Redwine	90%	100%	80%	100%	90%
Whitewine	80%	100%	90%	100%	100%
Bike	100%	90%	100%	100%	100%
N$_k$-EMCM vs.	P-ALICE	P-FV$_W$	QBC	Greedy	RAND
PM10	40%	70%	100%	100%	100%
Housing	50%	60%	70%	90%	100%
Forest	70%	80%	20%	100%	100%
CPS	30%	70%	100%	80%	80%
Concrete	50%	40%	20%	80%	100%
Redwine	70%	100%	30%	100%	90%
Whitewine	10%	40%	30%	50%	90%
Bike	60%	20%	70%	100%	100%

Fig. 5 shows the results on these data sets. We can see that our proposed algorithms still perform better than the baselines, demonstrating the effectiveness of our proposed AL algorithms. Similar results could be obtained on other data sets based on our empirical studies, which are omitted here.

E. Time Performance

In this section, we first analyze the computation complexity. Then, we compare the averaged CPU running time to examine the efficiency of our proposed AL algorithms.

As the amount of unlabeled examples is overwhelming over the labeled examples in AL, we only consider the time required by the informativeness computation with respect to unlabeled examples and exclude the data selection time, which is identical to the previous work [39]. Assume that there are m unlabeled examples in the pool set, and $x^+ \in \mathbb{R}^d$ is a d-dimensional feature vector. Recall that Z is the size of bootstrap for predictive distribution estimation and M is the number of individual trees in GBDT. For linear regression, the computation time needed by our N$_k$-EMCM is $O(Zmd)$. For B$_k$-EMCM, the time complexity is $O(kZmd)$ if we choose a batch of k examples. Because k is a proportion of the unlabeled examples, the cost of B$_k$-EMCM is actually $O(Zm^2d)$, indicating that the time increase is superlinear. For GBDT regression, due to the added step of feature mapping, the time complexity for N$_k$-EMCM and B$_k$-EMCM are $O((Z+d)mM)$ and $O((Zm+d)mM)$, respectively.

Now, we compare the CPU run time taken by our proposed algorithms versus the compared approaches. Due to the space limitation, we only present the comparisons of our proposed algorithms with linear regression. Here, we simply fix the batch size as 10. All algorithms were implemented using C++ in the Linux environment on a standard desktop PC with 2.27-GHz CPU (Intel Xeon) and 4 GB of memory. Table IV presents the comparison results, together with the size of the pool set. As compared in Table IV, the CPU running time of N$_k$-EMCM is comparable with QBC and greedy, but much more efficient than the other two competitors, i.e., P-ALICE and P-FV$_W$. In addition, we see that the time cost for B$_k$-EMCM is relatively higher than that of the N$_k$-EMCM counterparts. This is due to the reason that

TABLE IV
CPU RUNNING TIME FOR LINEAR REGRESSION (IN SECONDS). THE BATCH SIZE $k = 10$

Data sets	# Examples $\times$ Features ($\mathcal{U}$)	N$_1$-EMCM	B$_k$-EMCM	N$_k$-EMCM	P-ALICE	P-FV$_W$	QBC	Greedy
PM10	350 $\times$ 7	3.16	0.93	0.22	1.78	1.82	0.33	0.21
Housing	356 $\times$ 13	3.70	1.11	0.27	2.19	2.30	0.39	0.29
Forest	360 $\times$ 10	3.69	1.19	0.31	2.21	2.38	0.35	0.28
CPS	375 $\times$ 8	3.52	1.10	0.33	1.85	1.90	0.37	0.29
Concrete	730 $\times$ 8	5.16	1.19	0.34	2.32	2.45	0.55	0.32
Redwine	1119 $\times$ 11	7.21	2.58	0.56	5.62	6.03	0.65	0.52
Whitewine	3428 $\times$ 11	17.47	7.41	1.12	8.30	10.88	1.45	1.82
Bike	12165 $\times$ 14	92.25	39.61	5.37	51.82	59.46	6.76	6.81

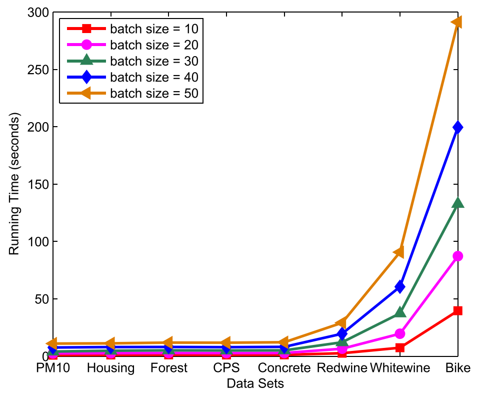


Fig. 6. Runtime of B$_k$-EMCM for linear regression.

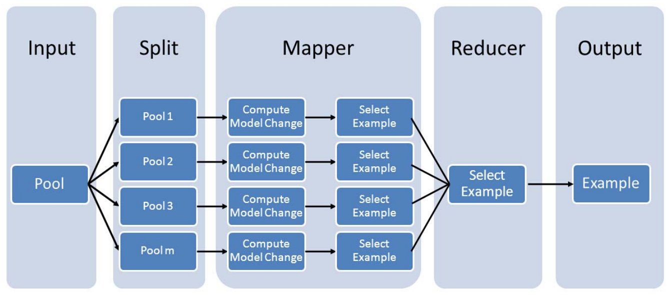


Fig. 7. Model change-based AL via MapReduce, and the model change computation is parallelized.

B$_k$-EMCM has to spend additional time in calculating the relationship among the selected examples in the set to reduce data redundancy. Finally, N$_1$-EMCM is observed to have the highest time complexity, since it has to retrain the model every new example is added to the training set, which matches our intuition.

For the proposed B$_k$-EMCM methods, we investigate their scalability by varying the size of batch from 10 to 50. Fig. 6 shows the CPU run time versus data sets, varying the sizes of the batch. We see that the time increase for B$_k$-EMCM appears superlinear with the required size of the batch, especially on our largest data set (i.e., Bike), which certainly matches our time complexity analysis in the above. Here, the running time varies from 1 to 290 s.

We here discuss the usability of our B$_k$-EMCM algorithms in the industrial environment. Although the time cost increases greatly with the batch size k, based on our real industry experiences, it still can be acceptable in many practical applications. First, in practice, a small fixed value of k might be sufficient to obtain satisfied results, in which case the batch size k becomes a constant and the time increase becomes linear again. Second, because the model change for the jth candidate data example could be independently calculated of each other, we can use multiple machines to run the algorithms in parallel with the Hadoop–MapReduce platform, and therefore, the computational time can be saved significantly. Fig. 7 shows our model change-based AL using MapReduce. The process is as follows. To choose the jth example, every mapper returns the top-1 data example inside its pool and the model change caused by this example. The reducer reads all these top-1 examples from different mappers and selects the final example x_j^*, which has the maximum model change. Then, the Master deletes x_j^* from the pool and starts to select the $(j+1)$th example.

VII. EXTENSIONS

The proposed expected model change-based AL framework is flexible, which could be generalized to other base learners. As the essential step of our proposed EMCM framework is to accurately estimate the model change, it is naturally applicable to the SGD-based model. A special case here is the regularized model, such as ridge regression. Based on the conclusions of the previous work [22], we can still approximate the model change as the gradient of the loss function at the new example.

However, applying EMCM to the non-SGD models is more complex. The key challenge lies in the measurement of model change, and hence, we need to carefully design the AL methods, taking particular consideration of the unique characteristics of learners. Because we target on AL in the context of regression in this paper, we generalize our proposed EMCM framework to GP regression here.

GP regression is a kernel approach, which can be formulated as the following model on the training set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$:

$$f(x;\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i \kappa(x_i, x) \tag{32}$$

where κ is a given kernel function. The weight vector (parameter) $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_n]^T$ can be obtained as

$$\boldsymbol{\alpha} = \left(\mathbf{K} + \sigma_n^2 \mathbf{I}\right)^{-1} \mathbf{Y} \tag{33}$$

where $\mathbf{K}$ and $\mathbf{I}$ are the kernel matrix of the training set $\mathcal{D}$ and the identity matrix, respectively. σ_n^2 is the assumed noise variance and $\mathbf{Y}$ is the label vector.

Similar to above, suppose a candidate data point x^+ is added to the training set with a given label y^+, the updated parameter vector $\boldsymbol{\alpha}_*$ can be computed with the closed form

$$\boldsymbol{\alpha}_* = \begin{bmatrix} \boldsymbol{\alpha} \\ 0 \end{bmatrix} + \frac{\boldsymbol{\kappa}^T\boldsymbol{\alpha} - y^+}{\sigma_{f_*}^2 + \sigma_n^2} \begin{bmatrix} \left(\mathbf{K} + \sigma_n^2 \mathbf{I}\right)^{-1}\boldsymbol{\kappa} \\ -1 \end{bmatrix} \tag{34}$$

where $\boldsymbol{\kappa}$ is the vector of pairwise kernel values of the training set and x^+, and $\sigma_{f_*}^2$ is the predictive variance of x^+. A detailed proof of this closed form can be found in [40].

Again, we can approximate the true model change with the expectation calculation, and the AL criterion is

$$x^* = \arg\max_{x \in \mathcal{U}} \int_Y ||\boldsymbol{\alpha}_* - [\boldsymbol{\alpha}^T, 0]^T|| P(y|x) dy. \tag{35}$$

In general, our EMCM framework is generic, which could be generalized to the non-SGD-based models. The key challenge lies in getting the closed-form solution of model change.

VIII. Conclusion and Future Work

In this paper, a novel AL framework, EMCM, is derived for regression. In light of the SGD learning rule, we use the derivative of the loss function to estimate the model change. Under the EMCM framework, we develop novel AL algorithms for both linear regression and nonlinear GBDT regression, which aim to select the example that leads to the maximum model change. Furthermore, by simulating the behavior of the SAL algorithms when applied for k iterations, we extend EMCM to BMAL algorithms (B-EMCM) to simultaneously choose a set of k most informative instances at each query time. Extensive experimental results on both UCI and StatLib benchmark data sets have demonstrated the effectiveness and the efficiency of our proposed algorithms.

The proposed EMCM framework is flexible. We also extend it to the GP regression in order to show its usability in a wide range of applications.

As our proposed B-EMCM methods are the approximations of the SAL counterparts, a potential limitation is that the estimate accuracy in the model change may decrease with the increase in the size of the batch, resulting in error accumulation. One possible solution to this problem is to adaptively determine the batch size taking particular consideration of the loss discrepancy between the approximated model and the true model trained after every newly selected data example. We will leave this as our future work. We also plan to derive some solid theoretical foundations on the proposed algorithms, including label complexity bounds and generalization error bounds.

References

[1] K. Brinker, "Incorporating diversity in active learning with support vector machines," in *Proc. 20th Int. Conf. Mach. Learn. (ICML)*, 2003, pp. 59–66.

[2] S. C. H. Hoi, R. Jin, and M. R. Lyu, "Large-scale text categorization by batch mode active learning," in *Proc. 15th Int. World Wide Web Conf. (WWW)*, 2006, pp. 633–642.

[3] S. C. H. Hoi, R. Jin, J. Zhu, and M. R. Lyu, "Batch mode active learning and its application to medical image classification," in *Proc. 23rd Int. Conf. Mach. Learn. (ICML)*, 2006, pp. 417–424.

[4] S. C. H. Hoi, R. Jin, and M. R. Lyu, "Batch mode active learning with applications to text categorization and image retrieval," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1233–1248, Sep. 2009.

[5] Y. Guo and D. Schuurmans, "Discriminative batch mode active learning," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2007, pp. 593–600.

[6] I. Zliobaite, A. Bifet, B. Pfahringer, and G. Holmes, "Active learning with drifting streaming data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 1, pp. 27–39, Jan. 2014.

[7] Y. Guo, "Active instance sampling via matrix partition," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2007, pp. 802–810.

[8] R. Chattopadhyay, Z. Wang, W. Fan, I. Davidson, S. Panchanathan, and J. Ye, "Batch mode active sampling based on marginal probability distribution matching," in *Proc. 18th ACM Int. SIGKDD Conf. KDD*, 2012, pp. 741–749.

[9] J. Azimi, A. Fern, X. Z. Fern, and G. Borradaile, "Batch active learning via coordinated matching," in *Proc. 29th Int. Conf. Mach. Learn. (ICML)*, 2012, pp. 1–8.

[10] R. Castro, R. Willett, and R. Nowak, "Faster rates in regression via active learning," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2005, pp. 179–186.

[11] D. D. Lewis and W. A. Gale, "A sequential algorithm for training text classifiers," in *Proc. 17th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr. (SIGIR)*, 1994, pp. 3–12.

[12] M. Sugiyama, "Active learning in approximately linear regression based on conditional expectation of generalization error," *J. Mach. Learn. Res.*, vol. 7, pp. 141–166, Jan. 2006.

[13] M. Sugiyama and S. Nakajima, "Pool-based active learning in approximate linear regression," *Mach. Learn.*, vol. 75, no. 3, pp. 249–274, 2009.

[14] D. A. Cohn, Z. Ghahramani, and M. I. Jordan, "Active learning with statistical models," *J. Artif. Intell. Res.*, vol. 4, pp. 129–145, Mar. 1996.

[15] O. Chapelle, P. Shivaswamy, S. Vadrevu, K. Weinberger, Y. Zhang, and B. Tseng, "Boosted multi-task learning," *Mach. Learn.*, vol. 85, nos. 1–2, pp. 149–173, 2011.

[16] M. Lin, K. Tang, and X. Yao, "Dynamic sampling approach to training neural networks for multiclass imbalance classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 4, pp. 647–660, Apr. 2013.

[17] H. Yu and S. Kim, "Passive sampling for regression," in *Proc. 10th Int. Conf. Data Mining (ICDM)*, 2010, pp. 1151–1156.

[18] W. Cai, Y. Zhang, and J. Zhou, "Maximizing expected model change for active learning in regression," in *Proc. 13th Int. Conf. Data Mining (ICDM)*, 2013, pp. 51–60.

[19] B. Settles, M. Craven, and S. Ray, "Multiple-instance active learning," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2008, pp. 1289–1296.

[20] B. Settles and M. Craven, "An analysis of active learning strategies for sequence labeling tasks," in *Proc. Int. Conf. Emprical Methods Natural Lang. Process. (EMNLP)*, 2008, pp. 1070–1079.

[21] P. Campigotto, A. Passerini, and R. Battiti, "Active learning of Pareto fronts," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 3, pp. 506–519, Mar. 2014.

[22] P. Donmez and J. G. Carbonell, "Optimizing estimated loss reduction for active sampling in rank learning," in *Proc. 25th Int. Conf. Mach. Learn. (ICML)*, 2008, pp. 248–255.

[23] Y. Freund, H. S. Seung, E. Shamir, and N. Tishby, "Selective sampling using the query by committee algorithm," *Mach. Learn.*, vol. 28, no. 2, pp. 133–168, 1997.

[24] X.-Y. Zhang, S. Wang, and X. Yun, "Bidirectional active learning: A two-way exploration into unlabeled and labeled data set," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 12, pp. 3034–3044, Nov. 2015.

[25] S. Tong and D. Koller, "Support vector machine active learning with applications to text classification," *J. Mach. Learn. Res.*, vol. 2, pp. 45–66, Mar. 2001.

[26] R. Burbidge, J. J. Rowland, and R. D. King, "Active learning for regression based on query by committee," in *Proc. 8th Int. Conf. Intell. Data Eng. Autom. Learn. (IDEAL)*, 2007, pp. 209–218.

[27] I. Dagan and S. P. Engelson, "Committee-based sampling for training probabilistic classifiers," in *Proc. 12th Int. Conf. Mach. Learn. (ICML)*, 1995, pp. 150–157.

[28] N. Abe and H. Mamitsuka, "Query learning strategies using boosting and bagging," in *Proc. 15th Int. Conf. Mach. Learn. (ICML)*, 1998, pp. 1–9.

[29] P. Melville and R. J. Mooney, "Diverse ensembles for active learning," in *Proc. 21st Int. Conf. Mach. Learn. (ICML)*, 2004, p. 74.

[30] E. J. de Fortuny and D. Martens, "Active learning-based pedagogical rule extraction," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 11, pp. 2664–2677, Nov. 2015.

[31] Z. Lu, X. Wu, and J. Bongard, "Active learning with adaptive heterogeneous ensembles," in *Proc. 9th Int. Conf. Data Mining (ICDM)*, 2009, pp. 327–336.

[32] B. Settles, *Active Learning*. San Rafael, CA, USA: Morgan & Claypool, 2012.

[33] N. Roy and A. McCallum, "Toward optimal active learning through sampling estimation of error reduction," in *Proc. 18th Int. Conf. Mach. Learn. (ICML)*, 2001, pp. 441–448.

[34] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Ann. Statist.*, vol. 29, no. 5, pp. 1189–1232, 2001.

[35] T. Fushiki, "Bootstrap prediction and Bayesian prediction under misspecified models," *Bernoulli*, vol. 11, no. 4, pp. 747–758, 2005.

[36] H. T. Nguyen and A. Smeulders, "Active learning using pre-clustering," in *Proc. 21st Int. Conf. Mach. Learn. (ICML)*, 2004, p. 79.

[37] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning* (Springer Series in Statistics). New York, NY, USA: Springer, 2001.

[38] S. Chakraborty, V. Balasubramanian, and S. Panchanathan, "Adaptive batch mode active learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 8, pp. 1747–1760, Aug. 2015.

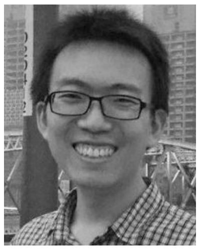
[39] A. I. Schein and L. H. Ungar, "Active learning for logistic regression: An evaluation," *Mach. Learn.*, vol. 68, no. 3, pp. 235–265, 2007.

[40] A. Freytag, E. Rodner, P. Bodesheim, and J. Denzler, "Labeling examples that matter: Relevance-based active learning with Gaussian processes," in *Proc. 35th German Conf. Pattern Recognit. (GCPR)*, 2013, pp. 282–291.

[41] G. Dong and V. Taslimitehrani, "Pattern-aided regression modeling and prediction model analysis," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 9, pp. 2452–2465, Sep. 2015.

Wenbin Cai (S'14) received the Ph.D. degree in information and communication engineering from Shanghai Jiao Tong University, Shanghai, China, in 2016.

He is currently a Software Engineer with the Microsoft Search Technology Center Asia, Beijing, China. His current research interests include machine learning and information retrieval, focusing on active learning and its applications to classification, regression, bot detection, cost-sensitive learning, and rank learning.



Muhan Zhang received the B.E. (Hons.) degree in information engineering from Shanghai Jiao Tong University, Shanghai, China, in 2015. He is currently pursuing the Ph.D. degree in computer science with Washington University in St. Louis, St. Louis, MO, USA.

His current research interests include machine learning, data mining, and their interdisciplinary applications.

Ya Zhang (M'05) received the B.S. degree from Tsinghua University, Beijing, China, and Ph.D. degree in information sciences and technology from Pennsylvania State University, State College, PA, USA.

She was with the Lawrence Berkeley National Laboratory, University of Kansas, Lawrence, KS, USA, and Yahoo! Laboratories, Sunnyvale, CA, USA. She has been a Professor with the Cooperative Medianet Innovation Center, Shanghai Jiao Tong University, Shanghai, China, since 2010. Her current research interests include data mining and machine learning, with applications to information retrieval, Web mining, and multimedia analysis.