Neural Processing Letters (2021) 53:1945–1962
https://doi.org/10.1007/s11063-021-10476-z

Pre-Training Acquisition Functions by Deep Reinforcement Learning for Fixed Budget Active Learning

Yusuke Taguchi[1] · **Hideitsu Hino**[2] · **Keisuke Kameyama**[1]

Accepted: 26 February 2021 / Published online: 21 March 2021

Abstract
There are many situations in supervised learning where the acquisition of data is very expensive and sometimes determined by a user's budget. One way to address this limitation is active learning. In this study, we focus on a fixed budget regime and propose a novel active learning algorithm for the pool-based active learning problem. The proposed method performs active learning with a pre-trained acquisition function so that the maximum performance can be achieved when the number of data that can be acquired is fixed. To implement this active learning algorithm, the proposed method uses reinforcement learning based on deep neural networks as as a pre-trained acquisition function tailored for the fixed budget situation. By using the pre-trained deep Q-learning-based acquisition function, we can realize the active learner which selects a sample for annotation from the pool of unlabeled samples taking the fixed-budget situation into account. The proposed method is experimentally shown to be comparable with or superior to existing active learning methods, suggesting the effectiveness of the proposed approach for the fixed-budget active learning.

Keywords Active learning · Reinforcement learning · Deep Q-Learning · Uncertainty sampling · Budget aware learning

Part of this work is supported by NEDO Grant No. JPNP18002, JST CREST JPMJCR1761 and Mirai Program Grant Number JPMJMI19G1.

✉ Hideitsu Hino
hino@ism.ac.jp

Yusuke Taguchi
s1820665@s.tsukuba.ac.jp

Keisuke Kameyama
kameyama@cc.tsukuba.ac.jp

[1] University of Tsukuba, Ibaraki, Japan

[2] The Institute of Statistical Mathematics, Tokyo, Japan

1 Introduction

In the framework of supervised learning, the predictive performance of a learned model should improve when the number of samples increases. However, it is not desirable to simply increase the number of samples when the cost of annotation or acquiring labels is high. Examples of such cases include problems that require expert knowledge for annotation as well as problems that require large-scale experiments. Also, the effect of each set of data on the training of the predictive model is not equal. Some data hardly change the model, and some data can deteriorate the performance of the model. Therefore, it is necessary to selectively annotate data that is considered useful for the model. To maximize the performance of the model, a framework for selective annotation called active learning [19,38] has been developed. Active learning methods are roughly divided into two categories: pool-based and stream-based methods. Pool-based methods assume that there is a large amount of data (called pool data) that consists of features only (i.e., without labels or response values), and the active learner selects which datum should be annotated next. The annotated datum is then included in the training dataset. In the next step of active learning, the learner then selects a datum from the remainder of the pool dataset. Stream-based active learning is a method that deals with the problem of determining whether to annotate sequentially given data individually. In this work, we focus on pool-based active learning.

In active learning, the most critical issue is how to design the *acquisition function* used for determining which datum to annotate in the current circumstances. The majority of existing approaches adopt some kind of measure of the difficulty of prediction as the sample selection criterion. These criteria often give good results empirically, but the same criterion is used throughout the active learning process in these methods. The best strategy for designing the acquisition function would differ according to *context*. In most of the conventional active learning methods, a fixed single selection criterion is used throughout the learning process, and the selected data may include outliers and other such samples that degrade the performance of the model. For example, uncertainty sampling [30] selects the data with the smallest discriminant posterior probability. In other words, the most difficult data for the current learning model is selected. However, this method may select outliers on the discrimination boundary. In order to avoid such a situation, it is necessary to change the criteria flexibly according to the context of learning process.

Suppose we want to learn a prediction model when *a budget is fixed in advance*, namely, the number of data to be labeled is pre-determined, which is an extremely common situation when developing a machine learning system. In this case, a better model should be obtained if we acquire the labeled data in a manner that considers the data acquisition order or context within the budget. To meet these demands, we propose a method that applies reinforcement learning [41] to active learning. The use of reinforcement learning for learning acquisition function used in active learning is already considered in [49]. However, the method proposed in [49] is designed only for classification problems and does not consider budget of learner. Considering the context within which the data is acquired, data is selected according to an appropriate criterion that reflects the current state of the learning model so that the model performance is maximized when the specified number of data is acquired. For this purpose, a deep Q-network (DQN) [32] is used to learn an acquisition function that takes the data context into account.

The major contributions of this paper is summarized as follows:

– We tackled the problem of learning the acquisition function suitable for fixed-budget active learning problem. Recent studies on active learning focus on the data-driven acqui-

sition function design, but, to the best of the authors knowledge, the acquisition function tailored for the fixed-budget situation is not investigated yet.
– To realize learning the fixed-budget acquisition function, we utilized the reinforcement learning. In particular, we adopt the DQN and the acquisition function which is trained in advance of the operational phase of the active learning. By using the reinforcement learning, the training of the acquisition function is done so as to select appropriate samples where the number of available samples is fixed.

The rest of the paper is organized as follows. In Sect. 2, related works on active learning and recent results on learning acquisition functions from data are summarized. Section 3 introduces the notion of reinforcement learning and its modern implementation, the deep Q-networks. Then, our proposed approach for the fixed budget active learning is presented in Sect. 4, and it is experimentally evaluated in Sect. 5. The last section is devoted to the discussion and conclusion.

2 Related Work

Active learning has a long history and is still actively researched [18,20,25] and applied variety of problems [9,31,35,36,40,42,44]. However, in most of the literature, the criterion used to select data from the pool data does not change according to the environment or context, so if the criterion is not appropriate for the current status of the learning model or pool data, the selected dataset will not improve the predictive model as expected.

Several recent studies have employed the meta-active learning approach, which aims to learn the acquisition function for the active learning from the dataset [3,14,28,33,47,50]. The authors of [50] and [3] applied the active learning strategy to one-shot learning. Active one-shot learning [50] is designed for stream-based active learning, in which reinforcement learning is adopted to learn whether to label a given sample or to ignore it. The method in [3] is a modified version of [50] for pool-based active learning. These methods utilize reinforcement learning [10,41] to learn the acquisition function. In particular, [50] is similar to our proposed method in that it uses a DQN. The method proposed in [50] has the advantage of being able to learn the environment with high precision using a DQN. However, because it is designed for cases in which an extremely small subset of pool data should be labeled, it is not suitable for a standard active learning problem. Also, [12] utilize the DQN for learning an appropriate acquisition function, but its formulation is heavily dependent on the Markov decision process, meaning that it is only applicable to stream-based active learning, while we consider the pool-based active learning in this study.

Recently, a methodology called learning active learning (LAL) was proposed in [28], in which the acquisition function for active learning is pre-trained before the actual learning phase. In the pre-training stage, a large number of datasets are collected. These datasets can be collected from other problem domains or even be artificially generated. Then, *features* are designed using the dataset and the predictive model to be learned. For example, the distances between a candidate datum to be annotated and its k-nearest neighbor data in the pool, the coefficients of a linear classifier (predictive model), or the average depth of random forest classifiers could be employed. These features are calculated and stacked to form a feature vector, and the feature vectors are used to train an acquisition function for improving the prediction model. Then, the trained acquisition function is used in the predictive model learning phase, in which the same feature vectors are extracted from the actual data and the predictive model.

A method that takes the feature extraction performed by LAL one step further by performing feature engineering has also been proposed [33]. Using reinforcement learning, the acquisition function for the embedded features is trained. This method is designed to perform the feature embedding and learn the acquisition function in an end-to-end manner. However, the predictive model for the method is currently limited to a two-class linear support vector machine [46]. Though in this paper we concentrate on classification setting, our method is applicable to both classification and regression settings.

In this study, we propose a method to pre-train the acquisition function for active learning by using deep Q-network with datasets from other domains. The method enables us to select data to be annotated according to the context of the learning process of a predictive model. For the predictive model, we adopt random forest [6], which can realize multi-class classification and regression in a unified framework, and it is easy to extract different kind of features from the trained model as explained in Sect. 4, but other predictive models can be plugged into our method. We note that even deep learning models can be used for classification in active learning, but its hypothesis space is too large and has its own difficulty [23] when applied to active learning.

We consider fixed budget regime in this study, and consider the optimal acquisition function under this circumstance, which is the main difference between existing work for learning acquisition function for active learning [28,33]. We note that in some cases, we run active learning algorithm without explicit limitation or budget for data annotation. In such cases, we encounter another problem, namely, when to stop learning. There are several works on the optimal stopping timing of active learning [2,4,22,26]. There are only few works in the literature of active learning in which the budget is explicitly considered [7,13], where the authors derived a budget aware *stream-based* active learning, which do not consider learning the acquisition function from data.

3 Preliminary for Reinforcement Learning

Our active learning model uses a pre-trained acquisition function, which is learned by reinforcement learning. There are many possibilities for implementing reinforcement learning and our main idea does not assume any specific realization of reinforcement learning. One of the modern and promising approaches is that based on the deep neural networks. In particular, DQN [32] is used to account for dynamic phenomena where time is explicitly involved. The consecutive data acquisition process corresponds to the notion of "time", and DQN is shown to work well in the literature of learning acquisition function for active learning [12,50].

In this section, we introduce a reinforcement learning method based on DQN to realize active learning in consideration of the context of data acquisition.

3.1 Q-Learning

This subsection presents an overview of Q-learning, which is a representative reinforcement-learning method. Reinforcement learning [41] is a field of machine learning in which an agent tries to maximize reward by taking *actions* under a *state* in which the agent is located. Q-learning is a theoretically sound reinforcement learning methods that has been empirically shown to perform well.

The aim in Q-learning is to obtain a function that calculates value for taking action $\boldsymbol{a}_t$ when a learner or an agent is in state $\boldsymbol{s}_t$ at a certain time t. Here, $\boldsymbol{s}_t$ is a collection of parameters

that specify the state or current situation of the agent, and $\boldsymbol{a}_t$ is a collection of parameters that represent the action that is taken in the current situation. We define a Q-function that accepts $(\boldsymbol{s}_t, \boldsymbol{a}_t)$ and outputs a reward. With this function, an action that maximizes the reward is taken. The ideal Q-function $Q^*(\boldsymbol{s}_t, \boldsymbol{a}_t)$ is defined by

$$Q^*(\boldsymbol{s}_t, \boldsymbol{a}_t) = \max_{\pi} \mathbb{E}\left[R_t | \boldsymbol{s}_t, \boldsymbol{a}_t, \pi\right]. \tag{1}$$

Here, π is a mapping from state $\boldsymbol{s}$ to action $\boldsymbol{a}$; that is, it is the strategy for deciding which action to take. In other words, $Q^*(\boldsymbol{s}, \boldsymbol{a})$ is the expectation of the obtained reward when the agent takes the ideal action $\boldsymbol{a}$ in a certain situation $\boldsymbol{s}$.

Reward R_t is composed of the cumulative immediate rewards from time $t = 0$ to $t = T$ with a discount factor as follows:

$$R_t = \sum_{\tau=t}^{T} \gamma^{\tau - t} r_\tau, \tag{2}$$

where T is the time in the final state, $\gamma \in [0, 1]$ represents the discount rate, and r_τ represents the immediate reward at time $t = \tau$. Applying Eq. (2) to Eq. (1) yields the following expression:

$$Q^*(\boldsymbol{s}_t, \boldsymbol{a}_t) = \mathbb{E}_{\boldsymbol{s}_{t+1} \sim \mathcal{P}_{t+1}} \left[r + \gamma \max_{\boldsymbol{a}_{t+1}} Q(\boldsymbol{s}_{t+1}, \boldsymbol{a}_{t+1}) \mid \boldsymbol{s}_t, \boldsymbol{a}_t \right],$$

where $\mathcal{P}_{t+1}$ is the probability distribution of state $\boldsymbol{s}_{t+1}$. The action $\boldsymbol{a}^*$ that maximizes the reward at time t constitute the sequence of actions $\{\boldsymbol{a}^*_\tau\}_{\tau=t}^{T}$, and in this sense, reinforcement learning considers the *context* of learning.

In Q-learning, the ideal Q-function in Eq. (1) is not available because of the lack of the ground truth distribution of R_t. As a result, the Q-function is typically represented as a table created from discrete states $\boldsymbol{s}$ and discrete actions $\boldsymbol{a}$. The value (reward) of each cell in this table is updated by the following formula:

$$\begin{aligned} Q(\boldsymbol{s}_{t+1}, \boldsymbol{a}_{t+1}) = \; & Q(\boldsymbol{s}_t, \boldsymbol{a}_t) \\ & + \alpha (r_{t+1} + \gamma \max_{\boldsymbol{a}} Q(\boldsymbol{s}_{t+1}, \boldsymbol{a}) - Q(\boldsymbol{s}_t, \boldsymbol{a}_t)), \end{aligned}$$

where $\alpha \in [0, 1]$ is a parameter indicating the learning rate. There are various ways to select the action during learning, but here we use a simple method called the ϵ-greedy method [41]. The ϵ-greedy method is a strategy that takes a random action with a probability of ϵ and maximizes the Q-function currently obtained with a probability of $1 - \epsilon$.

3.2 Deep Q-Network

A Deep Q-network (DQN) uses a deep neural network [15,29] instead of a table for representing the Q-function; hence, unseen states and actions can be evaluated. Here, we approximate the Q-function using a deep neural network determined by the parameter $\boldsymbol{\theta}$ as $Q^*(\boldsymbol{s}, \boldsymbol{a}) \approx Q(\boldsymbol{s}, \boldsymbol{a}; \boldsymbol{\theta})$. The DQN starts with a randomly initialized parameter $\boldsymbol{\theta}$ and optimizes the following objective function:

$$L_i(\boldsymbol{\theta}_i) = \mathbb{E}_{\boldsymbol{s}, \boldsymbol{a} \sim \epsilon} \left[\frac{1}{2} (q_i - Q(\boldsymbol{s}, \boldsymbol{a}))^2 \right], \tag{3}$$

where

$$q_i = \mathbb{E}_{s_{t+1}\sim\epsilon}\left[r + \gamma \max_{\boldsymbol{a}_{t+1}} Q(\boldsymbol{s}_{t+1}, \boldsymbol{a}_{t+1}; \boldsymbol{\theta}_{i-1})|\boldsymbol{s}_t, \boldsymbol{a}_t\right].$$

The index i indicates the i-th label annotation on the unlabeled data from the pool dataset. Parameter $\boldsymbol{\theta}_{i+1}$ is then optimized with q_{i+1}, calculated using $\boldsymbol{\theta}_i$, until i reaches a predefined time T, which is the maximum number of data to be acquired as determined by the budget. The final Q-function is the deep neural network determined by $\boldsymbol{\theta}$. The biggest difference between conventional Q-learning and a DQN is that conventional Q-learning treats the states and actions as discrete, whereas a DQN treats states and actions as continuous values. We use a DQN in active learning by designing the state and behavior appropriately because of its high empirical performance.

4 Proposed Method

This section describes the procedure for applying DQN to active learning. The framework is similar to that of [28], which considers the reduction of loss for the predictive model when data is added to the training set, and learns a predictor of the reduction by using datasets from other domains.

Learning the acquisition function would improve the performance of active learning. However, active learning is in general used under scarce data regime, and we cannot expect the acquisition function learned with a small number of data generalize well. In the proposed method, we use a large number of datasets that are collected from other domains or artificially generated to learn the acquisition function (i.e., the predictor of the reduction in loss). The acquisition function is modeled and learned within the framework of a DQN. The state and action in our formulation have the following design:

State: Parameters that represent the predictive model and that describe the training data (c.f. coefficients for regressor, averaged distance to other data in the pool).

Action: A parameter that determines which data to select (c.f. uncertainty of prediction evaluated by the current predictive model).

By designing the state and action in this way, we can learn a Q-function that predicts the amount of test loss reduction (reward) by selecting an unlabeled data (action) given the current predictive model and pool data (state). Once the Q-functions have been obtained, it is possible to select the data that reduces the test loss the most when a certain number of data are added to the training data.

In this work, we consider supervised learning problem of predicting response variable $y \in \{1, 2, \ldots, C\}$ with d-dimensional explanatory variable $\boldsymbol{x} \in \mathbb{R}^d$. A dataset $D = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\}$ is assumed to be given for training a predictive model.

4.1 Design of the State, Action and Reward

For implementing DQN, we have to design state and action as input and output of DQN. In this subsection, we first define the state, namely, the feature vector extracted from the predictive model and annotated dataset. Then the action of the learner is defined, which corresponds to the data selection policy in terms of the acquisition function for active learning. Finally, the reward for a certain action is defined as the amount of increase of accuracy by adding the annotated datum selected by the learner. We note that there are various possibilities for

designing state, action, and reward, and we do not rule out other designs not adopted here. Appropriateness of those designs would depend on the dataset, predictive model, budget, and many other factors, and optimization of the design is an open issue.

4.1.1 Design of the State

The state used for the DQN should consist of parameters that reflect the performance and structure of the current learning model. In [28], it is empirically shown that simple features, such as the variance of the classifier output or the predicted probability distribution over possible labels for a specific datapoint on synthetic data, is effective for training the acquisition function. Here, we explain the parameters adopted for defining the state in our framework.

An arbitrary predictive model is used in the proposed framework, but we here consider a random forest [6]. We adopt the OOB accuracy A_o as one of the features of the acquisition function because it should be useful for expressing the performance of the predictive model.

The random forest used as the predictive model performs random sampling with replacement of the given dataset k times. Let $\Phi_i,\ i = 1, \ldots, k$ be a subset of the given dataset to construct the i-th decision tree for ensemble learning. In the sampling with replacement, approximately 36% of all data are not sampled[1]. The unsampled data subset is called the out-of-bag (OOB) sample, and is used to assess the generalization performance of the predictive model [6]. The OOB accuracy is obtained by performing verification using this OOB sample as follows:

$$A_o = \frac{1}{|O|} \sum_{(\boldsymbol{x}, y) \in O} 1_y(\mathrm{RF}(\boldsymbol{x}; D)), \tag{4}$$

where $(\boldsymbol{x}_i, y_i) \in O = D \setminus \cup_{i=1}^{k} \Phi_i$, and $\mathrm{RF}(\boldsymbol{x}; D)$ is the output when the input $\boldsymbol{x}$ is given to the random forest trained over the dataset D. $1_{y_i}(S)$ is an indicator function, and returns 1 when the statement $y_i = S$ holds, and returns 0 otherwise. We adopted the OOB accuracy A_o because it directly express the performance of the predictive model.

We use decision trees for weak learners in random forest. Decision tree divides the feature space into regions and determines the output depending on which region the input feature belongs to. The average of the number of divided regions and the number of divisions in the decision tree is given by

$$\nu = \frac{1}{k} \sum_{i=1}^{k} (N_i^T + N_i^S), \tag{5}$$

where N_i^T is the number of terminal regions in the i-th tree, and N_i^S is the number of splits in the i-th tree among the k decision trees. The average of the number of regions that were above and below the threshold when splitting just before the end region of the decision tree is expressed by Eqs. (6) and (8), respectively.

[1] Consider the probability that certain sample x_i out of size n dataset is not sampled in the sampling procedure of the random forest. The probability that this sample is not selected in a single sampling procedure is $(n-1)/n$. Since random forest performs sampling with replacement n times from the size n dataset, the probability that the certain sample x_i is not selected is calculated as $\left(\frac{n-1}{n}\right)^n = \left(1 - \frac{1}{n}\right)^n \xrightarrow{n \to \infty} e^{-1} \simeq 0.36$.

$$\nu_r = \frac{1}{k}\sum_{i=1}^{k} |R_i|, \tag{6}$$

$$R_i = \bigcup_{e=1}^{N^i} \left\{ z \in \Gamma_e^i \mid z \geq \tau_e^i \right\}, \tag{7}$$

and

$$\nu_l = \frac{1}{k}\sum_{i=1}^{k} |L_i|, \tag{8}$$

$$L_i = \bigcup_{e=1}^{N^i} \left\{ z \in \Gamma_e^i \mid z < \tau_e^i \right\}. \tag{9}$$

Here Γ_e^i is the e-th region immediately before the end region in the decision tree $h(\cdot; \Phi_i)$ constructed using the i-th sample Φ_i, and where τ_e^i means the threshold for splitting in the e-th region of a decision tree. N^i indicates the number of regions immediately preceding the end region in the decision tree $h(\cdot; \Phi_i)$. These values are considered to be useful for characterizing the intrinsic data structure, and used as features for the acquisition function.

We also use the first m contribution ratios of the eigenvalues of the $d \times d$ matrix $X^\top X \in \mathbb{R}^{d\times d}$, where rows of the matrix X are annotated data $\boldsymbol{x}_i \in \mathbb{R}^d$ at the current stage of active learning. We note that $m \leq d$, the dimension of explanatory variable. The intuition for the use of eigenvalues and associated contribution ratios of the design matrix is that in statistical learning theory [45], eigenvalue of the data matrix plays an essential role for characterising the learnability. The contribution ratio of the j-th principal component is given by $\xi_j = \frac{\lambda_j}{\sum_{i=1}^{n}\lambda_i}$, where λ_i is the i-th eigenvalue, and n is the dimension of feature vector $\boldsymbol{x}$.

Finally, state vector $\boldsymbol{s}$ is defined by concatenating the above features as follows:

$$\boldsymbol{s} = [A_o, \nu, \nu_r, \nu_l, \xi_1, \ldots, \xi_m] \in \mathbb{R}^{4+m}. \tag{10}$$

4.1.2 Design of the Action

In the proposed method, the data selection corresponds to the action in the reinforcement-learning procedure. In this study, the action is designed using the indices used in existing active learning methods. We combine uncertainty sampling (US) [30] and the variance for prediction [1] to improve the performance. Combining other indices used for active learning, e.g., disagree probability [39], margin of the classification surface [43], would be possible, with a possible increase of computational cost. In the proposed method, the action value is determined so that the reward is maximized for a certain state. Then, from the pool dataset, we select the datum closest to that optimal action and assign a label to it.

The posterior probability of class discrimination is an index used in US [30], a commonly used active learning method, and it is defined by

$$x^* = \mathrm{argmin}_{\boldsymbol{x}\in\Pi} \mathrm{u}(\boldsymbol{x}), \quad \mathrm{u}(\boldsymbol{x}) = \max_c P(Y = c|\boldsymbol{x}). \tag{11}$$

Here, $P(Y = c|\boldsymbol{x})$ is the probability that the class of the response variable Y is c given $\boldsymbol{x} \in \mathbb{R}^d$, and Π is the pooled dataset. We note that we can consider solving both regression and classification problems by our method, but acquisition function is trained with classification

dataset. It is not absolutely necessary, but using classification dataset make the design of action easier.

Variance for prediction is another index widely used in active learning methods such as query by committee (QBC) [1], and is defined as $\mathrm{var}(\boldsymbol{x}) = \frac{1}{k}\sum_{i=1}^{k}(h(\boldsymbol{x};\Phi_i) - \mu(\boldsymbol{x}))^2$, $\mu(\boldsymbol{x}) = \frac{1}{k}\sum_{i=1}^{k} h(\boldsymbol{x};\Phi_i)$, where $h(\boldsymbol{x};\Phi_i)$ is a tree within the random forest and Φ_i is the subset used for training the tree.

Using the pre-trained DQN, the proposed method outputs the discriminant posterior probability and variance as a feature vector to characterize an ideal action. The action vectors are calculated for each datum in the pool, and the closest one to the ideal action is selected and labeled, which is expressed as follows:

$$\boldsymbol{x}^* = \mathrm{argmin}_{\boldsymbol{x}\in\Pi}\|\boldsymbol{a} - [\mathrm{u}(\boldsymbol{x}), \mathrm{var}(\boldsymbol{x})]^T\|_2, \boldsymbol{a} = [\mathrm{u}', \mathrm{var}']. \tag{12}$$

Here, u' and var' are the values of the discriminant posterior probability and variance, respectively, determined to be optimal by the DQN.

4.1.3 Design of Reward

We define the immediate reward as the amount of increase in accuracy obtained by adding a new training sample $(\boldsymbol{x}, y)$ to the predictive model. That is,

$$r_t = acc(D_t; D \cup \{(\boldsymbol{x}, y)\}) - acc(D_t; D), \tag{13}$$

where $acc(D_t; D)$ is the accuracy of the prediction of the model trained using dataset D and evaluated using dataset D_t.

4.2 Advantage of the Proposed Method

Because the reward in Q-learning is the cumulative sum of the immediate reward at each decision, the learned acquisition function takes into account the context, i.e., the situation of data acquisition under the condition that the maximum number of obtainable training data is fixed. Additionally, the output behavior (which is treated as optimal) can be used in combination with the criteria of any existing method; hence, the design of the state in the proposed method is highly flexible, unlike that of the method proposed in [33], which uses heuristics specific to certain tasks and predictive models. Although the use of data from other domains was inspired by [28], the proposed method updates DQN parameters by reinforcement learning.

The conceptual diagram and pseudo-code of the proposed method are shown in Fig. 1 and Algorithm 1, respectively.

5 Experiments

This section describes the evaluation of the proposed active learning method via a set of multi-class classification experiments with both artificial and real-world datasets[2]. We first investigate which dataset is useful for learning the acquisition function. Then, the proposed method is compared with existing methods over six datasets obtained from the UCI Machine

[2] Source code implemented by the first author to reproduce the experimental results is available from https://github.com/dxa0010/FBCA.

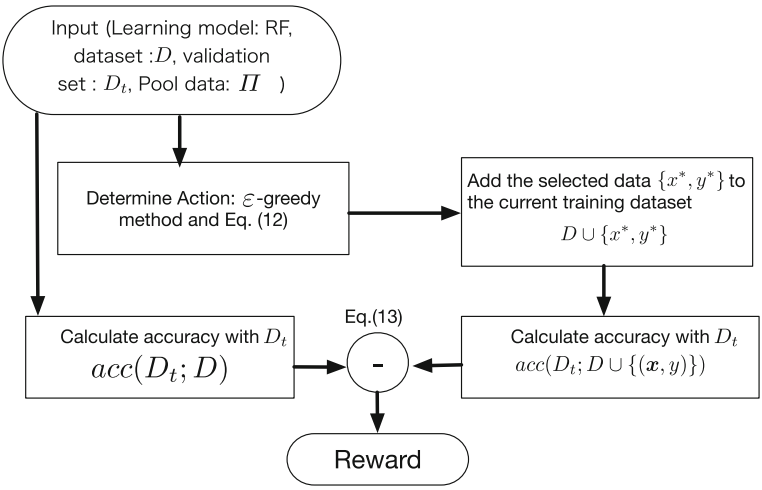


Fig. 1 One epoch in reinforcement learning for learning an acquisition function

Algorithm 1 Training DQN

Input: Dataset D, number of epochs (iteration for learning DQN) N_e, initial random parameter for DQN $\boldsymbol{\theta}$, budget T.

for $j = 1$ to N_e **do**
  Split D to training dataset and pool dataset
  for $t = 1$ to T **do**
    Train RF model using the training dataset, and calculate state $\boldsymbol{s}$ by Eq. (10)
    Calculate action value from $Q(\boldsymbol{s}, \boldsymbol{a}; \boldsymbol{\theta})$ and select data from the pool according to Eq. (12).
    Calculate immediate reward by Eq. (13)
  end for
  Train DQN. Find $\boldsymbol{\theta}_j$ that minimizes Eq. (3), and update the parameter: $\boldsymbol{\theta} = \boldsymbol{\theta}_j$
end for

Output: $\boldsymbol{\theta}$

Learning Repository. Finally, through a simple experiment, we demonstrate that the proposed method is able to select appropriate samples considering the context of the learning process. Throughout the experiments, the architecture of DQN is fixed as follows:

- input layer: 10 dimension for the above defined state features.
- three times repetition of 16 dimensional fully connected layer followed by Relu activation.
- output layer: 2 dimension for the above defined action features.

The architecture is relatively small and the performance could be improved by neural architecture search [11], which is left for our future work. The network is trained using Adam [24] with learning rate 0.001 and the cost function is the mean absolute error.

To define the state vector $\boldsymbol{s}$ in Eq. (10), the parameter m should be determined. Throughout the experiments, we set $m = 6$, which is the smallest number of dimensions of all of the datasets used in the experiments. In our preliminary experiments, we saw that larger m tends to offer better classification performance, but the difference were not significant.

Table 1 Number of classes, feature dimensions, and samples in the datasets used for training the acquisition function

Name	# of classes	# of features	# of samples
Dataset A	2	6	10,000
Dataset B	10	8	10,000
Dataset 1	5	7	10,000
Dataset 2	10	8	10,000
Dataset 3	15	10	10,000
Dataset 4	25	17	10,000

5.1 Dependence on the pre-training dataset

In the proposed method, as in [28], the data selection criterion (the acquisition function) is learned beforehand using datasets from other domains. In this study, we created six datasets for classification problems following the procedure proposed in [16]. Details of the generated artificial datasets are reported in Table 1.

Of these datasets, datasets A and B were used for training the DQN, and the remaining four datasets were used for verification. For all datasets, in the active learning experiment, 1,000 data were used as verification data, and the remaining 9,000 data were divided into training data and pool data. During learning, the number of training data for DQN was randomly varied to enable it to handle various situations. The budget for active learning was set to 100 samples, and the training of the DQN was completed when 100 samples were taken. The result of experimenting under these conditions is shown in Fig. 2.

Of all four types of datasets, the proposed method trained with dataset A yielded superior performance. This experiment shows that there is a large difference in performance depending on the learning source dataset. In the following experiment, the model trained with dataset A was used.

5.2 Experiments with Real-World datasets

In this section, we compare the performance of the proposed method with those of existing methods over six datasets from the UCI Machine Learning Repository: adult [27], car [5], winequality-red, white [8], googletrip-review, and tripadvisor-review [34]. Profile of the used datasets are summarized in Table 2. These datasets are popular benchmark datasets in machine learning and active learning, range from classical (adult, car, winequality) to relatively new (googetrip-review and tripadvisor-review), various input dimensionalities (6–23), includes both qualitative and quantitative features. In particular, the latter four datasets are considered as suitable for active learning where annotation is hard, namely, evaluating wine quality requires tasting by expert sommeliers, and giving a review to a tourist spot require visiting the place actually.

US, QBC, LAL, and random sampling were used as comparison methods. A similar method [33] relies on a two-class support vector machine as a learning model, so it is not suitable for an experiment with a multi-class discriminant. As stated in Sect. 5.1, the number of data to be acquired (the budget) from the pool data is 100. The result of the experiment using this setting is shown in Fig. 3.

The proposed method performed significantly better than the other methods on the googletrip-review and tripadvisor-review datasets, and it performed comparably to the other

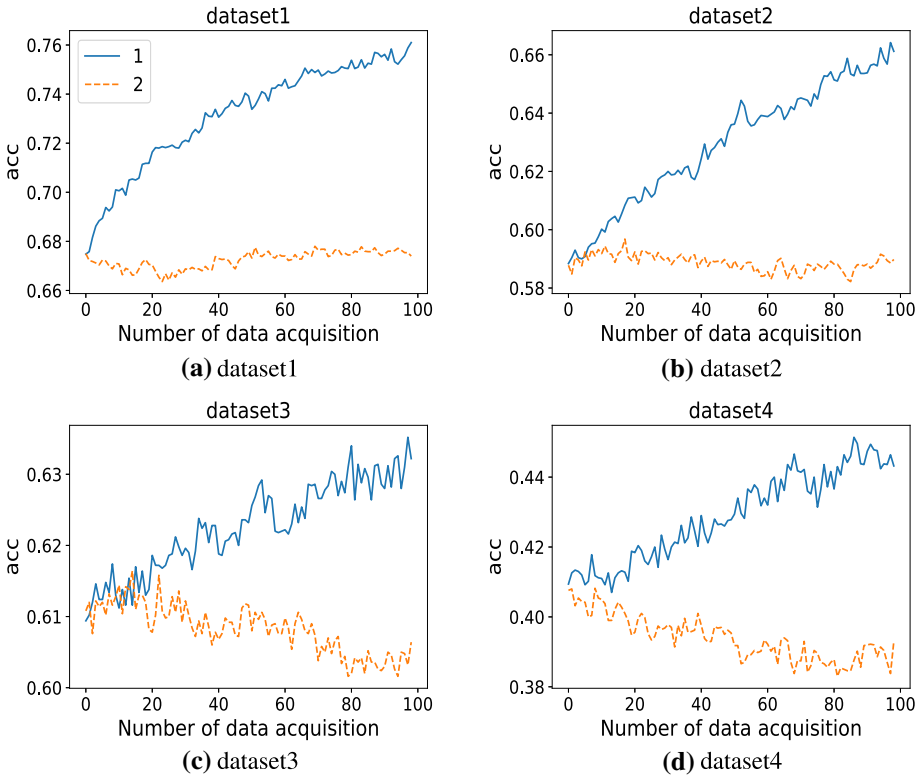
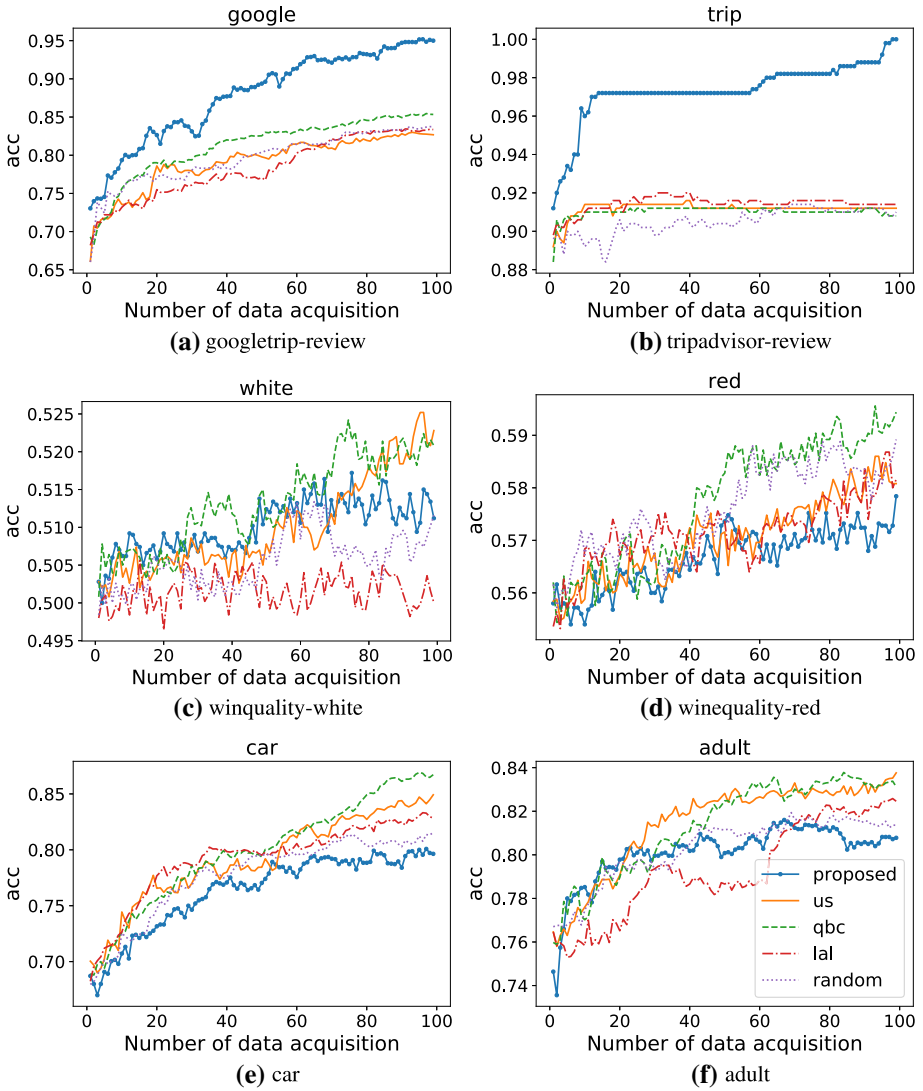


Fig. 2 Difference in performance depending on the dataset for learning the acquisition function. Of the 6 types of datasets shown in the Table 1, "1" is the one learned with datasetA, and "2" is the one learned with datasetB (line types are described in (**a**)). The vertical axis is the correct answer rate and the horizontal axis is the number of data to be acquired. Each plot is the averaged values in 5 times

Table 2 Profile of the datasets used for evaluation

Dataset	Dimension	# of initial samples	# of test samples	# of pools	Attribute type
Googletrip	23	10	1000	4446	Quantitative
Tripadvisor	10	10	1000	31551	Quantitative
Wine white	11	200	1000	3698	Mixed
Wine red	11	100	500	999	Mixed
Car	6	10	500	1218	Quantitative
Adult	14	10	1000	47742	Mixed

methods for wine-white and adult datasets. Among six datasets, our proposed method does not perform well compared to other methods for wine-red and car datasets. From Table 2, these two datasets have relatively smaller pool datasets, and it is possible that our proposed method requires larger pool datasets than other methods to ensure that the actual and pre-trained datasets have large enough intersection. The difference in performance between datasets could be partly due to the similarity between the dataset used for pre-training and the dataset used for active learning. We also conjecture the similarity of feature distribution to



(a) googletrip-review **(b)** tripadvisor-review **(c)** winquality-white **(d)** winequality-red **(e)** car **(f)** adult

Fig. 3 Comparison of active learning methods on six real-world datasets. The vertical axis shows the correct answer rate and the horizontal axis shows the number of data acquired. Results for **a** googletrip-review, **b** tripadvisor-review, **c** winequality-white, **d** winequality-red, **e** car, and **f** adult. Each plot is the average result of five-fold cross-validation

the datasets for pre-training is the most important factor to the performance of the proposed method. Investigation of the feature similarity and selection of the best dataset for learning acquisition functions is our important future work.

5.3 Evaluation of the Context Awareness

In this subsection, we compare the active learning methods with the oracle data selector to demonstrate that the proposed method considers the context of data selection. Here, the oracle

Table 3 Performance comparison of the oracle and proposed method after acquiring five data

Method	Accuracy mean±std	Index match rate
Oracle	93.59 ± 2.792	–
Proposed	93.59 ± 2.792	32.0%
Random	91.60 ± 2.939	16.0%

is a method that selects the most appropriate set of data. To make the combinatorial calculation feasible, the size of the pool dataset is restricted to 25 and the number of acquisitions (the budget) is set to five. For this setting, the acquisition function was trained using dataset A, and the active learner was tested on the tripadvisor-review dataset. In this experiment, we compare the classification accuracy of the final models and the matching rate of the selected subset of data. Table 3 shows the results of five-fold cross-validation.

When comparing the proposed method with the oracle, the averages of the five trials are exactly the same. The matching rate of the data selected by the proposed method is 32%, which is higher than that of random sampling. This indicates that the probability of obtaining a combination close to that of the oracle is increased by considering the context. The five data points actually selected are different to those obtained by the oracle because the data were acquired so that the performance is maximized over the combination of all five. Although the number of pool data was very low (25), the number of combinations of data acquisition ($_{25}C_5 = 53,130$) is sufficiently large. When acquiring data at random, the probability that all five selected data would match that of the oracle is 0.000019%. Hence, the results obtained by the proposed method are much better than the expected value of those obtained at random.

5.4 Computational Costs

Active learning is a methodology required in situations where measurement and experiments are costly, and it is unlikely that the calculation cost of the acquisition function will become a problem. For reference, Fig. 4 shows the time required to evaluate the acquisition function for each method used in our comparative experiment. Since computational time is affected by various factors such as the dimensionality of data, size of pooled dataset and distribution of pool or population dataset, we consider the relative computational times to those of random sampling, which is of the order of milliseconds[3]. We note that for our LAL and the proposed method, we have to train acquisition functions in advance. The computational cost for training acquisition function for LAL is around one hour, and that for DQN ($N_e = 5000$ epochs) in our method is around 20 hours. The acquisition functions can be trained in advance and the computational cost for training the acquisition function does not affect the running time for active learning. Also, the computational time would be reduced by parallel computation.

From Fig. 4, we see that the uncertainty sampling method is consistently faster than other methods. For the other three methods, the computational time is comparable.

6 Conclusion and Future Work

We proposed an active learning method suitable for a fixed budget regime. The proposed method considers the context of data acquisition using a random forest as a learning model

[3] We used Intel(R) core i7-4712MQ CPU 2.30GHz with 8GB RAM.

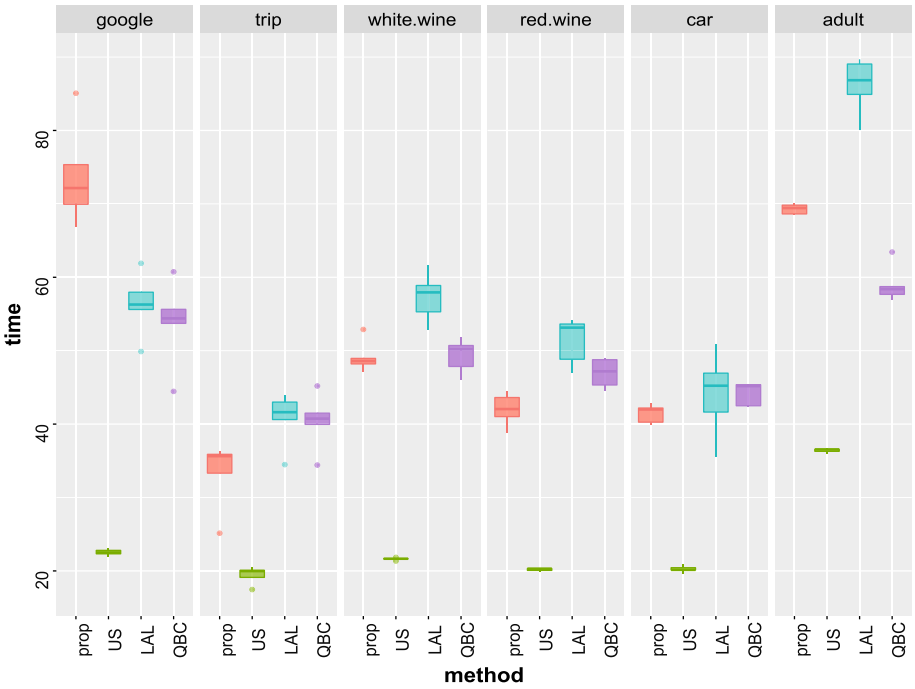


Fig. 4 Relative computational time to random sampling

and a reinforcement-learning DQN. When the data used for reinforcement learning in advance matches the data at the time of operation, a significant improvement in accuracy was seen when compared with the results of the existing method.

In the proposed method, learning is performed by extracting features from the target learning model and dataset. However, because this part relies on human-generated heuristics, it may affect performance depending on how the features are selected. In the future, we will make this part automatically learnable. In addition, we used a single dataset for training the acquisition function, but the generalization would be improved by mixing datasets from different domains. For that purpose, it will be necessary to investigate the difference in performance depending on the dataset used for learning the acquisition function.

As an important future work, theoretical underpinning of the proposed approach remains to be investigated. For example, sample complexity of active learning is established when the disagreement-based acquisition functions is employed [17,21,51]. However, the approach of learning acquisition function is recently proposed and its learning theory is yet to be developed. We expect that to develop a learning theory for the proposed method, combination of the analysis of online learning (which requires treatment as stochastic processes and martingale analysis) and a regret analysis for assessing the quality of the acquisition function trained with reinforcement learning would be necessary, making the problem difficult. Another important direction of future research is the strategy of multiple-selection or batch selection from pooled dataset. When we consider deep neural network as a predictive model, adding only one sample per iteration of active learning is not reasonable because of high cost for training the model. Also, it is unlikely that performance of a CNN changes with only one additional training datum. Several selection methods of multiple samples at a time in active learning is

recently studies [37,48], and incorporating the notion of *context* to multiple sample selection method would be of great practical importance.

Acknowledgements We express special thanks to the anonymous reviewers whose comments led to valuable improvements of this paper.

References

1. Abe N, Mamitsuka H (1998) Query learning strategies using boosting and bagging. In: ICML
2. Altschuler M, Bloodgood M (2019) Stopping Active Learning Based on Predicted Change of F Measure for Text Classification. In: Proceedings - 13th IEEE International Conference on Semantic Computing, ICSC 2019, pp. 47–54. https://doi.org/10.1109/ICOSC.2019.8665646
3. Bachman P, Sordoni A, Trischler A (2017) Learning algorithms for active learning. In: ICML
4. Bloodgood M, Vijay-Shanker K (2009) A method for stopping active learning based on stabilizing predictions and the need for user-adjustable stopping. In: Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009), pp. 39–47. Association for Computational Linguistics, Boulder, Colorado. https://www.aclweb.org/anthology/W09-1107
5. Bohanec M, Rajkovic V (1988) Knowledge acquisition and explanation for multi-attribute decision making. In: 8th Intl Workshop on Expert Systems and their Applications, pp. 59–78
6. Breiman L (2001) Random forests. Mach Learn 45(1):5–32
7. Chen Y, Luo H Ma T, Zhang C (2020) Active Online Domain Adaptation. Tech. rep. arXiv:2006.14481
8. Cortez P, Cerdeira A, Almeida F, Matos T, Reis J (2009) Modeling wine preferences by data mining from physicochemical properties. Decis Support Syst 47(4):547–553
9. Das S, Wong W, Dietterich TG, Fern A, Emmott A (2016) Incorporating expert feedback into active anomaly discovery. In: IEEE 16th International Conference on Data Mining, ICDM 2016, December 12-15, 2016, Barcelona, Spain, pp. 853–858. https://doi.org/10.1109/ICDM.2016.0102
10. Ebert S, Fritz M, Schiele B (2012) Ralf: A reinforced active learning formulation for object class recognition. In: CVPR. https://doi.org/10.1109/CVPR.2012.6248108
11. Elsken T, Metzen JH, Hutter F (2019) Neural architecture search: a survey. Journal of Machine Learning Research, 20:1–21 (http://jmlr.org/papers/v20/18-598.html)
12. Fang M, Li Y, Cohn T (2017) Learning how to active learn: A deep reinforcement learning approach. In: EMNLP 2017 - Conference on Empirical Methods in Natural Language Processing, Proceedings, pp. 595–605 . https://doi.org/10.18653/v1/d17-1063. https://github.com/
13. Fujii K, Kashima H (2016) Budgeted stream-based active learning via adaptive submodular maximization. In: Advances in Neural Information Processing Systems, pp. 514–522
14. Gal Y, Islam R, Ghahramani Z (2017) Deep Bayesian active learning with image data. In: 34th International Conference on Machine Learning, ICML 2017, vol. 3, pp. 1923–1932
15. Goodfellow I, Bengio Y, Courville A (2016) Deep Learning. MIT Press, Cambridge
16. Guyon I, Li J, Mader T, Pletscher PA, Schneider G, Uhr M (2007) Competitive baseline methods set new standards for the nips 2003 feature selection benchmark. Pattern Recogn Lett 28(12):1438–1444
17. Hanneke S, Yang L (2015) Minimax Analysis of Active Learning. J Mach Learn Res 16:3487–3602
18. Haußmann M, Hamprecht F, Kandemir M (2019) Deep active learning with adaptive acquisition. IJCAI International Joint Conference on Artificial Intelligence 2019–Augus:2470–2476. https://doi.org/10.24963/ijcai.2019/343
19. Hino H (2020) Active learning: Problem settings and recent developments. arXiv:2012.04225
20. Houlsby N, Hernández-Lobato JM, Huszár F, Ghahramani Z (2012) Collaborative Gaussian processes for preference learning. Adv Neural Inf Process Syst 3:2096–2104
21. Huang TK, Agarwal A, Hsu DJ, Langford J, Schapire RE (2015) Efficient and parsimonious agnostic active learning. In: C. Cortes, N.D. Lawrence, D.D. Lee, M. Sugiyama, R. Garnett (eds.) Advances in

Neural Information Processing Systems 28, pp. 2755–2763. Curran Associates, Inc. http://papers.nips.cc/paper/5939-efficient-and-parsimonious-agnostic-active-learning.pdf
22. Ishibashi H, Hino H (2020) Stopping criterion for active learning based on deterministic generalization bounds. In: The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy], pp. 386–397
23. Karzand M, Nowak RD (2020) MaxiMin active learning in overparameterized model classes. IEEE J Selected Areas Inf Theory 1(1):167–177. https://doi.org/10.1109/jsait.2020.2991518
24. Kingma DP, Ba JL (2015) Adam: A method for stochastic optimization. In: 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings
25. Kirsch A, van Amersfoort J, Gal Y (2019) Batchbald: efficient and diverse batch acquisition for deep bayesian active learning. In: Advances in Neural Information Processing Systems 32, pp. 7026–7037. Curran Associates, Inc. http://papers.nips.cc/paper/8925-batchbald-efficient-and-diverse-batch-acquisition-for-deep-bayesian-active-learning.pdf
26. Kobayashi T, Sugiyama M (2012) Early stopping heuristics in pool-based incremental active learning for least-squares probabilistic classifier. IEICE Transactions on Information and Systems E95.D(8):2065–2073. https://doi.org/10.1587/transinf.E95.D.2065
27. Kohavi R (1996) Scaling up the accuracy of naive-bayes classifiers: a decision-tree hybrid. In: KDD
28. Konyushkova K, Sznitman R, Fua P (2017) Learning active learning from data. In: NIPS
29. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: NIPS
30. Lewis DD, Gale WA (1994) A sequential algorithm for training text classifiers. In: SIGIR
31. Lookman T, Balachandran PV, Xue D, Yuan R (2019) Active learning in materials science with emphasis on adaptive sampling using uncertainties for targeted design. npj Computational Materials **5**(1). https://doi.org/10.1038/s41524-019-0153-8
32. Mnih V, Kavukcuoglu K, Silver D, Graves A, Antonoglou I, Wierstra D, Riedmiller MA (2013) Playing atari with deep reinforcement learning. CoRR
33. Pang K, Dong M, Wu Y, Hospedales TM (2018) Meta-learning transferable active learning policies by deep reinforcement learning. CoRR arXiv:1806.04798
34. Renjith S, Sreekumar A, Jathavedan M (2018) Evaluation of partitioning clustering algorithms for processing social media data in tourism domain. RAICS
35. Ricci F, Rokach L, Shapira B (2015) Active learning in recommender systems, pp. 809–846. Springer US, Boston, MA. https://doi.org/10.1007/978-1-4899-7637-6_24
36. del Rosario Z, Rupp M, Kim Y, Antono E, Ling J (2020) Assessing the frontier: active learning, model accuracy, and multi-objective candidate discovery and optimization. J Chem Phys 153(2), http://aip.scitation.org/doi/10.1063/5.0006124
37. Sener O, Savarese S (2018) Active learning for convolutional neural networks: A core-set approach. In: 6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings
38. Settles B (2009) Active learning literature survey. Computer Sciences Technical Report 1648. University of Wisconsin, Madison 52:55–66
39. Seung HS, Opper M, Sompolinsky H (1992) Query by committee. In: Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory, pp. 287–294. https://doi.org/10.1145/130385.130417
40. Stark F, Triebel R, Cremers D (2015) CAPTCHA Recognition with Active Deep Learning. In: 37th German Conference on Pattern Recognition, p. 94
41. Sutton RS, Barto AG (2018) Reinforcement learning: an introduction, 2nd edn. The MIT Press, Cambridge
42. Tian Y, Yuan R, Xue D, Zhou Y, Ding X, Sun J, Lookman T (2020) Role of uncertainty estimation in accelerating materials development via active learning. J Appl Phys 128(1), http://aip.scitation.org/doi/10.1063/5.0012405
43. Tong S, Koller D (2002) Support vector machine active learning with applications to text classification. J Mach Learn Res 2(1):45–66. https://doi.org/10.1162/153244302760185243
44. Ueno T, Hino H, Hashimoto A, Takeichi Y, Sawada M, Ono K (2018) Adaptive design of an x-ray magnetic circular dichroism spectroscopy experiment with gaussian process modelling. npj Computational Materials 4(1):4. https://doi.org/10.1038/s41524-017-0057-4
45. Vapnik VN (1995) The nature of statistical learning theory. Springer-Verlag, New York Inc
46. Vladimir V (1999) An overview of statistical learning theory. IEEE TNN 10(5):988–999
47. Wang K, Zhang D, Li Y, Zhang R, Lin L (2017) Cost-Effective Active Learning for Deep Image Classification. IEEE Trans Circuits Syst Video Technoly 27(12):1051–8215
48. Wang Z, Ye J (2015) Querying discriminative and representative samples for batch mode active learning. ACM Trans Knowl Discov Data 9(3):17

49. Wassermann S, Cuvelier T, Casas P (2019) RAL-Improving stream-based active learning by reinforcement learning. In: European Conference on Machine Learning and Principles and Practiceof Knowledge Discovery in Database, Workshop on Iterative Adaptive Learning, vol. 2444, pp. 32–47 (2019). https://hal.archives-ouvertes.fr/hal-02265426
50. Woodward M, Finn C (2017) Active one-shot learning. CoRR arXiv:1702.06559
51. Yan S, Chaudhuri K, Javidi T (2018) Active learning with logged data. In: J. Dy, A. Krause (eds.) Proceedings of the 35th International Conference on Machine Learning, *Proceedings of Machine Learning Research*, vol. 80, pp. 5521–5530. PMLR, Stockholmsmässan, Stockholm Sweden. http://proceedings.mlr.press/v80/yan18a.html