Machine Learning Approaches for Interactive Verification

Yu-Cheng Chou and Hsuan-Tien Lin

Department of Computer Science, National Taiwan University, Taipei 106, Taiwan

Abstract. Interactive verification is a new problem, which is closely related to active learning, but aims to query as many positive instances as possible within some limited query budget. We point out the similarity between interactive verification and another machine learning problem called contextual bandit. The similarity allows us to design interactive verification approaches from existing contextual bandit approaches. We compare the performance of those approaches on interactive verification. In particular, we propose to adopt the upper confidence bound (UCB) algorithm, which has been widely used for the contextual bandit, to solve the interactive verification problem. Experiment results demonstrate that UCB reaches superior performance for interactive verification on many real-world datasets.

Keywords: active learning, contextual bandit, upper confidence bound

1 Introduction

Breast cancer is the most frequently diagnosed cancer in woman (Rangayyan et al., 2007). Breast cancer screening is a strategy to achieve an earlier diagnosis in asymptomatic women for breast cancer. A common technique for screening is mammography. Somehow interpreting mammogram images is difficult and requires radiology experts, while hiring radiology experts is usually expensive. In breast cancer screening, most of the efforts are spent on interpreting mammogram images from healthy individuals. But actually only the mammogram images from the patients with breast cancer require the diagnosis from radiology experts. If we can select a subset of patients that are asymptomatic, we can save radiology experts a lot of efforts. One possible way to do so is to let computers select the subset automatically in a computer-aided diagnosis (CAD) system.

CAD systems are designed to assist radiology experts in interpreting mammogram images (Rangayyan et al., 2007; Li and Zhou, 2007). A CAD system can prompt potential unhealthy region of interests (ROIs) for radiology experts to verify. A typical CAD session can be decomposed into three stages: labeling stage, where radiology experts perform the reading of some mammogram images and record the label (malignant or benign) for each ROI; learning stage, where a learning algorithm within the CAD system builds a classifier to predict the labels of ROIs for future mammogram images based on the labels obtained from labeling stage; verification stage, where radiology experts analyze the prompts given by the CAD system to verify whether the ROIs are malignant or

benign. A CAD system can reduce the efforts spent in breast cancer screening by selecting worthy-verified ROIs for radiology experts. Such a problem, which allows human experts to verify something (malignant ROIs) selected by computers (CAD system), is named the "verification problem" in this work.

In a verification problem, there are two stages that require the efforts of human experts: the labeling stage and the verification stage. These two stages are different from the point of view of the system. In the labeling stage, the system requests label of an ROI for learning; in the verification stage, the system prompts an ROI that is considered to be positive (malignant) for verification. Nevertheless, these two stages are similar from human experts' point of view. Both of them require radiology experts to diagnose on an ROI and return the diagnosis. We call the request of diagnosis as a "query" in the verification problem. Given the similarity between the labeling stage and verification stage, we propose to combine these two stages together: a human expert can do the verification while doing the labeling; and the feedback of the verification can be treat as the labeling result. By combining the learning and verification, the system can get the flexibility to decide how to distribute limited human resources on these two stages to achieve better performance. Given limited query budget, how could we most efficiently distribute and utilize the queries to verify as many malignant ROIs as possible? This is the main question of this work.

In this paper, we formalize the question above by defining a new problem called interactive verification. The problem describes a procedure that performs verification through the interaction between the system and the human experts. By interacting with humans, the system aims to verify as many positive instances as possible within limited query budget, and the query result can be immediately used to learn a better classifier. An effective approach for the problem can then help reduce the overall human efforts.

In our work, we first point out the similarity of interactive verification to the popular contextual bandit problem (Langford and Zhang, 2007). We also discuss the similarity of interactive verification to the active learning problem. Then, we design four possible interactive verification approaches based on the similarities. In particular, one of the four is called the upper confidence bound (UCB), which is adopted from a state-of-the-art family of contextual bandit algorithms. We conduct experiments on real world datasets to study the performance of these approaches. The results demonstrate that UCB leads to superior performance.

The rest of this paper is organized as follows. In Section 2, we define the interactive verification problem and compare it to other problems. We describe our design of the four approaches to solve the problem in Section 3. Finally, we present the experiment results in Section 4 and conclude our work in Section 5.

2 Problem Setting

Given a set of instances $X = \{x_1, ..., x_m\}$, where each instance x_i is associated with a label $Y(x_i) \in \{-1, 1\}$. We define the set of positive instances $P = \{x_i \in X | Y(x_i) = 1\}$, which is the set of the instances that require verification. Interactive verification is an iterative process. In the first iteration, we assume that an interactive verification learner knows the labels of one positive instance and one negative instance as initial instances

and do not know the labels of other instances. On the t-th iteration, the learner is asked to select an instance s_t from unlabeled (un-verified) dataset U, where $U = \{x_i \in X | x_i \neq s_\tau, \forall \tau < t\}$. The learner then receives the label $Y(s_t)$ to update its internal model. The goal is to verify as many positive instances as possible within T iterations. That is, we want to maximize

$$\sum_{t=1}^{T} \frac{[Y(s_t) = 1]}{|P|}. \tag{1}$$

Sabato et al. (2013) also proposed an equivalent problem called "auditing", which aims to minimize the number of labeled negative instances needed to classify all of the instances accurately. The work compares the similarity and differences between auditing and active learning, and only studies one baseline auditing algorithm. In this work, we consider designing and comparing different approaches for the interactive verification problem.

As pointed out by Sabato et al. (2013), immediate tools for interactive verification can be easily found in active learning. Active learning is a form of supervised learning in which the learner can interactively ask for information (Settles, 2009). The spirit of active learning is to believe that the information amount carried by each instance is different. By choosing informative instances to query, the learner can obtain an accurate model with only few labeled instances, thereby reducing human efforts.

Pool-based active learning is a widely used setting for active learning, which assumes that the learner can only query the instances chosen from a given dataset pool (Lewis and Gale, 1994). The setting of pool-based active learning is almost the same as interactive verification: both of them allow the learner to query an instance to obtain its label in each iteration. The difference between them is the different goals. Active learning focuses on getting an accurate model; on the other hand, interactive verification aims to maximize the number of verified positive instances. Although the goals are different, the similar setting allows tools of active learning to be possibly used for interactive verification.

In this work, we will connect interactive verification to another problem called contextual bandit. The contextual bandit problem is a form of multi-armed bandit problem, where a player faces some slot machines and wants to decide in which order to play them (Auer et al., 2000). In every iteration, the player can select one slot machine (action) from some action set A. Then, the player will receive a randomize reward decided by the distribution under the corresponding slot machine (action). The goal is to maximize the rewards received by the player after a given number of iterations. One key property of the multi-armed bandit problem is that we could only get partial information from environment: only the reward of the selected action will be revealed. If an action has never been selected, the player will not have information about it. Thus, it is necessary to spend some iterations to explore the actions that the player is not familiar with. Somehow only doing the exploration cannot maximize the total rewards, and the player also needs to spend some iterations to exploit the action with high expected rewards. The key to solve the multi-armed bandit problem is to find the balance between the exploration and the exploitation. In addition to the setting above, the contextual bandit problem allows the learner (player) to receive some context information about the

environment prior to making selections in every iteration (Langford and Zhang, 2007). The context information makes it possible for contextual bandit algorithms to exercise a more strategic choice according to the context.

In a first glance, the setting of contextual bandit appears very different from interactive verification. A closer look at the two problems, however, reveal that the trade-off between the exploration and the exploitation in contextual bandit is similar to the trade-off between the learning stage and the verification stage in the interactive verification. In particular, if we define a special contextual bandit problem as follows: The action set A consists of the choices to query each unlabeled instance; the context represent the features of each unlabeled instance; the reward is 1 if the selected action (queried instance) is a positive one, and 0 otherwise. Then, we see that maximizing the cumulative rewards in such a contextual bandit problem is exactly the same as maximizing (1). The connection leads to new possibilities in designing interactive verification approaches, which will be discussed in the next section.

Although we find the similarity between contextual bandit and interactive verification, there is still a big difference. In a contextual bandit problem, each action is usually allowed to be selected several times. Then, the actions that are more likely to produce high rewards could be selected more often. In interactive verification, however, each instance is supposed to be queried at most once. That is, in the corresponding contextual bandit problem, each action can be selected at most once. The difference make it non-trivial to apply existing contextual bandit algorithms for interactive verification.

3 Approaches

For the convenience of discussion, we first outline a general framework for interactive verification approaches. In every iteration, we use a base learner to train the model from labeled instances, and then the learner chooses the next instance to be queried according to a scoring function computed from the model. The general framework is shown in Algorithm 1. By defining the scoring function, we define the behavior of an approach to interactive verification.

Algorithm 1 General approach to interactive verification

Require: Base learner, B; Unlabeled instances, U; Labeled instances, L; Number of iterations, T;

for $t = 1$ to T **do**
  model $M = B(L)$
  for all $u \in U$ **do**
    Compute scoring function: $S(u, M)$
  end for
  $s_t = \arg\max_u S(u, M)$
  $L = L \cup \{(s_t, Y(s_t))\}$
  $U = U \backslash \{s_t\}$
end for

In this work, we use support vector machine (SVM) with linear kernel as our base learner, and denote w_t to be the liner weights we get from the base learner in the beginning of every iteration.

3.1 Greedy Approach

The goal of our problem is to verify as many positive instances as possible. The most intuitive solution is querying the instance which be considered most likely to be positive by current model in every iteration, i.e. the instance with highest $p(y = 1|x_i)$. When using SVM as base learner, the instance to be queried comes with the largest decision value. That is, the scoring function of the greedy approach is simply

$$S(x_i, w_t) = x_i^\intercal w_t.$$

Greedy approach only considers how possible an instance to be positive in each iteration. It ignores the information amount carried by each instance. If we start from a biased model, the greedy approach may perform poorly. Here, we give an example that the greedy approach will fail. Consider the case shown in Figure 1. There are two clusters of red positive instances and one big cluster of blue negative instances in the figure. Without loss of generality, we assume the initial positive instance is in the top red positive cluster. The model we start with will be the dashed line. The optimal model is the solid line, which is very different from the dashed line. By running greedy approach on this dataset, we can easily verify the positive instances in top cluster. But after all the instances in top positive cluster is queried, greedy approach will prefer to query the instances in the negative cluster than query the instance in bottom positive cluster. To solve this issue, we may need to do some explorations to help us find the instances in the bottom positive cluster.

3.2 Random then Greedy

In the previous subsection we discuss the risk of not doing exploration. Here we propose an approach using the random as exploration method to solve interactive verification problems: random then greedy (RTG). Randomly selecting an instance to query is a naive yet reasonable strategy to do the exploration. It can provide some unbiased information. Then, we use greedy approach described in he previous section for exploitation (verification). In this approach we do an one-time switching from exploration to exploitation. We use the parameter ϵ to decide the ratio between exploration and exploitation. That is, the scoring function of RTG is

$$S(x_i, w_t) = \begin{cases} random(), & \text{if } t \leq \epsilon T \\ x_i^\intercal w_t, & \text{otherwise} \end{cases}.$$

3.3 Uncertainty Sampling then Greedy

As the discussion in Section 2, the setting of the interactive verification is pretty similar to the active learning problem. It is natural to attempt to use tools of active learning

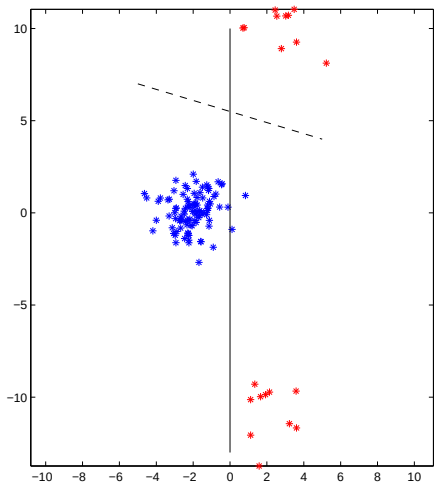


Fig. 1. Artificial dataset

for interactive verification. Uncertainty sampling is one of the most commonly used algorithm for active learning (Settles, 2009). The idea is to query the instances that the current model is least certain on how to label it. For probabilistic learning models, uncertainty sampling queries the instances with probability to be positive close to 50%. Uncertainty sampling can also be employed with non-probabilistic learning model. When using SVM as the base learning model, uncertainty sampling queries the instance closest to the linear decision boundary (Tong and Koller, 2001).

To apply the uncertainty sampling on the interactive verification, we can borrow the framework from RTG as described in previous section. We use greedy as exploitation method and use uncertainty sampling as our new exploration method to replace random sampling. We call this approach uncertainty sampling then greedy (USTG). The scoring function of USTG is

$$S(x_i, w_t) = \begin{cases} \frac{1}{|x_i^\intercal w_t|+1}, & \text{if } t \leq \epsilon T \\ x_i^\intercal w_t, & \text{otherwise} \end{cases}.$$

Uncertainty sampling may suffer from a biased model like the greedy approach. When starting with a model of bad quality, the instances that are selected by uncertainty sampling may not be very informative. Thus, using the uncertainty sampling as exploration method cannot totally solve the issue of biased model in the greedy approach.

3.4 Upper Confidence Bound

Upper confidence bound (UCB) is an algorithm to solve the multi-armed bandit problem (Auer et al., 2000). The idea of UCB is to keep the upper bound of plausible rewards

of the actions and select the action according this value. In the traditional multi-armed bandit problem, there is no contextual features. The prediction of confidence bound is based on how many times we select the action. In an interactive verification problem, each action can be only applied once, and hence the algorithm for multi-armed bandit problem cannot be applied to the interactive verification directly. But as our discussion in Section 2, we can transform an interactive verification problem to a contextual bandit problem. The UCB-type algorithm for contextual bandit problem may suit for the interactive verification.

LinUCB is a UCB-type algorithm for contextual bandit problem, which assumes the problem has linear payoffs (Li et al., 2010). The expected payoff of an action with context x_i is $x_i^\intercal w^*$ with some unknown w^*. Let D be a matrix of dimension $m \times d$, whose rows correspond to m labeled instance be queried so far and b as the corresponding labels. By applying ridge regression, we could get $\hat{w} = (D^\intercal D + I)^{-1} D^\intercal b$, so $x_i^\intercal \hat{w}$ will be the estimation of the reward. According to (Walsh et al., 2009), with probability at least $1 - \delta$, $|x_i^\intercal \hat{w} - x_i^\intercal w^*| \leq \hat{\alpha}\sqrt{x_i^\intercal (D^\intercal D + I_d)^{-1} x_i}$, for any $\delta > 0$, where $\hat{\alpha} = 1 + \sqrt{\ln(2/\delta)/2}$. It makes $\sqrt{x_i^\intercal (D^\intercal D + I_d)^{-1} x_i}$ a suitable upper confidence bound measurement. In every iteration, LinUCB will query the instance x_i with largest $x_i^\intercal \hat{w} + \hat{\alpha}\sqrt{x_i^\intercal (D^\intercal D + I_d)^{-1} x_i}$.

Since the interactive verification does not have the assumption of linear payoff, we use our original base learner SVM instead of ridge regression. We treat confidence term in LinUCB as a term to measure the uncertainty of each instance in unsupervised learning view. If the learner is not certain on the instance, the confidence term will be large; otherwise, it will be small. By using confidence term from LinUCB, we can find the instances that worthy to be explored. The value of confidence term can also help to decide the switching timing between exploration and exploitation. We add the confidence term to the decision value that is produced from SVM and connect these two terms with a parameter α. The scoring function of the UCB approach to interactive verification is

$$S(x_i, w_t) = x_i^\intercal w_t + \alpha\sqrt{x_i^\intercal (D^\intercal D + I_d)^{-1} x_i}.$$

3.5 Discussions

We have now discussed four different approaches to solve interactive verification problems. Among them, the greedy approach could be seen as a special case of the other three approaches. All four approaches all apply greedy approach during exploitation. But these four approaches have different philosophy for exploration. The greedy approach spend all the iterations for exploitation; the exploration method used by RTG is random sampling, which can get unbiased information; the exploration method used by USTG is uncertainty sampling, which is widely used for active learning; UCB uses the confidence term from LinUCB to decide which instances are worthy of being explored and when the learner should do the exploration.

Now we compare the strategies on switching between exploration and exploitation. Greedy approach does not do the switching at all; RTG and USTG share a similar framework by only doing a one-time switching from exploration to exploitation; UCB uses the confidence term to decide the switching between exploration and exploitation auto-

matically. That is, it is possible for UCB to switch between exploration and exploitation several times.

4 Experiment

4.1 Datasets and Experiment Setting

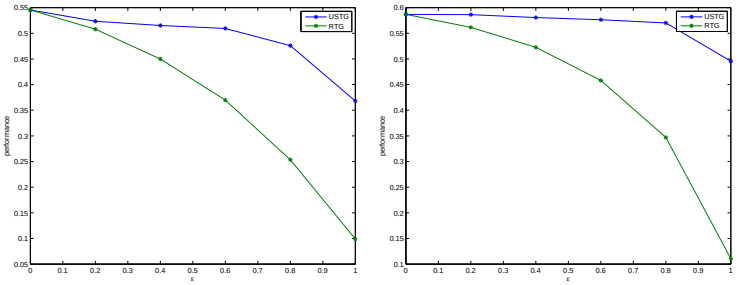
We conduct experiments on eight real-world datasets to compare the performance of the four approaches proposed in Section 3. Table 1 shows the datasets that we use. Among them, the KDD Cup 2008 dataset is a breast cancer screening dataset as discussed in Section 1. As the table shows, the percentages of positive instances, which may greatly affect the performance of interactive verification algorithm, are very different from different datasets. To do a fair comparison, we do the re-sampling on all the datasets to control the percentages of positive instances in each dataset. We separate the positive instances from negative instances in original dataset, and sample P positive instances and N negative instances from corresponding set. For convenience, we set $N = 1000$ all the time and only adjust the value of P in our experiments. We repeat each experiment 1000 times with different initial instances, which include one positive instance and one negative instance. We used (1) as the evaluation metric. The results and the discussions can be seen in following sections. The KDD Cup 2008 dataset will be studied further in Section 4.4.

Table 1. Dataset characteristics

Dataset	Number of instances	Number of positive instances	Positive rate
KDD Cup 2008	102294	623	0.6%
spambase	4601	1813	39.4%
a1a	1605	395	24.6%
cod-rna	59535	19845	33.3%
mushrooms	8124	3916	48.2%
w2a	3470	107	3%
covtype.binary	581012	297711	51.2%
ijcnn1	49990	4853	9.7%

4.2 Effect of ϵ

In this section we demonstrate the effect of different ϵ in RTG and USTG. We conduct experiments on the KDD Cup 2008 dataset with $P = 50, 100$ and $T = 100$. We change the value of ϵ from 0 to 1. The results are shown in Figure 2. The performance decreases when ϵ increase both for RTG and USTG, and $\epsilon = 0$ is one of the best choice. The rest of the datasets show the same trend. RTG and USTG with $\epsilon = 0$ are actually the greedy approach. As our discussion before, the greedy approach spent all the iterations in exploitation. The results that greedy approach has best performance seem to



(a) KDD Cup 2008 with $P = 100$ (b) KDD Cup 2008 with $P = 50$

Fig. 2. The effect of ϵ

suggest that spending queries on improving model quality is not important for interactive verification. Nevertheless, if we take a closer look on greedy approach, we will find out that instances selected by greedy approach could benefit on both verification and model quality.

The story is that, the instance selected by greedy approach the instance with highest possibility to be positive among all the unlabeled instances. It will have the highest probability to be a positive instance, and hence the query is likely to be a successful verification; on the other hand, even if greedy approach queries a negative instance, it may not totally be a bad news. The instance selected by greedy approach is the instance that considered most possible to be positive by current model. The truth that the instance is actually a negative instance is very informative. The query result may greatly improve the model quality. So no matter what result we get from querying the instance selected by greedy approach, we either successfully verify a positive instance or label an informative negative instance. In other word, greedy approach often either does a successful exploitation or does an efficient exploration.

Although greedy approach has such good property in the interactive verification, it still will have poor performance on the dataset shown in Figure 1. The reason that the good property of greedy approach does not work is that the instance selected by greedy approach may actually have low possibility to be positive. It may happen when there is no better choice for greedy approach to select. Consider the biased model shown as dashed line in Figure 1, the instances in negative cluster are considered to be negative instances by the model. But since the instances in bottom positive cluster are misclassified as extremely negative ones, the greedy approach will still select the instance in negative cluster to query. To solve this issue, we should do the exploration when the instance selected by greedy approach does not have high enough possibility to be positive, and do the exploitation when the instance selected by greedy has high enough possibility to be positive. It is actually what UCB does: when the first term in UCB is large, it will do the exploitation; when the first term is small, it will do the exploration. So UCB may be a better choice to solve interactive verification problems than the greedy approach.

4.3 Comparison of All Approaches

In this section, we conduct experiments for comparing four approaches on all eight datasets. We set $P = 50, 100$ and $T = 100$. For RTG and USTG, we set ϵ to be 0.2, the best observed choice among $\epsilon > 0$. For the parameter α in UCB, we consider 0.2 and 0.4. Table 2 shows the result of our experiments. We treat datasets with different P as different datasets. The results show that greedy outperform RTG and USTG. It is consistent to our finding in the previous subsection. The table also shows that the best α for UCB is dataset dependent, so parameter tuning may be necessary for UCB. Generally, $\alpha = 0.2$ is a good choice. UCB with $\alpha = 0.2$ has best performance both for $P = 50$ and $P = 100$ cases. When $P = 50$, UCB totally outperform greedy. But when $P = 100$, although UCB with $\alpha = 0.2$ still has the best performance, the gap between it and greedy is smaller. The reason behind that is when P increase from 50 to 100 while T is still fix to 100, there may be not much iterations left after greedy finish querying the instances with high probability to be positive, so the ability to dynamically switch to the exploration stage will be less significant. The results also show that UCB, which does dynamic switching from the exploration stage to the exploitation stage approach, has better performance than RTG and USTG, which does an one-time switching.

4.4 Real-world Task

In this subsection, we conduct experiments on the KDD Cup 2008 dataset without re-sampling. The KDD Cup 2008 challenge focuses on the problem of early detection of breast cancer from X-ray images of the breast. In this dataset, only 623 out of 102294 ROIs are malignant mass lesions. The percentage of positive instance is only around 0.6%. The P is given by the dataset, which equals to 623. We set T to be 623 and 1243 separately, which are the value of P and twice the P. We do each experiment 20 times. The result is shown in Table 3. Although the difference is small when $T = 623$, UCB apparently has best performance when $T = 1243$. The result is consistent with our experiments on the re-sampled datasets.

5 Conclusion

Interactive verification is a new problem. We pointed out that the trade-off between the learning stage and the verification stage is similar to the trade-off between exploration and exploitation in the contextual bandit problem, and transformed interactive verification to a special contextual bandit problem. We discussed the pros and cons of three basic approaches: greedy, RTG, and USTG, and showed that applying greedy on the interactive verification leads to better results. We also showed the potential risk of the greedy approach for interactive verification, and proposed to adopt UCB, which has been widely used for contextual bandit, to solve interactive verification. UCB avoids the risk that the greedy approach may encounter. The experimental results on re-sampled datasets and a real-world task show that greedy is quite competitive and UCB performs the best among four approaches.

Table 2. Experiment results

Dataset	Algorithm	$P = 50$	$P = 100$
KDD Cup 2008	greedy	0.5868 ± 0.0040 (3)	**0.5454 ± 0.0022 (2)**
	RTG($\epsilon = 0.2$)	0.5615 ± 0.0035 (5)	0.5080 ± 0.0018 (5)
	USTG($\epsilon = 0.2$)	0.5863 ± 0.0032 (4)	0.5235 ± 0.0023 (4)
	UCB($\alpha = 0.2$)	0.5968 ± 0.0031 (2)	0.5434 ± 0.0018 (3)
	UCB($\alpha = 0.4$)	**0.6055 ± 0.0027 (1)**	**0.5467 ± 0.0015 (1)**
spambase	greedy	**0.7467 ± 0.0024 (1)**	**0.6055 ± 0.0012 (1)**
	RTG($\epsilon = 0.2$)	0.7042 ± 0.0020 (4)	0.5422 ± 0.0012 (5)
	USTG($\epsilon = 0.2$)	0.7429 ± 0.0023 (2)	0.5905 ± 0.0012 (2)
	UCB($\alpha = 0.2$)	0.7306 ± 0.0020 (3)	0.5856 ± 0.0013 (3)
	UCB($\alpha = 0.4$)	0.6965 ± 0.0022 (5)	0.5559 ± 0.0013 (4)
a1a	greedy	**0.3883 ± 0.0034 (4)**	0.3754 ± 0.0020 (2)
	RTG($\epsilon = 0.2$)	0.3535 ± 0.0035 (5)	0.3413 ± 0.0018 (5)
	USTG($\epsilon = 0.2$)	**0.3898 ± 0.0035 (3)**	0.3585 ± 0.0018 (4)
	UCB($\alpha = 0.2$)	**0.3915 ± 0.0034 (1)**	**0.3775 ± 0.0019 (1)**
	UCB($\alpha = 0.4$)	**0.3909 ± 0.0031 (2)**	0.3711 ± 0.0019 (3)
cod-rna	greedy	0.7249 ± 0.0027 (3)	0.6251 ± 0.0012 (2)
	RTG($\epsilon = 0.2$)	0.6763 ± 0.0024 (5)	0.5610 ± 0.0012 (5)
	USTG($\epsilon = 0.2$)	0.7155 ± 0.0025 (4)	0.6074 ± 0.0012 (4)
	UCB($\alpha = 0.2$)	**0.7333 ± 0.0024 (1)**	**0.6265 ± 0.0012 (1)**
	UCB($\alpha = 0.4$)	0.7297 ± 0.0025 (2)	0.6236 ± 0.0012 (3)
mushrooms	greedy	0.9710 ± 0.0014 (4)	**0.9125 ± 0.0008 (1)**
	RTG($\epsilon = 0.2$)	0.9715 ± 0.0012 (3)	0.8112 ± 0.0006 (5)
	USTG($\epsilon = 0.2$)	0.9600 ± 0.0008 (5)	0.8776 ± 0.0005 (4)
	UCB($\alpha = 0.2$)	0.9776 ± 0.0007 (2)	0.9109 ± 0.0006 (2)
	UCB($\alpha = 0.4$)	**0.9837 ± 0.0006 (1)**	0.9031 ± 0.0005 (3)
w2a	greedy	0.5944 ± 0.0030 (3)	0.5498 ± 0.0016 (2)
	RTG($\epsilon = 0.2$)	0.5371 ± 0.0032 (5)	0.4933 ± 0.0016 (5)
	USTG($\epsilon = 0.2$)	0.5931 ± 0.0028 (4)	0.5393 ± 0.0015 (3)
	UCB($\alpha = 0.2$)	**0.6160 ± 0.0024 (1)**	**0.5601 ± 0.0013 (1)**
	UCB($\alpha = 0.4$)	0.6064 ± 0.0023 (2)	0.5314 ± 0.3883 (4)
covtype.binary	greedy	0.2202 ± 0.0026 (5)	0.2306 ± 0.0021 (5)
	RTG($\epsilon = 0.2$)	0.2342 ± 0.0027 (3)	0.2388 ± 0.0017 (4)
	USTG($\epsilon = 0.2$)	0.2294 ± 0.0026 (4)	0.2491 ± 0.0021 (3)
	UCB($\alpha = 0.2$)	0.2536 ± 0.0024 (2)	0.2554 ± 0.0021 (2)
	UCB($\alpha = 0.4$)	**0.2798 ± 0.0024 (1)**	**0.2649 ± 0.0021 (1)**
ijcnn1	greedy	0.5220 ± 0.0027 (3)	0.4705 ± 0.0023 (3)
	RTG($\epsilon = 0.2$)	0.4668 ± 0.0034 (5)	0.4247 ± 0.0015 (5)
	USTG($\epsilon = 0.2$)	0.5184 ± 0.0028 (4)	0.4607 ± 0.0019 (4)
	UCB($\alpha = 0.2$)	0.5402 ± 0.0029 (2)	0.4750 ± 0.0021 (2)
	UCB($\alpha = 0.4$)	**0.5598 ± 0.0025 (1)**	**0.4849 ± 0.0018 (1)**
Average Rank	greedy	3.25	2.25
	RTG($\epsilon = 0.2$)	4.38	4.88
	USTG($\epsilon = 0.2$)	3.75	3.5
	UCB($\alpha = 0.2$)	1.75	1.88
	UCB($\alpha = 0.4$)	1.88	2.5

Table 3. KDD Cup 2008

Dataset	Algorithm	$T = 623$	$T = 1243$
KDD Cup 2008	greedy	**0.3649 ± 0.0037**	0.4831 ± 0.0059
	RTG($\epsilon = 0.2$)	0.3062 ± 0.0022	0.4482 ± 0.0023
	USTG($\epsilon = 0.2$)	**0.3659 ± 0.0013**	0.4802 ± 0.0058
	UCB($\alpha = 0.2$)	**0.3660 ± 0.0016**	**0.4917 ± 0.0029**
	UCB($\alpha = 0.4$)	**0.3655 ± 0.0013**	**0.4897 ± 0.0048**

6 Acknowledgment

We thank Profs. Yuh-Jye Lee, Shou-De Lin, the anonymous reviewers, and the members of the NTU Computational Learning Lab for valuable suggestions. This work is mainly supported by National Science Council (NSC 101-2628-E-002-029-MY2) of Taiwan.

References

Auer, P., Cesa-Bianchi, N., Fischer, P., Informatik, L.: Finite-time analysis of the multi-armed bandit problem. Machine Learning 2-3, 235–256 (2000)

Langford, J., Zhang, T.: The epoch-greedy algorithm for contextual multi-armed bandits. In: Proceedings of the Conference on Neural Information Processing Systems (2007)

Lewis, D.D., Gale, W.A.: A sequential algorithm for training text classifiers. In: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval. pp. 3–12 (1994)

Li, L., Chu, W., Langford, J., Schapire, R.E.: A contextual-bandit approach to personalized news article recommendation. In: Proceedings of the International Conference on World Wide Web. pp. 661–670 (2010)

Li, M., Zhou, Z.H.: Improve computer-aided diagnosis with machine learning techniques using undiagnosed samples. IEEE Transactions on Systems, Man, and Cybernetics, Part A 37(6), 1088–1098 (2007)

Rangayyan, R.M., Fabio, J.A., Desautels, J.L.: A review of computer-aided diagnosis of breast cancer: Toward the detection of subtle signs. Journal of the Franklin Institute 344(3–4), 312–348 (2007)

Sabato, S., Sarwate, A.D., Srebro, N.: Auditing: Active learning with outcome-dependent query costs. In: Proceedings of the Conference on Neural Information Processing Systems (2013)

Settles, B.: Active learning literature survey. Tech. rep., University of Wisconsin–Madison (2009)

Tong, S., Koller, D.: Support vector machine active learning with applications to text classification. Journal of Machine Learning Research 2, 45–66 (2001)

Walsh, T.J., Szita, I., Diuk, C., Littman, M.L.: Exploring compact reinforcement-learning representations with linear regression. In: Proceedings of the Conference on Uncertainty in Artificial Intelligence. pp. 591–598 (2009)