

國立臺灣大學電機資訊學院資訊工程學系

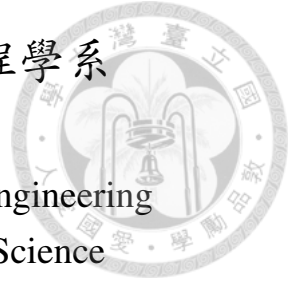
碩士論文

Department of Computer Science and Information Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis



以機器學習方法進行互動驗證

Machine Learning Approaches for Interactive Verification

周育正

Yu-Cheng Chou

指導教授：林軒田 博士

Advisor: Hsuan-Tien Lin, Ph.D.

中華民國 102 年 7 月

July, 2013



## 致謝

感謝林軒田老師、536和217的夥伴們以及兩年來在各方面幫助過我的人們。



## 中文摘要

驗證問題是一個有很多應用且需要使用人力的問題。機器學習可以減少花費在驗證問題上的人力。透過結合驗證問題中的學習和驗證兩個階段，我們提出一個稱做“互動驗證”的新問題。這個新問題可以藉著自由分配學習和驗證來更有效的運用人力。我們提出使用情境式拉霸問題 (Contextual Bandit Problem) 中的上信賴界 (Upper Confidence Bound) 方法來解決互動驗證問題。在真實世界資料上的實驗結果證實了上信賴界可以有效的解決互動驗證問題。

關鍵詞：機器學習、主動學習、情境式拉霸問題



# Abstract

The verification problem comes with many applications and requires human efforts. Machine learning can help reduce human efforts spent on verification. By combining the learning and verification stages in a verification problem, we formalize the needs as a new problem called interactive verification. The problem allows an algorithm to flexibly use the limited human resource on learning and verification together. We propose to adopt 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 has superior performance on interactive verification on many real-world datasets.

**Keywords:** Machine Learning, Active Learning, Contextual Bandit Problem



# Contents

致謝	iii
中文摘要	iv
Abstract	v
<b>1 Introduction</b>	<b>1</b>
<b>2 Problem Setting</b>	<b>4</b>
2.1 Interactive Verification Problem . . . . .	4
2.2 Motivation . . . . .	4
2.3 Comparison to Active Learning Problem . . . . .	7
2.4 Comparison to Contextual Bandit Problem . . . . .	8
<b>3 Approaches</b>	<b>10</b>
3.1 Greedy Approach . . . . .	10
3.2 Random then Greedy . . . . .	11
3.3 Uncertainty Sampling then Greedy . . . . .	12
3.4 Upper Confidence Bound . . . . .	13
3.5 Discussions . . . . .	14
<b>4 Experiment</b>	<b>15</b>
4.1 Dataset Generation and Experiment Setting . . . . .	15
4.2 Effect of $\epsilon$ . . . . .	16
4.3 Comparison of All Approaches . . . . .	18
4.4 Real-world Task . . . . .	19
<b>5 Conclusion</b>	<b>20</b>
<b>Bibliography</b>	<b>20</b>



# List of Figures

2.1	One-dimensional example . . . . .	6
3.1	Artificial dataset . . . . .	11
4.1	The effect of $\epsilon$ . . . . .	16



# List of Tables

4.1	Dataset characteristics . . . . .	15
4.2	Experiment results . . . . .	18
4.3	KDD Cup 2008 . . . . .	19



# Chapter 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. 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 possibly asymptomatic, we can save radiology experts a lot of efforts. And one possible way is to let computers select the subset automatically.

Computer-aided diagnosis (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

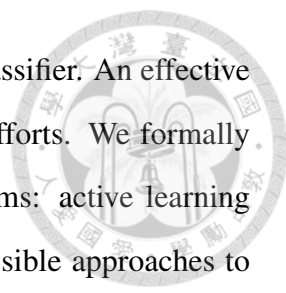


by selecting worthy-verified ROIs for radiology experts. Since the goal is to verify the malignant ROIs, we called the problem that requires human experts to verify something (malignant ROIs) selected by computers (CAD system) as “verification problem”.

Aside from breast cancer screening, verification problem has many applications. For example, an administrator of a social media platform may want to detect misbehaved users and suspend their accounts [Villatoro-Tello et al., 2012]. The number of users is usually huge, making it infeasible or challenging to check all the users by real humans, yet automatically suspending users’ accounts may lead to complaints. A better way is to hire humans to verify the users that are classified as misbehaved and let the humans decide to suspend the account or not.

In a verification problem, there are two stages that require the efforts of human experts: labeling stage and verification stage. These two stages are difference from the point of view of the system. In labeling stage, the system requests label of a ROI for learning; in verification stage, the system request diagnosis on a ROI that is considered to be positive (malignant) for verification. But actually, these two stages are similar from human experts’ point of view. Both of them require radiology experts to diagnose on a ROI and return the diagnosis. We called the request of diagnosis as a “query” in the verification problem. Given the similarity between the labeling stage and verification stage, we propose combining 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 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? It is the question we want to answer in this work.

In this paper, we formalize the question above by defining a new problem called the interactive verification problem. It is a procedure that does verification through the interaction between the system and the human experts. By interacting with humans, the system aims to verify as many possible positive instances as possible within limited query bud-



get, 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. We formally identify the similarity of this problem to two very different problems: active learning problem and contextual bandit problem. Then, we propose four possible approaches to solve interactive verification problem based on the similarity. In particular, one of the four is called the upper confidence bound (UCB), which is borrowed from the contextual bandit problem. 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 thesis is organized as follows. In Chapter 2 we define the interactive verification problem and compare it to other problem. We propose four approaches to solve the problem in Chapter 3. Finally, we present the experiment results in Chapter 4 and conclude our work in Chapter 5.



# Chapter 2

## Problem Setting

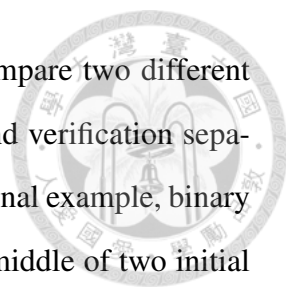
### 2.1 Interactive Verification Problem

Given a set of instances  $X = \{x_1, \dots, x_m\}$ , 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, 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

$$\sum_{t=1}^T \frac{[Y(s_t) = 1]}{|P|}. \quad (2.1)$$

### 2.2 Motivation

Before we propose approaches for general interactive verification problems, we show a special case of a one-dimensional separable dataset to motivate our further study. As shown in Figure 2.2, there are  $m$  instances on a line and a hypothesis (threshold) to perfectly classify all the instances. The learner is required to decide which instance to query



in every iteration to efficiently verify the positive instances. We compare two different approaches to solve this problem. Approach 1 considers learning and verification separately. To learn the underlying hypothesis in a separable one-dimensional example, binary search is the optimal strategy. Approach 1 starts querying from the middle of two initial instances, as the blue arrow shown in the figure. When binary search is done, the learner will learn the optimal hypothesis. After that, the learner can do the verification according to the optimal hypothesis without wasting any query. Approach 1 seems to be optimal both in learning stage and verification stage.

Approach 2 considers learning and verification together. Although the learner does not know the optimal hypothesis when there are only two labeled instances, it is apparent that the leftmost instance is a positive instance. So the learner can start verification with high confidence though learning is not even started yet. Approach 2 queries from the leftmost instance for verification, as the green arrow shown in figure. By keeping verifying the instance that the learner is most certain to be positive, approach 2 queries the instances according to the order from left to right, until a negative instance is queried. Another good thing about approach 2 is that when the learner finishes the verification, it can also learn the underlying hypothesis from the feedback of verification.

To compare approach 1 and approach 2, we compute the human efforts spent on verification when all the positive instances are verified. Since the number of queries spent on verifying positive instances is fixed, we can equivalently check the number of queries wasted on negative instances. Binary search used in approach one require  $O(\log m)$  queries to learn the underlying hypothesis, and hence may need to query as many as  $O(\log m)$  negative instances. Then, in the verification stage, since the learner does verifications according to the optimal hypothesis, no more queries will be wasted. The overall number of queries wasted on negative instances when using approach one is  $O(\log m)$ . When using approach 2, only one negative instance will be queried. The number of queries wasted on negative instances when using approach 2 is  $O(1)$ . Thus, approach 2 is better than approach 1. This dataset demonstrates the benefit of combining the learning stage and verification stage. By starting verification even when the learner does not know

about the optimal hypothesis and learning from feedbacks of verifications, approach 2 shows how the flexibility that the learner gets from combining these two stage can be useful.

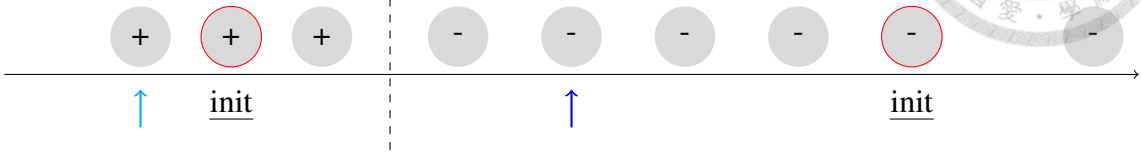


Figure 2.1: One-dimensional example

We now outline a general framework for solving the interactive verification problem. In an interactive verification problem, we focus on how to decide which instance is to be queried next. 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. In every iteration, the learner queries the instance with the largest score. The general framework shown in Algorithm 1. By defining the scoring function, we define the behavior of an approach to interactive verification problem.

---

**Algorithm 1** General approach to interactive verification problem

---

**Require:** Base learner,  $B$ ; Unlabeled instances,  $U$ ; Labeled instances,  $L$ ; Amount of iterations,  $T$ ;

- 1: **for**  $t = 1$  to  $T$  **do**
  - 2:   model  $M = B(L)$
  - 3:   **for all**  $u \in U$  **do**
  - 4:     Compute scoring function:  $S(u, M)$
  - 5:   **end for**
  - 6:    $s_t = \arg \max_u S(u, M)$
  - 7:    $L = L \cup \{(s_t, Y(s_t))\}$
  - 8:    $U = U \setminus \{s_t\}$
  - 9: **end for**
- 

In general, there are two possible criteria we may want to consider in our scoring function: the possibility that an instance is positive and the information amount carried by an instance. These two criteria correspond to the verification stage and learning stage respectively. In our one-dimensional example, binary search used by approach 1 is focusing on querying the informative instances; on the other hand, approach 2 keeps querying the instance with highest possibility to be positive. In our one-dimensional case, keep querying the instance with the highest possibility to be positive is the optimal solution.

But for more general cases with higher dimensions and noise, the model learned from two initial instances may be biased. Then, querying some informative instances may be a better strategy than keep doing verifications according to a biased model.

The two criteria can be conflicting. An instance with high possibility to be positive is the instance that the model has high confidence on correctly classifying; on the other hand, an instance with large information amount is usually the instance that the model has low confidence on correctly classifying. So the difficulty of interactive verification is to balance between the trade-off of the verification stage and learning stage. If we only focus on improving the model quality and do not reserve enough iterations for verifications, then only few positive instances will be verified; on the other hand, if we spend all the iterations for verifications, we may suffered from biased model. The trade-off between model quality and number of iterations reserved for verifications is the key to design a scoring function for solving interactive verification problem. To design a better scoring function, we may borrow the idea from existing algorithms of other related problems.

## 2.3 Comparison to Active Learning Problem

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 learn an accurate model with few labeled instances. Active learning can reduce human efforts in learning stage. Pool-based active learning is a widely used setting for the active learning problem, which assumes 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 as same as the interactive verification problem: both of them allow the learner to query an instance in each iteration. The difference between them is the goal of these two problems. Active learning problem focuses on getting an accurate model; on the other hand, interactive verification problem aims to maximize the number of verified positive instances. Although the goal is different from these two problems, the measurement of information amount used in algorithms for

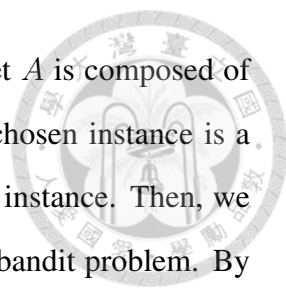
active learning problem can be used in solving interactive verification problem.



## 2.4 Comparison to Contextual Bandit Problem

Contextual bandit problem is a form of multi-armed bandit problem. The multi-armed bandit problem is a problem that a player faces some slot machines and wants to decide in which order to play them [Auer et al., 2000]. In every iteration, player could pick one action, which is the choice of a slot machine, from the action set  $A$ . After the action is acted, the player will receive a randomize reward decided by the distribution under the correspond slot machine. The goal is to maximize the rewards received by the player after a given number of iterations. The feature 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 acted, we will have no information about it. Thus, it is necessary to spend some iterations to explore the actions we are not familiar with. Somehow only doing exploration cannot maximize the total rewards, and we also need 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 exploration and exploitation. Although the setting of multi-armed bandit problem is very different from the interactive verification problem, the trade-off between exploration and exploitation is similar to the trade-off between the learning stage and verification stage in the interactive verification problem.

In a contextual bandit problem, the learner will receive a context from the environment in every iteration [Langford and Zhang, 2007]. The rest setting is almost as the same as the multi-armed bandit problem. In a contextual bandit problem, we assume the expected reward is a function of the context. By learning these functions under each action, the learner could make optimal choice when the new context arrives. The contexts added in contextual bandit problem make it possible to connect the problem to supervised learning problems, especially the interactive verification problem. In here we will show how to map an interactive verification problem to a contextual bandit problem. To do the mapping, we need to define the context, actions set and reward in the interactive verification



problem. The context will be the feature of instances. The action set  $A$  is composed of choosing each unlabeled instance to query. The reward is 1 if the chosen instance is a positive instance; the reward is 0 if the chosen instance is a negative instance. Then, we successfully map an interactive verification problem to a contextual bandit problem. By optimizing the mapped contextual bandit problem, we can optimize the original interactive verification problem.

Although we find the similarity between contextual bandit problem and interactive verification problem, there is still a big difference between them. In a contextual bandit problem, each action is allowed to be selected several times. The actions that are more likely to produce high rewards could be selected more often. In interactive verification problem, each instance is supposed to be queried at most once. So in a contextual bandit problem mapped from an interactive verification problem, each action can be selected at most once. The difference make it difficult to directly apply existing algorithms for contextual bandit problem on an interactive verification problem. But the idea of how to deal with the trade-off between exploration and exploitation can help us find the balance between learning stage and verification stage.





## Chapter 3

# Approaches

In this chapter, we propose four approaches to solve the interactive verification problem, and discuss their relations. The practical comparison will be made in Chapter 4. We use support vector machine (SVM) with linear kernel as our base learner, and denote  $w_t$  to be the model we get from base learner in the beginning of every iteration.

### 3.1 Greedy Approach

The goal of our problem is to verify as much 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, it is the instance with largest decision value. That is, the scoring function of the greedy approach is

$$S(x_i, w_t) = x_i^T w_t.$$

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 3.1. There are two clusters of red positive instances and one big cluster of blue negative instances in the figure.

Without loss 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 use some amount of exploration to help us find the instances in the bottom positive cluster.

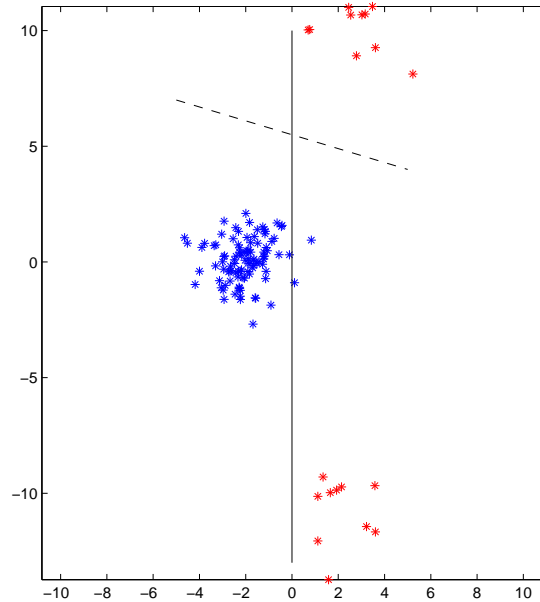


Figure 3.1: Artificial dataset

### 3.2 Random then Greedy

In the previous section we discuss the risk of not doing exploration. In here we propose an approach using random as exploration method to solve the interactive verification problem: 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 the previous section for exploitation (verification). In this approach we do an one-time switching from exploration to exploitation. We

use the parameter  $\epsilon$  decides the ratio between exploration and exploitation. That is, the scoring function of RTG is

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



### 3.3 Uncertainty Sampling then Greedy

As the discussion in Section 2.3, the setting of the interactive verification problem is pretty similar to the active learning problem. It is natural to wonder if we can apply algorithms for the active learning on the interactive verification. Uncertainty sampling is one of the most commonly used algorithm to solve active learning problem [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 learning model, it queries the instance closest to the linear decision boundary in a SVM model [Tong and Koller, 2001].

To apply the uncertainty sampling on interactive verification problem, 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 method. We call this approach uncertainty sampling then greedy (USTG). The scoring function of USTG is

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

Uncertainty sampling may suffer from a biased model like greedy approach. With a model with 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 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 interactive verification problem. But as our discussion in Section 2.4, we can map an interactive verification problem to a contextual bandit problem. The UCB-type algorithm for contextual bandit problem may suit for interactive verification problem.

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  $x_i^\top w^*$  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  $\hat{w} = (D^\top D + I)^{-1} D^\top b$ , so  $x_i^\top \hat{w}$  will be the estimation of the reward. According to [Walsh et al., 2009], with probability at least  $1 - \delta$ ,  $|x_i^\top \hat{w} - x_i^\top w^*| \leq \hat{\alpha} \sqrt{x_i^\top (D^\top D + I_d)^{-1} x_i}$ , for any  $\delta > 0$ , where  $\hat{\alpha} = 1 + \sqrt{\ln(2/\delta)/2}$ . It makes  $\sqrt{x_i^\top (D^\top D + I_d)^{-1} x_i}$  a suitable upper confidence bound measurement. In every iteration, LinUCB will query the instance  $x_i$  with largest  $x_i^\top \hat{w} + \hat{\alpha} \sqrt{x_i^\top (D^\top D + I_d)^{-1} x_i}$ .

Since the interactive verification problem 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  $\alpha$ . The scoring function of UCB in interactive verification problem is

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



### 3.5 Discussions

In this chapter we proposed four different approaches to solve the interactive verification problem. Among them, the greedy approach could be seen as a special case of the other three approaches. All four approaches all apply greedy approach in exploitation stage. But these four approaches have different philosophy in the exploration stage. The greedy approach spend all the iterations in exploitation stage; the exploration method used by RTG is random method, which can get unbiased information; the exploration method used by USTG is uncertainty sampling, which is a widely used algorithm in 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 stage and exploitation stage. Greedy approach does not do the switching at all, since it does the exploitation all the time; RTG and USTG share a similar framework, only do an one-time switching from exploration stage to exploitation stage; UCB uses the confidence term to decide the switching between exploration and exploitation automatically. It is possible for UCB to switch between exploration and exploitation stage several times. We will see the practical comparisons of these four approaches in Chapter 4.



# Chapter 4

## Experiment

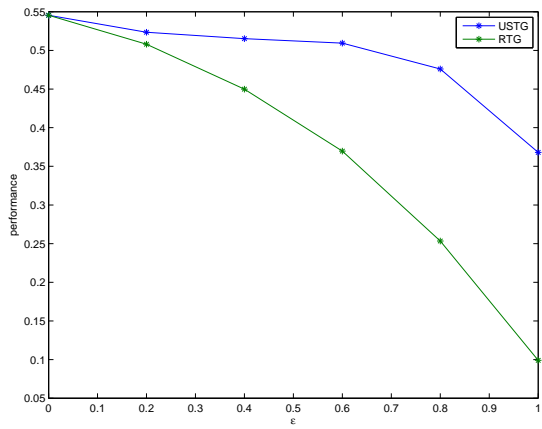
### 4.1 Dataset Generation and Experiment Setting

We conduct experiments on six real-world datasets to compare the performance of the four approaches we proposed in Chapter 3. Table 4.1 shows the datasets that we use. Among them, *KDDCup2008* is a breast cancer screening dataset. As the table shows, the percentages of positive instances, which may greatly affect the performance, are very different from other datasets. To do a fair comparison, we do the resampling 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 convenient, we set  $N = 1000$  all the time and only adjust the value of  $P$  in our experiments. We repeat each experiment 1000 times. The results and the discussions can be seen in following sections.

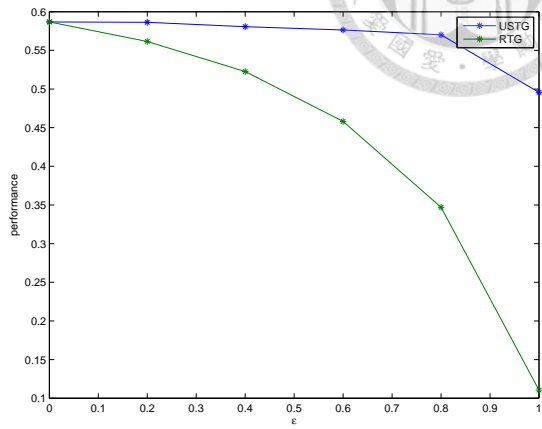
Table 4.1: Dataset characteristics

Dataset	Number of instances	Number of Positive instances	Percentage of positive instances
KDDCup2008	102294	623	0.6%
spambase	4601	1813	39.4%
ala	1605	395	24.6%
cod-rna	59535	19845	33.3%
mushrooms	8124	3916	48.2%
w2a	3470	107	3%

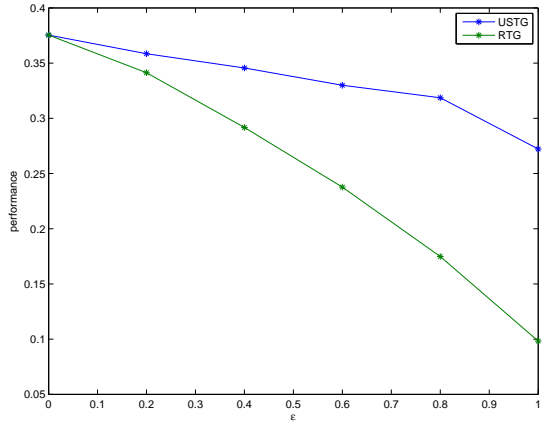
## 4.2 Effect of $\epsilon$



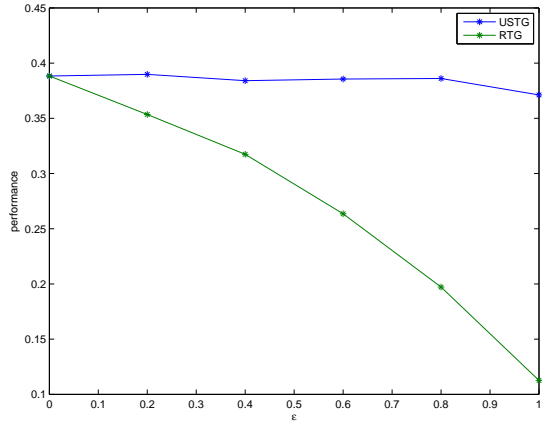
(a) KDDCup2008 with  $P = 100$



(b) KDDCup2008 with  $P = 50$



(c) a1a with  $P = 100$



(d) a1a with  $P = 50$

Figure 4.1: The effect of  $\epsilon$

In this section we demonstrate the effect of different  $\epsilon$  in RTG and USTG. We conduct experiments on dataset *KDDCup2008* and *a1a* with  $P = 50$  and  $P = 100$ . We change the value of  $\epsilon$  from 0 to 1. The results are shown in Figure 4.2. The performance decreases when  $\epsilon$  increase both for RTG and USTG. In all the cases, the best choice is  $\epsilon = 0$ , and it is actually the greedy approach. As our discussion before, the greedy approach spent all the iterations in exploitation. The results that greedy approach outperforms both RTG and USTG seem to suggest that spending queries on improving model quality is not important in the interactive verification problem. But if we take a closer look on greedy approach, we will find out that instances selected by greedy approach could benefit on both verification

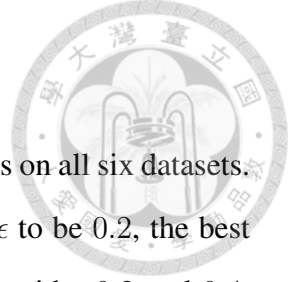
and model quality.



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 interactive verification problem, it still will have poor performance on the dataset shown in Figure 3.1. 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 3.1, 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 problem than the greedy approach.





### 4.3 Comparison of All Approaches

In this section, we conduct experiments for comparing four approaches on all six datasets. We set  $P$  to be 50 and 100 separately. For RTG and USTG, we set  $\epsilon$  to be 0.2, the best observed choice among  $\epsilon > 0$ . For the parameter  $\alpha$  in UCB, we consider 0.2 and 0.4. Table 4.2 shows the result of our experiments. We treat datasets with different  $P$  as different datasets. In five datasets out of twelve datasets, greedy has best performance. It is consistent to our conclusion in the previous section that greedy approach is a very competitive approach for interactive verification. UCB with  $\alpha = 0.2$  outperforms greedy in seven datasets and has a tie with greedy approach in two datasets. It shows that UCB is indeed a better choice to solve interactive verification problem than greedy approach. The results also show that UCB, which does dynamic switching from exploration stage to exploitation stage approach, has better performance than RTG and USTG, which does one-time switching.

Table 4.2: Experiment results

Dataset	Algorithm	$P = 50$	$P = 100$
KDDCup2008	greedy	0.5868 $\pm$ 0.0040	<b>0.5454 <math>\pm</math> 0.0022</b>
	RTG( $\epsilon = 0.2$ )	0.5615 $\pm$ 0.0035	0.5080 $\pm$ 0.0018
	USTG( $\epsilon = 0.2$ )	0.5863 $\pm$ 0.0032	0.5235 $\pm$ 0.0023
	UCB( $\alpha = 0.2$ )	0.5968 $\pm$ 0.0031	0.5434 $\pm$ 0.0018
	UCB( $\alpha = 0.4$ )	<b>0.6055 <math>\pm</math> 0.0027</b>	<b>0.5467 <math>\pm</math> 0.0015</b>
spambase	greedy	<b>0.7467 <math>\pm</math> 0.0024</b>	<b>0.6055 <math>\pm</math> 0.0012</b>
	RTG( $\epsilon = 0.2$ )	0.7042 $\pm$ 0.0020	0.5422 $\pm$ 0.0012
	USTG( $\epsilon = 0.2$ )	0.7429 $\pm$ 0.0023	0.5905 $\pm$ 0.0012
	UCB( $\alpha = 0.2$ )	0.7306 $\pm$ 0.0020	0.5856 $\pm$ 0.0013
	UCB( $\alpha = 0.4$ )	0.6965 $\pm$ 0.0022	0.5559 $\pm$ 0.0013
ala	greedy	<b>0.3883 <math>\pm</math> 0.0034</b>	0.3754 $\pm$ 0.0020
	RTG( $\epsilon = 0.2$ )	0.3535 $\pm$ 0.0035	0.3413 $\pm$ 0.0018
	USTG( $\epsilon = 0.2$ )	<b>0.3898 <math>\pm</math> 0.0035</b>	0.3585 $\pm$ 0.0018
	UCB( $\alpha = 0.2$ )	<b>0.3915 <math>\pm</math> 0.0034</b>	<b>0.3775 <math>\pm</math> 0.0019</b>
	UCB( $\alpha = 0.4$ )	<b>0.3909 <math>\pm</math> 0.0031</b>	0.3711 $\pm$ 0.0019
cod-rna	greedy	0.7249 $\pm$ 0.0027	0.6251 $\pm$ 0.0012
	RTG( $\epsilon = 0.2$ )	0.6763 $\pm$ 0.0024	0.5610 $\pm$ 0.0012
	USTG( $\epsilon = 0.2$ )	0.7155 $\pm$ 0.0025	0.6074 $\pm$ 0.0012
	UCB( $\alpha = 0.2$ )	<b>0.7333 <math>\pm</math> 0.0024</b>	<b>0.6265 <math>\pm</math> 0.0012</b>
	UCB( $\alpha = 0.4$ )	0.7297 $\pm$ 0.0025	0.6236 $\pm$ 0.0012
mushrooms	greedy	0.9710 $\pm$ 0.0014	<b>0.9125 <math>\pm</math> 0.0008</b>
	RTG( $\epsilon = 0.2$ )	0.9715 $\pm$ 0.0012	0.8112 $\pm$ 0.0006
	USTG( $\epsilon = 0.2$ )	0.9600 $\pm$ 0.0008	0.8776 $\pm$ 0.0005
	UCB( $\alpha = 0.2$ )	0.9776 $\pm$ 0.0007	0.9109 $\pm$ 0.0006
	UCB( $\alpha = 0.4$ )	<b>0.9837 <math>\pm</math> 0.0006</b>	0.9031 $\pm$ 0.0005
w2a	greedy	0.5944 $\pm$ 0.0030	0.5498 $\pm$ 0.0016
	RTG( $\epsilon = 0.2$ )	0.5371 $\pm$ 0.0032	0.4933 $\pm$ 0.0016
	USTG( $\epsilon = 0.2$ )	0.5931 $\pm$ 0.0028	0.5393 $\pm$ 0.0015
	UCB( $\alpha = 0.2$ )	<b>0.6160 <math>\pm</math> 0.0024</b>	<b>0.5601 <math>\pm</math> 0.0013</b>
	UCB( $\alpha = 0.4$ )	0.6064 $\pm$ 0.0023	0.5314 $\pm$ 0.3883

## 4.4 Real-world Task



Table 4.3: KDD Cup 2008

Dataset	Algorithm	$T = 623$	$P = 1243$
KDDCup2008	greedy	<b>0.3649</b> $\pm$ <b>0.0037</b>	0.4831 $\pm$ 0.0059
	RTG( $\epsilon = 0.2$ )	0.3062 $\pm$ 0.0022	0.4482 $\pm$ 0.0023
	USTG( $\epsilon = 0.2$ )	<b>0.3659</b> $\pm$ <b>0.0013</b>	0.4802 $\pm$ 0.0058
	UCB( $\alpha = 0.2$ )	<b>0.3660</b> $\pm$ <b>0.0016</b>	<b>0.4917</b> $\pm$ <b>0.0029</b>
	UCB( $\alpha = 0.4$ )	<b>0.3655</b> $\pm$ <b>0.0013</b>	<b>0.4897</b> $\pm$ <b>0.0048</b>

In this section, we conduct experiments on *KDDCup2008* dataset without resampling. 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 4.4. 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 resampled datasets.



## Chapter 5

### Conclusion

In this work we first discussed the importance and provided some applications of the verification problem. Then, we proposed a new problem: interactive verification. By combining the learning stage and verification stage, the learner gets the flexibility to utilize each query effectively. The interactive verification problem can reduce overall human efforts. We pointed out that the trade-off between learning stage and verification stage is similar to the trade-off between exploration and exploitation in the bandit problem, and mapped the interactive verification problem to the contextual bandit problem. We proposed four approaches to solve interactive verification problem: greedy, RTG, USTG, and UCB. Among them, UCB borrows the idea from an algorithm of contextual bandit problem. According to the experiment results on resampled datasets and a real-world task, greedy method is competitive and UCB performs the best among four approaches.



## Bibliography

Peter Auer, Nicolò Cesa-Bianchi, Paul Fischer, and Lehrstuhl Informatik. Finite-time analysis of the multi-armed bandit problem. Machine Learning, (2-3):235–256, 2000.

John Langford and Tong Zhang. The epoch-greedy algorithm for contextual multi-armed bandits. In Proceedings of the Conference on Neural Information Processing Systems, 2007.

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

Lihong Li, Wei Chu, John Langford, and Robert E. Schapire. A contextual-bandit approach to personalized news article recommendation. In Proceedings of the International Conference on World Wide Web, pages 661–670, 2010.

Ming Li and Zhi-Hua Zhou. 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.

R. M. Rangayyan, J. A. Fabio, and J. L. Desautels. 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.

Burr Settles. Active learning literature survey. Technical report, University of Wisconsin–Madison, 2009.

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

Esau Villatoro-Tello, Antonio Jua rez Gonza lez, Hugo Jair Escalante, Manuel Montes y Go mez, and Luis Villasen or Pineda. A two-step approach for effective detec- tion of misbehaving users in chats. In Proceedings of the Conference and Labs of the Evaluation(Online Working Notes/Labs/Workshop), 2012.

Thomas J. Walsh, Istvan Szita, Carlos Diuk, and Michael L. Littman. Exploring compact reinforcement-learning representations with linear regression. In Proceedings of the Conference on Uncertainty in Artificial Intelligence, pages 591–598, 2009.