New Generation Computing, 8 (1991) 337-347
OHMSHA, LTD. and Springer-Verlag

Teachability in Computational Learning

Ayumi SHINOHARA and Satoru MIYANO
Research Institute of Fundamental Information Science,
Kyushu University 33, Fukuoka 812, Japan.

Received 27 November 1990

Abstract This paper considers computational learning from the viewpoint of teaching. We introduce a notion of teachability with which we establish a relationship between the learnability and teachability. We also discuss the complexity issues of a teacher in relation to learning.

Keywords: Teaching, Computational Learning, PAC-learning, Computational Complexity.

1 Introduction

Many attentions have been directed to the learning theory based on computational complexity theory. However, few approaches have been taken to the problem from the viewpoint of teaching [9].

In this paper we define a notion of teachability. It formalizes the possibility that a teacher can make *all* consistent learners understand a concept by using a small number of examples. We establish a relationship between the notions of teachability and learnability.

For this purpose, we first discuss the notions of learnability and learnability from selected examples simply based on the number of examples required to learn. The learnability from selected examples introduced here is one intended for teaching while the learnability is one studied as the PAC-learnability [10]. These two learnabilities can be seen to be equivalent and any concept class teachable by examples is learnable. However, we show that there is a concept class which is learnable but not teachable by examples.

We also relate these learnabilities with the time required for learning. We show that there is a gap between the polynomial-time learnability from selected examples and the polynomial-time learnability under the assumption of **RP**≠**NP**. These results mean that time for learning is important to consider the effect of teaching.

The complexity issues of a teacher are investigated in relation to learning. We regard the work of a teacher as selecting a small size set of good examples called a key. Then the problem of finding a minimum size key is shown to be **NP**-complete. On the other hand, there is a polynomial-time approximation algorithm for it. We also present two concept classes. One is easy to teach and learn. The other is easy to teach but hard to learn.

Although the proofs for these results are technically easy, this paper clarifies some fundamental problems which arise naturally when we consider the effect of teaching to learning.

2 Preliminaries on Learnabilities

Let $X = \Sigma^*$ be the set of all strings on the binary alphabet $\Sigma = \{0, 1\}$. X_n denotes the set of all strings of length n or less for $n \geq 0$. A *concept* c is a subset of X. Viewed in another way, a concept is a function $c : \Sigma^* \to \{0, 1\}$, where $c(x) = 1$ implies x is in the concept and $c(x) = 0$ otherwise. A concept means a function or a set depending on the context. A *concept class* is a nonempty set $C \subseteq 2^X$ of concepts. For a concept $c \in C$ and an integer $n \geq 0$, we define the *nth subclass* of C by $C_n = \{c_n \mid c \in C\}$, where $c_n = c \cap X_n$. An *example* on $x \in X$ for a concept c is a pair $\langle x, c(x)\rangle$. For $\bar{x} = (x_1, \ldots, x_m) \in X^m$ and a concept c, the *sample* of size m on $\bar{x}$ is defined by $sam_c(\bar{x}) = (\langle x_1, c(x_1)\rangle, \ldots, \langle x_m, c(x_m)\rangle)$. The *sample space* of C is defined to be $S_C = \{sam_c(\bar{x}) \mid c \in C, m \geq 0, \bar{x} \in X^m\}$. For convenience, we assume hereafter that a polynomial $p(x_1, \ldots, x_n)$ is nondecreasing with respect to each argument x_i, and its value is rounded up to an integer $\lceil p(x_1, \ldots, x_n)\rceil$.

In this section we discuss two kinds of learnabilities which are defined by the existence of a learner identifying the given concept class. One is the usual learnability introduced in [10]. The other is concerned with teaching. It is called the learnability from selected examples, which formalizes a view that there is at least one learner who can identify the target concept if appropriate examples are chosen and given to the learner by her/his teacher.

Let C be a concept class. A mapping $A : S_C \to C$ is called a *learn mapping* for C. $\mathcal{A}_C$ denotes the set of all learn mappings for C. For $A \in \mathcal{A}_C$ and $c \in C$, a *hypothesis* on $sam_c(\bar{x})$ by A is the concept $h = A(sam_c(\bar{x}))$, where $\bar{x} \in X^m$. For a probability distribution P on X_n, we define $error_{A,c,P}(\bar{x}) = P(c \oplus h)$, where the probability on $X - X_n$ is assumed to be 0 and $c \oplus h$ denotes the symmetric difference $c \cup h - c \cap h$.

Definition 1 Let C be a concept class and A be a learn mapping for C. We say that a concept class C is *learnable* by A if there exists a polynomial $p(\cdot,\cdot,\cdot)$ satisfying the condition (1):

(1) For any integer $n \geq 0$, any concept $c \in C$, any real numbers ε, δ $(0 < \varepsilon, \delta < 1)$, and any probability distribution P on X_n, the following inequality holds:

$$P^{p(n,1/\varepsilon,1/\delta)}\left(\bar{x} \in X_n^{p(n,1/\varepsilon,1/\delta)} \;\middle|\; error_{A,c,P}(\bar{x}) < \varepsilon\right) > 1 - \delta.$$

We say that C is *learnable* if there exists such learn mapping $A \in \mathcal{A}_C$.

We are interested in whether the learnability can be possibly changed if a helpful teacher selects good examples for a learner. To formally discuss this matter, we define the learnability from selected examples as follows.

Definition 2 Let C be a concept class and A be a learn mapping for C. We say that a concept class C is *learnable from selected examples* by A if there exists a polynomial $p(n)$ satisfying the condition (2):

(2) For any integer $n \geq 0$ and any concept $c \in C$, there exists $\bar{x} \in X_n^{p(n)}$ such that $h_n = c_n$, where $h = A(sam_c(\bar{x}))$.

We say that C is *learnable from selected examples* if there exists such learn mapping $A \in \mathcal{A}_C$.

Natarajan [6, 7] introduced an interesting complexity measure for concept classes.

Definition 3 ([6, 7]) For a concept class C and $n \geq 0$, the *dimension* of the nth subclass C_n is defined by $\dim C_n = \log_2 |C_n|$. We say that a concept class C is of *polynomial dimension* if there exists a polynomial $d(n)$ such that $\dim C_n \leq d(n)$ for all $n \geq 0$.

The following proposition summarizes the equivalence of the above learnabilities in this framework.

Proposition 1 Let C be a concept class. Then the following statements are equivalent:

(a) C is learnable from selected examples.

(b) C is learnable.

(c) C is of polynomial dimension.

Proof (a) $\Rightarrow$ (c) Assume that C is learnable from selected examples by a learn mapping $A \in \mathcal{A}_C$. Let $p(n)$ be the polynomial which satisfies the condition (2) of Definition 2. For each $n \geq 0$, let S_n be the set of samples of size $p(n)$ on X_n, i.e., $S_n = \{sam_c(\bar{x}) \mid \bar{x} \in X_n^{p(n)}, c \in C\}$. Then $|S_n| \leq (2|X_n|)^{p(n)}$. Let H_n be the set of hypotheses of A for the samples of size $p(n)$, i.e., $H_n = \{A(s) \in C \mid s \in S_n\}$. Then $|H_n| \leq |S_n|$. On the other hand, $|C_n| \leq |H_n|$ since for each $c \in C$ there exists $h \in H_n$ with $c_n = h_n$. Thus $|C_n| \leq (2|X_n|)^{p(n)}$ holds and it means $\dim C_n = \log |C_n| \leq p(n) \log 2|X_n| \leq p(n)(n+2)$, which shows that C is of polynomial dimension.

(c) $\Rightarrow$ (a) Suppose that C is of polynomial dimension, i.e., there is a polynomial $d(n)$ such that $|C_n| \leq 2^{d(n)}$ holds for all $n \geq 0$. For each n, we define an equivalence relation $\overset{n}{\equiv}$ on C by $c \overset{n}{\equiv} c' \Leftrightarrow c_n = c'_n$ for c and c' in C. Since C is divided into at most $2^{d(n)}$ equivalence classes, we can associate each equivalence class with an identifier (n, i) with $1 \leq i \leq 2^{d(n)}$. These identifiers can be encoded with the samples of size $d(n) + \log n$ or less using two distinct examples of C. Let $A \in \mathcal{A}_C$ be a learn mapping which maps the code of (n, i) to a concept in the equivalence class corresponding to (n, i). For samples which are not the codes defined above, A is defined arbitrarily. Then we can easily see that A identifies any concept in C, i.e., C is learnable from selected examples by A.

(b) $\Leftrightarrow$ (c) By [7]. □

We observed that the learnability and learnability from selected examples are equivalent. It means in some sense that teachers are worthless. But there may be a possibility that a teacher can help a learner in saving time.

With polynomial-time restriction, we show that a gap exists between the polynomial-time learnability and the polynomial-time learnability from selected examples.

We assume that a concept class C is at most countably infinite. Further we assume a representation system for C as follows: An *index* of C is a function $I : C \to 2^{\Sigma^*}$ such that $c \neq c'$ implies $I(c) \cap I(c') = \emptyset$ for any c, $c' \in C$. A *representation* of a concept $c \in C$ is an element of $I(c)$. It should be noted that the learnability of a concept class depends on the representation system.

A *learning algorithm* for C is an algorithm which takes a sample of $c \in C$ as an input and outputs a representation of a concept $h \in C$, which is called a *hypothesis*. $\mathcal{L}_C$ denotes the set of all learning algorithms for C.

The following is the usual definition of polynomial-time learnability [1, 7].

Definition 4 Let C be a concept class and A be a learning algorithm for C. We say that a concept class C is *polynomial-time learnable* by A if the following conditions hold:

(3) A runs in polynomial time with respect to the length of the input.

(4) There exists a polynomial $p(\cdot, \cdot, \cdot)$ such that for any integer $n \geq 0$, any concept $c \in C$, any real numbers ε, δ $(0 < \varepsilon, \delta < 1)$, and any probability distribution P on X_n, if A takes a sample of size $p(n, \frac{1}{\varepsilon}, \frac{1}{\delta})$ which is generated randomly

and independently according to P, then A outputs a representation of a hypothesis h such that $P(c \oplus h) < \varepsilon$ with probability at least $1 - \delta$.

We say that C is *polynomial-time learnable* if there exists such learning algorithm $A \in \mathcal{L}_C$.

In a way analogous to Definition 2, we define as follows.

Definition 5 Let C be a concept class and A be a polynomial-time learning algorithm for C. We say that a concept class C is *polynomial-time learnable from selected examples* by A if the following conditions hold:

(5) A runs in polynomial time with respect to the input length.

(6) There exists a polynomial $p(n)$ such that for any integer $n \geq 0$ and any concept $c \in C$, there exists $\bar{x} \in X_n^{p(n)}$ such that if A takes $sam_c(\bar{x})$ then A always outputs a representation of a hypothesis h with $h_n = c_n$.

We say that C is *polynomial-time learnable from selected examples* if there exists such learning algorithm $A \in \mathcal{L}_C$.

Definition 6 For an integer $n \geq 0$, a bit vector $\vec{a} \in \{0,1\}^n$ and an integer k with $0 \leq k \leq \vec{a} \cdot \vec{a}$, an n-argument *threshold function* is defined by $Th_k(\vec{a}) = \{\vec{x} \in \{0,1\}^n \mid \vec{a} \cdot \vec{x} \geq k\}$, where $\vec{a} \cdot \vec{x}$ denotes the inner product of $\vec{a}$ and $\vec{x}$. THRESHOLD denotes the concept class of all threshold functions, i.e., THRESHOLD $= \{Th_k(\vec{a}) \mid \vec{a} \in \{0,1\}^n,\ 0 \leq k \leq \vec{a} \cdot \vec{a},\ n \geq 0\}$.

Proposition 2 If **RP≠NP**, there exists a concept class of polynomial dimension which is not polynomial-time learnable but polynomial-time learnable from selected examples.

Proof The concept class THRESHOLD is of polynomial dimension but not polynomial-time learnable if **RP≠NP** by [5, 8]. On the other hand, every n-argument threshold function $Th_k(\vec{a}) \in$ THRESHOLD can be expressed by the bit vector $\vec{a} \in \{0,1\}^n$ and the integer k $(0 \leq k \leq n)$. Here, we can encode the pair $(\vec{a}, k)$ with $n + \log n$ examples. An algorithm which decodes these samples in polynomial time will learn C from selected examples. Therefore C is polynomial-time learnable from selected examples. □

The above proposition claims that a teacher may improve the quality of information. The consideration on time required in learning is also important when a teacher is allowed.

3 Teachability

This section defines a notion of teachability. Intuitively, we say that a concept class is teachable if it can be taught to *all* learners who follow their teacher. This notion differs from the learnability from selected examples introduced in Section 2 on the point that it deals with plural learners.

A learn mapping $A \in \mathcal{A}_C$ for a concept class C is *consistent* if it satisfies the following condition (7):

(7) For all $c \in C$ and all $\bar{x} = (x_1, \ldots, x_m) \in X^m$ $(m \geq 1)$, $h(x_i) = c(x_i)$ holds for each $i = 1, \ldots, m$, where $h = A(sam_c(\bar{x}))$.

Namely, a consistent learn mapping produces a hypothesis consistent with a given sample. We denote by $\mathcal{A}_C^{\natural}$ the set of all consistent learn mappings for C.

Definition 7 A concept class C is *teachable by examples* if there exists a polynomial $p(n)$ satisfying the following condition (8):

(8) For any integer $n \geq 0$ and any concept $c \in C$, there exists $\bar{x} \in X_n^{p(n)}$ with $h_n = c_n$ for all $A \in \mathcal{A}_C^{\natural}$, where $h = A(sam_c(\bar{x}))$.

We show the relation between teachability and learnability.

Theorem 1

(a) If a concept class C is teachable by examples, then C is learnable.

(b) There is a concept class which is not teachable by examples but learnable.

Proof (a) Suppose that C is teachable by examples. It is obvious that there is a consistent learn mapping A_0. Then A_0 learns C from selected examples. By Proposition 1, C is learnable.

(b) For each pair of integers i and j $(i \geq 1,\ 0 \leq j < 2^i)$, $[i, j]$ denotes the string which encodes j using i bits in binary. For example, $[1, 0] = 0$, $[3, 0] = 000$, and $[3, 5] = 101$. λ denotes the null string. Now we consider the concept class $C = \{c^{00}\} \cup \{c^{ij} \mid i \geq 1,\ 0 \leq j < 2^i\}$, where $c^{00} = \{\lambda\}$ and $c^{ij} = \{\lambda,\ [i, j]\}$ for each $i \geq 1$ and $0 \leq j < 2^i$.

First, since $|C_n| = 2^{n+1} - 1$, we see that C is of polynomial dimension. Hence C is learnable by Proposition 1.

On the other hand, let n be an arbitrary fixed integer. Let us choose an integer m and $\bar{x} = (x_1, \ldots, x_m) \in X_n^m$ such that $A(sam_{c^{00}}(\bar{x}))_n = c_n^{00}$ for all $A \in \mathcal{A}_C^{\natural}$. We shall derive that $\bar{x}$ must contain all strings in $X_n - \{\lambda\}$. Suppose that there is a string $[i, j]$ with $1 \leq i \leq n$ and $0 \leq j < 2^i$ which does not appear in $\bar{x}$. Then we consider a learn mapping A_{ij} defined as follows:

$$A_{ij}(s) = \begin{cases} c^{kl} & \text{if } \langle [k, l], 1 \rangle \text{ appears in } s \text{ for some } k \geq 1 \text{ and } 0 \leq l < 2^k \\ c^{ij} & \text{else if } \langle [i, j], 0 \rangle \text{ does not appear in } s \\ c^{00} & \text{otherwise.} \end{cases}$$

Note that A_{ij} is well-defined since no two distinct positive examples $\langle [k,j],1\rangle$ and $\langle [k',j'],1\rangle$ appear in a sample $s \in S_C$. It is not difficult to see that A_{ij} is consistent, i.e., $A_{ij} \in \mathcal{A}_C^{\natural}$, because c^{ij} disagrees with c^{00} only on the point $[i,j]$. Since $i \leq n$, $A_{ij}(sam_{c^{00}}(\bar{x}))_n = c_n^{ij} = c^{ij} \neq c^{00} = c_n^{00}$ holds, which contradicts the assumption $A_{ij}(sam_{c^{00}}(\bar{x}))_n = c_n^{00}$. Therefore $m \geq |X_n - \{\lambda\}| = 2^{n+1} - 2$. Thus there is no polynomial which bounds m. Hence C is not teachable by examples. □

The above theorem claims that a learnable concept can not always be taught to all consistent learners correctly. On the other hand, it is known that if a concept class is learnable then every consistent learner can learn it [6, 7]. This suggests that for the concept class defined in the proof of (b) of Theorem 1, natural examples are more valuable than any polynomial number of examples selected by any teacher.

4 Burden of a Teacher

In the conventional computational learning theory, a teacher is defined as an oracle for a learner [10]. It is not required to consider computational complexity or even computability. But our concern is to study learning via teaching. In the section 2, we have already shown that the consideration on time required by *learner* is important in this setting. Thus we should take account of the complexity required by a *teacher*.

In this section, we regard the work of a teacher as choosing a small size sample called a *key*, which can express the target concept without any misunderstanding. We prove the following observations formally in the subsequent subsections.

- Teachability can be characterized by the size of a key.
- A concept which is easy to teach is not always easy to learn.
- It is intractable to make a key of minimum size in general. The work of a teacher is not easy.
- But approximate solutions can be found in polynomial time.

4.1 Key and teachability

Definition 8 Let S be a finite set and E be a family of subsets of S. Let $c \in E$. A subset T of S is a *key* to the set c in E if for all $c' \in E$ with $c \neq c'$, T contains some $x \in T$ with $c(x) \neq c'(x)$. We simply call T a key to c if E is clear from the context. The *size* of T is the number of elements in T.

Note that S is a trivial key to all $c \in E$. In other words, for each $c \in E$, there is at least one key to c. There may be some keys to c of different size. We can characterize the teachability of a concept class by the size of a key as follows.

Theorem 2 Let C be a concept class. Then the following (a) and (b) are equivalent:

(a) C is teachable by examples.

(b) There exists a polynomial $p(n)$ such that for each $n \geq 1$ and $c \in C$, there is a key of size $p(n)$ to c_n in the nth subclass C_n.

Proof Assume that C is teachable by examples and let $p(n)$ be a polynomial satisfying the condition (8) of Definition 7. Then for each concept $c_n \in C_n$, there exists $\bar{x} = (x_1, \ldots, x_{p(n)}) \in X_n^{p(n)}$ such that $A(sam_c(\bar{x}))_n = c_n$ for all $A \in A_C^{\natural}$. Now let T be the set $\{x_1, \ldots, x_{p(n)}\}$. It is not difficult to see that T is a key to c_n for C_n and $|T| \leq p(n)$ holds. The converse can also be shown similarly. □

The following figure shows the relations obtained so far.

Sample size		**Polynomial-time Restriction**
learnable from selected examples	···	polynomial-time learnable from selected examples
‖		∦ if **RP**≠**NP**
learnable	···	polynomial-time learnable
‖		‖
polynomial dimension	···	polynomial dimension and polynomial-time randomized hypothesis finder (see [1, 3])
∪∦		
teachable by examples		
‖		
polynomial size key		

4.2 Teachability and learnability

If we consider that the work of a teacher is to make a key of small size to the target concept, we can measure the complexity of the teacher by the complexity of finding such key. If a small size key can be constructed to each concept in short time, we regard that the concept class is easy to teach.

We show that MONOMIAL is a concept class which is both easy to teach and easy to learn. On the other hand, we also see that THRESHOLD can be taught easily but is not easy to learn. Thus a concept which is easy to teach is not always easy to learn.

It is known that THRESHOLD is not polynomial-time learnable if **RP** $\neq$ **NP** [5, 8].

Theorem 3 On the concept class THRESHOLD, a key of size at most $2n$ to an n-argument threshold function can be constructed in $O(n^2)$ time.

Proof Let $Th_k(\vec{a})$ be an n-argument threshold function. In the case of $k = 0$, it can be easily seen that $T = \{0^n\}$ is a key to $Th_0(\vec{a}) = \{0,1\}^n$.

Now we consider the case of $k \geq 1$. For the vector $\vec{a} = a_1 \cdots a_n$, let $J = \{i \mid a_i = 1\}$. We can assume $|J| \geq k$. For each $i = 1, \ldots, n$, let J_i be any subset of $J - \{i\}$ with $|J_i| = k - 1$ and let $\vec{u_i} = u_1 \cdots u_n \in \{0,1\}^n$ be the characteristic bit vector of $J_i \cup \{i\}$, i.e., $u_j = 1$ if $j \in J_i \cup \{i\}$ and $u_j = 0$ otherwise for $j = 1, \ldots, n$. For each $i = 1, \ldots, n$, let $\vec{v_i} = u_1 \cdots u_{i-1} \bar{a}_i u_{i+1} \cdots u_n$, where $\bar{a}_i = 1 \Leftrightarrow a_i = 0$. Let $U = \{\vec{u_i} \mid i \in J\}$, $V = \{\vec{v_i} \mid i = 1, \ldots, n\}$, and $T = U \cup V$. We show that T is a key to $Th_k(\vec{a})$. Note that $\vec{u_i} \in Th_k(\vec{a})$ holds for each $\vec{u_i} \in U$ since $\vec{u_i} \cdot \vec{a} = k$ and that $\vec{v_i} \notin Th_k(\vec{a})$ holds for each $\vec{v_i} \in V$ since $\vec{v_i} \cdot \vec{a} = k - 1$. Let $\vec{b}$ and l be a bit vector and an integer which satisfy both (i) $\vec{u_i} \in Th_l(\vec{b})$ for all $\vec{u_i} \in U$ and (ii) $\vec{v_i} \notin Th_l(\vec{b})$ for all $\vec{v_i} \in V$. We show that the above $\vec{b}$ and l are unique and the same as $\vec{a}$ and k, respectively. For an index i with $a_i = 1$, since $\vec{u_i}$ and $\vec{v_i}$ differ only at the ith bit, we see that $b_i = 1$ and $l = k$. For an index i with $a_i = 0$, we see $b_i = 0$ by (ii) Therefore T is a key to $Th_k(\vec{a})$ and its size is $|T| = |U| + |V| = \vec{a} \cdot \vec{a} + n \leq 2n$. It is not difficult to see that T can be constructed from given $\vec{a}$ and k in $O(n^2)$ time. □

Definition 9 For an integer $n \geq 0$ and a bit vector $\vec{a} \in \{0,1\}^n$, an n-argument *monotone monomial* is defined by $mono(\vec{a}) = \{\vec{x} \in \{0,1\}^n \mid \vec{a} \cdot \vec{x} = \vec{a} \cdot \vec{a}\}$. MONOMIAL denotes the concept class of monotone monomials, i.e., MONOMIAL $= \{mono(\vec{a}) \mid \vec{a} \in \{0,1\}^n,\ n \geq 0\}$.

Viewed in another way, MONOMIAL is the subclass of THRESHOLD since MONOMIAL $= \{Th_k(\vec{a}) \mid \vec{a} \in \{0,1\}^n,\ k = \vec{a} \cdot \vec{a},\ n \geq 0\}$.

MONOMIAL is known to be polynomial-time learnable [10]. We also obtain the following result.

Corollary 1 On the concept class MONOMIAL, a key of size at most n to an n-argument monotone monomial can be constructed in $O(n^2)$ time.

4.3 Making a minimum size key is intractable

Definition 10 The *minimum key problem* (MKP) is defined as follows:

Instance: A collection E of subsets of a finite set S, a set $c^* \in E$, and a positive integer $q \leq |S|$.

Question: Is there a key to c^* of size q or less?

Theorem 4 The minimum key problem is **NP**-complete.

Proof We use an **NP**-complete problem called the hitting set problem [2]: Given a collection E of subsets of a finite set S and a positive integer $q \leq |S|$, decide whether there is a subset T with $|T| \leq q$ such that T contains at least one element from each subset in E. MKP is **NP**-hard since the hitting set problem is a special case with $c^* = \emptyset$. Obviously, MKP is in **NP**. □

4.4 Approximation algorithm for the minimum key problem

Theorem 5 Let E be a collection of subsets of a finite set S. A key to each $c^* \in E$ with size $k(c^*)(\log_e |E|+1)$ or less can be computed in time $O(|S|^2|E|)$, where $k(c^*)$ is the minimum size of a key to c^*.

Proof First we convert an instance of MKP to that of the minimum set cover problem which is stated as follows: Given a collection U of subsets of a finite set Z and a positive integer $l \leq |U|$, decide whether U contains a set cover for Z of size l or less, i.e., a subset $U' \subseteq U$ with $|U'| \leq l$ such that every element of Z belongs to at least one member of U'.

Let $S = \{x_1, \ldots, x_n\}$ and $E = \{c^*, c_1, \ldots, c_m\} \subseteq 2^S$ constitute an instance of MKP, where $c^*, c_1, \ldots, c_m$ are mutually distinct. Then let $Z = \{z_1, \ldots, z_m\}$, where $z_1, \ldots, z_m$ are m distinct elements. For $i = 1, \ldots, n$, we define $u_i = \{z_j \in Z \mid c^*(x_i) \neq c_j(x_i)\}$. Let $U = \{u_1, \ldots, u_n\}$. Then U can be computed in $O(mn)$ time. For a subset U' of U, let $T_{U'} = \{x_i \in S \mid u_i \in U'\}$. It is not difficult to see that U' is a set cover for Z if and only if $T_{U'}$ is a key to c^*.

Johnson [4] devised an $O(|U|^2|Z|)$ time greedy algorithm that produces a set cover of size at most $K(\log_e M + 1)$, where K is the minimum number of sets from U needed for covering Z and $M = \max\{|u_i| \mid u_i \in U\}$. With this algorithm, we can find a key of size at most $k(c^*)(\log_e |E| + 1)$ in $O(|S|^2|E|)$ time since $M \leq |E|$. □

5 Conclusion

Teaching and learning may be closely but delicately related. It is said that:

> "To teach is to learn."
> "Learning is one thing, and teaching is quite another."
> "There is nothing so good for learning as teaching."

These words involve some truths though they seem inconsistent one another. This paper tried to capture such truths in a mathematical framework.

References

[1] Blumer, A., Ehrenfeucht, A., Haussler, D., and Warmuth, M.K., "Learnability and the Vapnik-Chervonenkis dimension", Journal of the ACM, Vol. 36, No. 4, pp. 929–965, 1989.

[2] Garey, M.R. and Johnson, D.S., Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman, 1979.

[3] Haussler, D., Kearns, M., Littlestone, N., and Warmuth, M.K., "Equivalence of models for polynomial learnability", in Proceedings of the 1st Workshop on Computational Learning Theory, Morgan Kaufmann, pp. 42–55, 1988.

[4] Johnson, D.S., "Approximation algorithms for combinatorial problems", Journal of Computer and System Sciences, Vol. 9, pp. 256–278, 1974.

[5] Kearns, M., Li, M., Pitt, L., and Valiant, L., "On the learnability of boolean formulae", in Proceedings of the 19th Annual ACM Symposium on Theory of Computing, pp. 285–295, 1987.

[6] Natarajan, B.K., "On learning boolean functions", in Proceedings of the 19th Annual ACM Symposium on Theory of Computing, pp. 296–304, 1987.

[7] Natarajan, B.K., "On learning sets and functions", Machine Learning, Vol. 4, No. 1, pp. 67–97, 1989.

[8] Pitt, L. and Valiant, L.G., "Computational limitations on learning from examples", Journal of the ACM, Vol. 35, No. 4, pp. 965–984, 1988.

[9] Shinohara, A. and Miyano, S., "A foundation of algorithmic teaching", Technical Report RIFIS-TR-CS-22, Kyushu University, December, 1989.

[10] Valiant, L.G., "A theory of the learnable", Communications of the ACM, Vol. 27, No. 11, pp. 1134–1142, 1984.