Finding rules in data

Tu-Bao Ho[1,2]

[1] Institute of Information Technology, Vietnamese Academy of Science and Technology, 18 Hoang Quoc Viet, Hanoi, Vietnam
[2] Japan Advanced Institute of Science and Technology (JAIST)
1-1 Asahidai, Nomi, Ishikawa, Japan *bao@jaist.ac.jp*

Abstract. In the first year of my preparation for doctor thesis at INRIA in the group of Edwin, I worked on the construction of an inference engine and a knowledge base, by consulting various group members, for building an expert system guiding the data analysis package SICLA of the group. One day, Edwin asked me whether one can automatically generate rules for expert systems from data, and I started my new research direction. Since that time, my main work has been machine learning, especially finding rules in data. This paper briefly presents some learning methods we have developed.

1 Introduction

Twenty years ago, machine learning was in its infancy with few work and applications. The wave of artificial intelligence (AI) in early of years 1980s has fostered the development of machine learning. From the joined work on conceptual clustering with Michalski, Edwin found his interest in this young field of machine learning (Michalski et al., 1983). As a doctor candidate in his group at that time, he suggested if I can work on finding new ways to generate rules for expert systems from data, instead of working as knowledge engineers who try to acquire knowledge from human experts.

There have been a great progress in the field of machine learning. It has become an established area with sound foundation, rich techniques and various applications. Machine learning becomes one of the most active areas in computer science. This paper briefly presents some of our main work in machine learning since those days in the group of Edwin, from supervised learning (Ho et al., 1988), (Nguyen and Ho, 1999), (Ho and Nguyen, 2003) to unsupervised learning (Ho, 1997), (Ho and Luong, 1997), and some recent work on text clustering (Ho and Nguyen, 2002), (Ho et al., 2002), (Le and Ho, 2005), bioinformatics (Pham and Ho, 2007), kernel methods (Nguyen and Ho, 2007).

2 Rule induction from supervised data

In this section we briefly present three supervised learning methods of CABRO1 (Ho et al., 1988), CABRO2 (Nguyen and Ho, 1999), and LUPC (Ho and Nguyen, 2003).

2.1 CABRO1

CABRO1 (Construction Automatique a Base de Regles a partir d'Observation) is a method of rule induction from supervised data.

Let $D_1, D_2, ..., D_p$ be p finite domains and $D_1 \times D_2 \times ... \times D_p$. Elements of D are called *objects* and denoted by $\omega = (d_1, d_2, ..., d_p)$ where $d_j \in D_j$ for $j \in J = \{1, 2, ..., p\}$. A variable-value pair (X_j, d_j) defines an *elementary assertion* $A_{X_j=d_j}$ that determines the set $\omega_{X_j=d_j}$ of objects of D which have the value $d_j \in \{d_{j1}, d_{j2}, ..., d_{jq}\}$ for the variable X_j: $A_{X_j=d_j} : D \longrightarrow \{true, false\}$, $\omega = (d_1, ..., d_p) \mapsto A_{X_j=d_j}(\omega) = true,\ if\ X_j(\omega) = d_j$ and $\omega = (d_1, ..., d_p) \mapsto A_{X_j=d_j}(\omega) = false,\ if\ X_j(\omega) \neq d_j$.

We consider an *assertion* as a conjunction of elementary assertions: $A = \bigwedge(X_j, d_j),\ \ j \in J' \subseteq J\ \ and\ \ d_j \in D_j$, where $\bigwedge$ denotes the logical conjunction. An assertion A is a Boolean function from $D \longrightarrow \{true, false\}$, and it is also the identification function for the set: $\omega_A = \{\omega \in D \mid A(\omega) = true\}$.

Variables correspond to $j \in J'$ are said to be *tied to* the assertion. Variables correspond to $j \in J \setminus J'$ are said to be *free from* the assertion. Number of tied variables is called *length* of the assertion. One says also that assertion A *covers* the set ω_A. Assertion A is said to be *more general than* assertion B iff $\omega_B \subseteq \omega_A$. Assertion A is said to be *better than* assertion B iff $card(\omega_A) > card(\omega_B)$. A is a *representative assertion* generated from an object $\omega \in E$ if A is one of the best assertions formed by elementary assertions generated from ω.

Denote $\Re = \Re_C \cup \Re'_C$ the set of assertions to be found for C and C'. Naturally, assertions generated for each concept, for instance C, have to satisfy two following constraints: (1) *Covering*: Each observed object of the learning set E has to be recognized by an assertion of $\Re_C$: $E \subseteq \bigcup_{A \in \Re_C} \omega_A$, and (2) *Discriminating*: Assertions of C do not misrecognize members of $E' : \omega_A \cap E' = \emptyset$.

It is clear that the less general an assertion, the more discriminant it is. Depending on the data nature, one retains general but not perfect discriminant assertions or discriminant but not sufficient general assertions. The *belief measure* $\mu(A)$ for the assertion A of C is estimated as the ratio of the number of examples of C matched by A and the total number of examples of C and C' matched by A: $\mu(A) = card(\omega_A \cap E)/card(\omega_A \cap \Omega), (0 < \mu(A) \leq 1)$.

An assertion A is said *β-discriminant* if $\mu(A) \geq \beta$. In fact, instead of finding discriminant assertions one finds β-discriminant assertions depending on an acceptance threshold $\beta\ \ (0 < \beta \leq 1)$.

The main algorithm of CABRO1 is based on a *general-to-specific search*: one starts from an 'empty' assertion which is the 'most general' because all of its variables are free, then one ties the value $X_j(\omega)$ to this assertion so that the assertion covers approximately a maximum number of objects of E (the generality of the assertion will be diminished but it may remain non β- discriminant). This phase is repeated with the remaining values until one finds a β-discriminant assertion such that the next attempt does not improve the covering of the assertion. We propose a dual algorithm of the CABRO1

algorithm, based on a *specific-to-general* search strategy, in order to find a representative assertion A_ω from an object $\omega \in E$. On the contrary with CABRO1 algorithm, the dual algorithm starts from a 'full' assertion which is the 'most specific' and covers only the object ω. One tries to increase its generality and to diminish its speciality simultaneously in order to obtain a representative assertion.

2.2 CABRO2

The starting point of rough set theory (Pawlak, 1991) is the assumption that our "view" on elements of an object set O depends on an indiscernibility relation among them, that means an equivalence relation $E \subseteq O \times O$. Two objects $o_1, o_2 \in O$ are said to be *indiscernible* w.r.t E if $o_1 E o_2$. The *lower* and *upper* approximations of any $X \subseteq O$, w.r.t. an equivalence relation E, are defined as

$$E_*(X) = \{o \in O : [o]_E \subseteq X\}, \quad E^*(X) = \{o \in O : [o]_E \cap X \neq \emptyset\}$$

where $[o]_E$ denotes the equivalence class of objects which are indiscernible with o w.r.t the equivalence relation E. A subset P of the set of attributes used to describe objects of O determines an equivalence relation that divides O into equivalence classes each containing objects having the same values on all attributes of P. A key concept in the rough set theory is the *degree of dependency* of a set of attributes Q on a set of attributes P, denoted by $\mu_P(Q)$ $(0 \leq \mu_P(Q) \leq 1)$, defined as $\mu_P(Q) = |\bigcup_{[o]_Q} / P_*([o]_Q)|/|O|$.

If $\mu_P(Q) = 1$ then Q totally depends on P; if $0 < \mu_P(Q) < 1$ then Q partially depends on P; if $\mu_P(Q) = 0$ then Q is independent of P. The measure of dependency is fundamental in rough set theory as based on it important notions are defined, such as reducts and minimal sets of attributes, significance of attributes, etc.

This argument can be generalized and formulated for a measure of degree of dependency of an attribute set Q on an attribute set P

$$\mu'_P(Q) = \frac{1}{|O|} \sum_{[o]_P} max_{[o]_Q} |[o]_Q \bigcap [o]_P|$$

Theorem. *For every sets P and Q of attributes we have*

$$max_{[o]_Q} |[o]_Q|/|O| \leq \mu'_P(Q) \leq 1$$

We can define that Q totally depends on P iff $\mu'_P(Q) = 1$; Q partially depends on P iff $max_{[o]_Q} |[o]_Q|/|O| < \mu'_P(Q) < 1$; Q is independent of P iff $\mu'_P(Q) = max_{[o]_Q} |[o]_Q|/|O|$.

Given two arbitrary attribute sets P and Q, we define *R-measure* for the dependency of Q on P

$$\mu_P(Q) = \frac{1}{|O|} \sum_{[o]_P} max_{[o]_Q} \frac{|[o]_Q \bigcap [o]_P|^2}{|[o]_P|}$$

Learn-Positive-Rule$(Pos, Neg, mina, minc)$

1. $RuleSet = \phi$
2. $\alpha, \beta \leftarrow$ **Initialize**$(Pos, mina, minc)$
3. while $(Pos \neq \phi$ & $(\alpha, \beta) \neq (mina, minc))$
4. $\quad NewRule \leftarrow$ **BestRule**$(Pos, Neg, \alpha, \beta)$
5. $\quad$ if $(NewRule \neq \phi)$
6. $\quad\quad Pos \leftarrow Pos \setminus Cover^{+}(NewRule)$
7. $\quad\quad RuleSet \leftarrow RuleSet \cup NewRule$
8. $\quad$ else **Reduce**(α, β)
9. $RuleSet \leftarrow$ **PostProcess**$(RuleSet)$
10. return$(RuleSet)$

BestRule$(Pos, Neg, \alpha, \beta)$

11. $CandRuleset = \phi$
12. **AttValPairs**$(Pos, Neg, \alpha, \beta)$
13. while **StopCond**$(Pos, Neg, \alpha, \beta)$
14. $\quad$ **CandRules**$(Pos, Neg, \alpha, \beta)$
15. $BestRule \leftarrow$ First $CandidateRule$ in $CandRuleset$
16. return$(BestRule)$

Fig. 1. The scheme of algorithm LUPC

When consider Q as the class attribute and P a descriptive attribute, we can use $\mu_P(Q)$ as a measure for attribute selection in decision tree learning. CABRO2 is the decision tree induction using *R-measure* that has performance as high as state-of-the-art methods such C4.5 (Nguyen and Ho, 1999).

2.3 LUPC

LUPC (Learning Unbalanced Positive Class) is a separate-and-conquer rule induction method to *learn minority classes in unbalanced datasets*. LUPC consequently learns a rule set from Pos and Neg given user-specified minimum accuracy threshold ($mina$) and minimum cover ratio ($minc$). We can partially order the goodness of rules in terms of accuracy or support. Given two thresholds α and β, $0 \leq \alpha, \beta \leq 1$, on accuracy and support of rules, respectively. A rule R is $\alpha\beta$-strong if $acc(R) \geq \alpha$ and $sup(R) \geq \beta$. An $\alpha\beta$-strong rule R_i is said better than an $\alpha\beta$-strong rule R_j with respect to α if R_i has accuracy higher than that of R_j. An $\alpha\beta$-strong rule R_i is better than an $\alpha\beta$-strong rule R_j with respect to β if R_i has support higher than that of R_j. LUPC distinguishes three alternatives that occur in practice and that lead to the three corresponding types of search heuristics:

1. *Bias on rule accuracy*: It is to sequentially find rules with cover ratio equal and greater than $minc$ but accuracy is as large as possible.
2. *Bias on rule cover ratio*. It is to sequentially find rules with accuracy equal and greater than $mina$ but the cover ratio is as large as possible.
3. *Alternative bias on rule cover ratio and accuracy*. LUPC starts with highest values of α and β, and alternatively learns rules with bias on either accuracy or cover ratio, then reduces one of the corresponding α or β

while keeping the other. The search is done until reaching the stopping. condition.

Note that $cov^+(R)$ can be quickly determined because $|Pos| \ll |Neg|$. When searching for $\alpha\beta$-strong rules, a candidate rule will be eliminated without continuing to scan though large set Neg if this property holds during scanning.

Proposition 1. *Given a threshold α, a rule R is not $\alpha\beta$-strong for any arbitrary β if $cov^-(R) \geq ((1-\alpha)/\alpha) \times cov^+(R)$.*

Figure 1 presents the scheme of algorithm LUPC that consists of two main procedures *Learn-Positive-Rule* and *BestRule* (Ho and Nguyen, 2003). LUPC has been applied to study stomach cancer and hepatitis with successes.

3 Conceptual clustering

A theory of concept lattices has been studied under the name *formal concept analysis* (FCA) (Wille, 1982). Considers a *context* as a triple $(\mathcal{O}, \mathcal{D}, \mathcal{R})$ where $\mathcal{O}$ be a set of objects, $\mathcal{D}$ be a set of primitive descriptors and $\mathcal{R}$ be a binary relation between $\mathcal{O}$ and $\mathcal{D}$, i.e., $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{D}$ and $(o, d) \in \mathcal{R}$ is understood as the fact that object o has the descriptor d. For any object subset $X \subseteq \mathcal{O}$, the largest tuple common to all objects in X is denoted by $\lambda(X)$. For any tuple $S \in \mathcal{T}$, the set of all objects satisfying S is denoted by $\rho(S)$. A tuple S is *closed* if $\lambda(\rho(S)) = S$. Formally, a *concept* C in the classical view is a pair (X, S), $X \subseteq \mathcal{O}$ and $S \subseteq \mathcal{T}$, satisfying $\rho(S) = X$ and $\lambda(X) = S$. X and S are called *extent* and *intent* of C, respectively. Concept (X_2, S_2) is a *subconcept* of concept (X_1, S_1) if $X_2 \subseteq X_1$ which is equivalent to $S_2 \supseteq S_1$, and (X_1, S_1) is then a *superconcept* of (X_2, S_2).

It was shown that λ and ρ define a Galois connection between the power sets $\wp(\mathcal{O})$ and $\wp(\mathcal{D})$, i.e., they are two order-reversing one-to-one operators. As a consequence, the following properties hold which will be exploited in the learning process:

$$\begin{gathered}
\text{if} \quad S_1 \subseteq S_2 \quad \text{then} \quad \rho(S_1) \supseteq \rho(S_2) \quad \text{and} \quad \lambda\rho(S_1) \subseteq \lambda\rho(S_2) \\
\text{if} \quad X_1 \subseteq X_2 \quad \text{then} \quad \lambda(X_1) \supseteq \lambda(X_2) \quad \text{and} \quad \rho\lambda(X_1) \subseteq \rho\lambda(X_2) \\
S \subseteq \lambda\rho(S), \quad X \subseteq \rho\lambda(X) \\
\rho\lambda\rho = \rho, \quad \lambda\rho\lambda = \lambda, \quad \lambda\rho(\lambda\rho(S)) = \lambda\rho(S) \\
\rho(\textstyle\bigcup_j S_j) = \bigcap_j \rho(S_j), \quad \lambda(\bigcup_j X_j) = \bigcap_j \lambda(X_j)
\end{gathered}$$

The basic theorem in FCA states that the set of all possible concepts from a context $(\mathcal{O}, \mathcal{D}, \mathcal{R})$ is a *complete lattice*[1] $\mathcal{L}$, called Galois lattice, in which infimum and supremum can be described as follows:

$$\bigwedge_{t \in T} (X_t, S_t) = (\bigcap_{t \in T} X_t, \lambda\rho(\bigcup_{t \in T} S_t))$$

[1] A lattice $\mathcal{L}$ is complete when each of its subsetf X has a least upper bound and a greatest lower bound in $\mathcal{L}$.

Table 1. Scheme of OSHAM conceptual clustering

Input	concept hierarchy H and an existing splittable concept C_k.
Result	H formed gradually.
Top-level	call OSHAM(root concept, $\emptyset$).

1. While C_k is still splittable, find a new subconcept of it that corresponds to the hypothesis minimizing the quality function $q(C_k)$ among η hypotheses generated by the following steps
   (a) Find a "good" attribute-value pair concerning the best cover of C_k.
   (b) Find a closed attribute-value subset S containing this attribute-value pair.
   (c) Form a subconcept C_{k_i} with the intent is S.
   (d) Evaluate the quality function with the new hypothesized subconcept.
   Form intersecting concepts corresponding to intersections of the extent of the new concept with the extent of existing concepts excluding its superconcepts.
2. If one of the following conditions holds then C_k is considered as unsplittable
   (a) There exist not any closed proper feature subset.
   (b) The local instances set C_k^r is too small.
   (c) The local instances set C_k^r is homogeneous enough.
3. Apply recursively the procedure to concepts generated in step 1.

$$\bigvee_{t\in T}(X_t, S_t) = (\rho\lambda(\bigcup_{t\in T} X_t), \bigcap_{t\in T} S_t)$$

OSHAM (Making Automatically a Hierarchy of Structured Objects) is our proposed conceptual clustering method (Ho, 1997), (Ho and Luong, 1997). OSHAM allow generating descriptive rules from symbolic unsupervised datasets.

4 Tolerance rough set model and applications

The *tolerance rough set model* (TRSM) aims to enrich the document representation in terms of semantics relatedness by creating tolerance classes of terms in $\mathcal{T}$ and approximations of subsets of documents. The model has the root from rough set models and its extensions. The key idea is among three properties of an equivalence relation R in an universe U used in the original rough set model (reflexive: xRx; symmetric: $xRy \rightarrow yRx$; transitive: $xRy \wedge yRz \rightarrow xRz$ for $\forall x, y, z \in U$), the transitive property does not always hold in natural language processing, information retrieval, and consequently text data mining. In fact, words are better viewed as overlapping classes which can be generated by *tolerance relations* (requiring only reflexive and symmetric properties).

The key issue in formulating a TRSM to represent documents is the identification of tolerance classes of index terms. We employ the co-occurrence

Table 2. The TRSM nonhierarchical clustering algorithm

Input	The set $\mathcal{D}$ of documents and the number K of clusters.
Result	K clusters of $\mathcal{D}$ associated with cluster membership of each document.

1. Determine the initial representatives $R_1, R_2, ..., R_K$ of clusters $C_1, C_2, ..., C_K$ as K randomly selected documents in $\mathcal{D}$.
2. For each $d_j \in \mathcal{D}$, calculate the similarity $S(\mathcal{U}(\mathcal{R}, d_j), R_k)$ between its upper approximation $\mathcal{U}(\mathcal{R}, d_j)$ and the cluster representative $R_k, k = 1, ..., K$. If this similarity is greater than a given threshold, assign d_j to C_k and take this similarity value as the cluster membership $m(d_j)$ of d_j in C_k.
3. For each cluster C_k, re-determine its representative R_k.
4. Repeat steps 2 and 3 until there is little or no change in cluster membership during a pass through $\mathcal{D}$.
5. Denote by d_u an unclassified document after steps 2, 3, 4 and by $\mathrm{NN}(d_u)$ its nearest neighbor document (with non-zero similarity) in formed clusters. Assign d_u into the cluster that contains $\mathrm{NN}(d_u)$, and determine the cluster membership of d_u in this cluster as the product $m(d_u) = m(\mathrm{NN}(d_u)) \times S(\mathcal{U}(\mathcal{R}, d_u), \mathcal{U}(\mathcal{R}, \mathrm{NN}(d_u)))$. Re-determine the representatives R_k, for $k = 1, ..., K$.

of index terms in all documents from $\mathcal{D}$ to determine a tolerance relation and tolerance classes. Denote by $f_{\mathcal{D}}(t_i, t_j)$ the number of documents in $\mathcal{D}$ in which two index terms t_i and t_j co-occur. We define an uncertainty function I depending on a threshold θ as $I_\theta(t_i) = \{t_j \mid f_{\mathcal{D}}(t_i, t_j) \geq \theta\} \cup \{t_i\}$.

It is clear that the function I_θ defined above satisfies the condition of $t_i \in I_\theta(t_i)$ and $t_j \in I_\theta(t_i)$ iff $t_i \in I_\theta(t_j)$ for any $t_i, t_j \in \mathcal{T}$, and so I_θ is both reflexive and symmetric. This function corresponds to a tolerance relation $\mathcal{I} \subseteq \mathcal{T} \times \mathcal{T}$ that $t_i \mathcal{I} t_j$ iff $t_j \in I_\theta(t_i)$, and $I_\theta(t_i)$ is the tolerance class of index term t_i. A vague inclusion function ν, which determines how much X is included in Y, is defined as $\nu(X, Y) = |X \cap Y|/|X|$

This function is clearly monotonous with respect to the second argument. Using this function the membership function, a similar notion as that in fuzzy sets, μ for $t_i \in \mathcal{T}, X \subseteq \mathcal{T}$ can be defined as $\mu(t_i, X) = \nu(I_\theta(t_i), X) = |I_\theta(t_i) \cap X|/|I_\theta(t_i)|$

With these definitions we can define a tolerance space as $\mathcal{R} = (\mathcal{T}, I, \nu, P)$ in which the *lower approximation* $\mathcal{L}(\mathcal{R}, X)$ and the *upper approximation* $\mathcal{U}(\mathcal{R}, X)$ in $\mathcal{R}$ of any subset $X \subseteq \mathcal{T}$ can be defined as

$$\mathcal{L}(\mathcal{R}, X) = \{t_i \in \mathcal{T} \mid \nu(I_\theta(t_i), X) = 1\}$$

$$\mathcal{U}(\mathcal{R}, X) = \{t_i \in \mathcal{T} \mid \nu(I_\theta(t_i), X) > 0\}$$

The term-weighting method is extended to define weights for terms in the upper approximation $\mathcal{U}(\mathcal{R}, d_j)$ of d_j. It ensures that each term in the upper approximation of d_j but not in d_j has a weight smaller than the weight of

Table 3. TRSM-based hierarchical agglomerative clustering algorithm

---

Input A collection of M documents $\mathcal{D} = \{d_1, d_2, \ldots, d_M\}$
Result Hierarchical structure of $\mathcal{D}$

Given: a collection of M documents $\mathcal{D} = \{d_1, d_2, \ldots, d_M\}$
a similarity measure $sim : \mathcal{P}(\mathcal{D}) \times \mathcal{P}(\mathcal{D}) \to R$
for $j = 1$ **to** M **do**
$C_j = \{d_j\}$ **end**
$H = \{C_1, C_2, , \ldots, C_M\}$
$i = M + 1$
while $|H| > 1$
$(C_{n_1}, C_{n_2}) = \text{argmax}_{(C_u, C_v) \in H \times H} sim(\mathcal{U}(\mathcal{R}, C_u), \mathcal{U}(\mathcal{R}, C_v)$
$C_i = C_{n_1} \cup C_{n_2}$
$H = (H \setminus \{C_{n_1}, C_{n_2}\}) \cup \{C_i\}$
$i = i + 1$

---

any term in d_j.

$$w_{ij} = \begin{cases} (1 + \log(f_{d_j}(t_i))) \times \log \frac{M}{f_{\mathcal{D}}(t_i)} & \text{if } t_i \in d_j, \\ \min_{t_h \in d_j} w_{hj} \times \frac{\log(M/f_{\mathcal{D}}(t_i))}{1+\log(M/f_{\mathcal{D}}(t_i))} & \text{if } t_i \in \mathcal{U}(\mathcal{R}, d_j) \setminus d_j \\ 0 & \text{if } t_i \notin \mathcal{U}(\mathcal{R}, d_j) \end{cases}$$

The vector length normalization is then applied to the upper approximation $\mathcal{U}(\mathcal{R}, d_j)$ of d_j. Note that the normalization is done when considering a given set of index terms. Denote the document set by $\mathcal{D} = \{d_1, d_2, \ldots, d_M\}$ where $d_j = (t_{1j}, w_{1j}; t_{2j}, w_{2j}; \ldots; t_{rj}, w_{rj})$ and $w_{ij} \in [0, 1]$. The set of all terms from $\mathcal{D}$ is denoted by $\mathcal{T} = \{t_1, t_2, \ldots, t_N\}$. In information retrieval, a query is given the form $Q = (q_1, w_{1q}; q_2, w_{2q}; \ldots; q_s, w_{sq})$ where $q_i \in \mathcal{T}$ and $w_{iq} \in [0, 1]$.

Table 2 and Table 3 describe two general TRSM-based nonhierarchical and hierarchical clustering algorithms. The TRSM-based nonhierarchical clustering algorithm can be considered as a reallocation clustering method to form K clusters of a collection $\mathcal{D}$ of M documents. The main point of the TRSM-based hierarchical clustering algorithm is at each merging step it uses upper approximations of documents in finding two closest clusters to merge.

In (Ho et al., 2002), we have applied TRSM and TRSM-based clustering algorithms to information retrieval and text analysis tasks. Interestingly, the TRSM cluster-based retrieval achieved higher recall than that of full retrieval in our experiments, especially the TRSM cluster-based retrieval usually offers precision higher than that of full retrieval in most experiments, and achieves recall and precision nearly as that of full search just after searching on one or two clusters.

5 Acknowledgments

I would like to express my deep thanks to Edwin Diday. His insight has been invaluable to me. I also wish to thanks people in his group for our long cooperation, especially Patrice Bertrand, Marc Csernel, Guy Cucumel, Eric Demonchaux, Rolan Ducournau, Meireille Gettler-Summa, Yves Lechevallier, Joel Quinqueton, Henri Ralambondrainy, Myriam Touati.

References

HO, T.B., QUINQUETON, J., RALAMBONDRAINY, H., (1986): Using Expert System Techniques for Interpretation of Data Analysis Results. In: F. De Antoni, N. Laura, A. Rizzi (Eds.): *Proceedings of COMPSTAT'86*. Physica-Verlag Heidelberg Wien, 308–311.

HO, T.B. (1987): On the Design and Implementation of an Expert System Using the Inference Engine COTO. *Computers and Artificial Intelligence 6 (4), 297-310.*

HO, T.B., DIDAY, E., GETTLER-SUMMA, M. (1988): Generating Rules for Expert Systems from Observations. *Pattern Recognition Letters 7 (5), 265–271.*

HO, T.B. (1990): General-to-Specific and Specific-to-General Algorithms in the CABRO Concept Learning Method. *Proceedings of 1st Pacific Rim International Conference on Artificial Intelligence PRICAI'90*, 619–624.

HO, T.B. (1997): Discovering and Using Knowledge From Unsupervised Data. *Decision Support Systems 21(1), 27–41.*

HO, T.B., LUONG, C.M. (1997): Using Case-Based Reasoning in Interpreting Unsupervised Inductive Learning Results. *Proceedings of International Joint Conference on Artificial Intelligence IJCAI'97.* Morgan Kaufmann, 258–263.

NGUYEN, T.D., HO, T.B. (1999): An Interactive-Graphic System for Decision Tree Induction. *Journal of Japanese Society for Artificial Intelligence 14(1), 131–138.*

HO, T.B., NGUYEN, N.B. (2002): Document Clustering by Tolerance Rough Set Model. *International Journal of Intelligent Systems 17(2), 131–138.*

HO, T.B., KAWASAKI, S., NGUYEN, N.B. (2002): Cluster-based Information Retrieval with a Tolerance Rough Set Model. *International Journal of Fuzzy Logic and Intelligent Systems 2(1), 26–32.*

HO, T.B., NGUYEN, D.D. (2003): Learning Minority Classes and Chance Discovery. *Journal New Generation Computing 21(2), 149–161.*

LE, S.Q., HO, T.B. (2005): An Association-based Dissimilarity Measure for Categorical Data. *Pattern Recognition Letters 26(6), 2549-2557.*

NGUYEN, C.H., HO, T.B. (2007): Kernel Matrix Evaluation. *Twentieth International Joint Conference on Artificial Intelligence IJCAI'07* (in press).

MICHALSKI, R., STEPP, R., DIDAY, E. (1983): Clustering objects into classes characterized by conjuctive concepts. In *Progress in Pattern Recognition, volume 1.* North Holland.

PAWLAK, Z. (1991): *Rough sets: Theoretical Aspects of Reasoning About Data*, Kluwer Academic Publishers.

PHAM, T.H., HO, T.B. (2007): A hyper-heuristic for descriptive rule induction. *International Journal of Data Warehousing and Mining 3(1), 54–66.*

WILLE, R. (1982): Restructuring lattice theory: An approach based on hierarchies of concepts, Rival, I. (Ed.) *Ordered Sets*, 445–470.