Clustering and Refinement of Hierarchical Concept from Categorical Databases Based on Rough Sets

Been-Chian Chien and Su-Yu Liao
Department of Information Engineering
I-Shou University
Kaohsiung 840, Taiwan, R.O.C
Email: cbc@isu.edu.tw

Abstract— Discovering knowledge from large databases is a challenge in many applications. The implicit meanings of knowledge can be repressed by different knowledge representations. A concept hierarchy is a concise and general form of knowledge representation. Hierarchical concept description can organize relationships of data and express knowledge embedded in databases explicitly. In this paper, we propose a new scheme based on rough sets to cluster and refine the concept hierarchy automatically for a given data set with nominal attributes. The proposed scheme consists of two algorithms: the concept clustering algorithm and the concept refinement algorithm. The experimental results show that the concept hierarchy mined by the proposed scheme contains meaningful concept in comparison with the previous approaches. The analyses of the algorithms also show that the proposed scheme is efficient and scaleable for large databases. It is also able to be extended to mining meaningful rules from databases.

I. INTRODUCTION

Knowledge discovery in databases (KDD) is one of the important research topics in both areas of machine learning and databases. The objective of KDD is to extract knowledge automatically from huge amounts of data. The knowledge mined from large practical databases can be applied to many applications. For different applications, the different types of knowledge lead to various methodologies in the development of data mining techniques. For example, classification learning algorithms for decision trees construction [15], the Apriori algorithm for mining association rules [1], and unsupervised learning algorithms for generating clusters [9]. Regardless of the mining approaches involved, we can imagine the various types of discovered knowledge as the concept in the database based on different viewpoints.

For representing concepts explicitly, a meaningful and clear concept description is required. Concept hierarchies are a concise and general form of concept description widely used. Many knowledge representations are variants of concept hierarchy in different presentations. For transaction databases, as an example, association rules are the set of rules describing the relative relationships in a partial ordering hierarchy of large item sets. Decision tree is another type of concept hierarchy used for generating classification rules and classifying the categories of data. The other type of concept hierarchy, like clusters, is a one-level concept grouping data with high similarity into classes.

A concept hierarchy defines a sequence of mappings from a set of low-level concepts to higher-level, more general concepts on one or a set of attribute domains. Such mappings may organize the set of concept in partial order according to a general-to-specific ordering and form a tree-like structure. The most general concept can be a universal concept, whereas the most specific concepts correspond to the specific values of attributes in the database, as Fig. 1. Each node in a concept hierarchy represents a concept which helps to express knowledge and data relationships in a database as a concise, high level term [4]. Formally, we define a hierarchy \( \mathcal{H} \) on a set of domains \( D_1, \ldots, D_k \), we have \( \mathcal{H} : \{D_1 \times \ldots \times D_k\} \Rightarrow \mathcal{H}_{i+1} \Rightarrow \ldots \Rightarrow \mathcal{H}_0 \), where \( \mathcal{H}_i \) denotes the set of concept at the primitive level, \( \mathcal{H}_{i+1} \) denotes the set of concept at one level higher than those at \( \mathcal{H}_i \) and \( \mathcal{H}_0 \) represents the most general concept on the top level denoted as “ANY”, as Fig. 2.

It is reasonable that various concept hierarchies can be constructed from the same attributes based on different users’ viewpoints or preferences. For constructing concept hierarchies, many methods for numerical or categorical data have been developed. We summarize the related methods
into three kinds in the following. The first kind of methods is to specify concept hierarchies by users, experts or data analysts according to a partial or total ordering of attributes explicitly at the schema level. A user or expert can easily define a concept hierarchy by semantics of schema or his/her need. Another kind of methods is that given a set of attributes, the system then tries to generate the partial ordering of attributes automatically so as to construct a meaningful concept hierarchy. Some related researches on generating hierarchies in a single numerical attribute [4] and finding the partial ordering in a set of categorical attributes [8] have been proposed. The third kind of methods is to define a portion of a concept hierarchy by clustering explicit data in databases. Most of methods of this kind focus on automatic generation of concept by grouping data using clustering techniques. Without knowledge of data semantics, a concept hierarchy will be generated from the distinct values of attributes at the lowest level of hierarchy; then the more general concept are generated in higher levels. The related researches include Cluster/2 by Michalski and Stepp [11], COBWEB by Fisher [3], parallel clustering by Hong and Mao [7], generality-based clustering by Talavera and Bejar [16], etc. It is still an interesting and important problem for finding complete a hierarchical structure from a large database efficiently and meaningfully.

The purpose of this paper is to automatically construct a meaningful concept hierarchy in databases. We introduce rough entropy based on rough sets to measure the generality of concept described by the subset of attributes in a database. We had proved that the values of rough entropy for a set of attributes are increasing monotonously while the attributes are removed one by one from the set of attributes. This property accords with the fact that concept in higher level of hierarchy is more general than concept in lower level. Based on proposed rough entropy, we give a concept clustering algorithm to generate main concept levels and group the most relevant concepts into one. For producing more detailed hierarchy structures, the concept refinement algorithm is designed. To evaluate the performance of the proposed scheme, some experiments and comparisons with the other researches are made. The results show that the proposed method is efficient and the mined concept levels contain meaningful concept knowledge.

The paper is organized as follows: Section 2 introduces rough sets theory and rough entropy. In Section 3, we presented the two proposed algorithms based on rough entropy for generating and refining the concept hierarchy from categorical databases. In Section 4, we describe the experimental results. Conclusions and future work are made in Section 5.

II. ROUGH ENTROPY

The rough set theory was first proposed by Zdzislaw Pawlak in 1982 [12][14]. Rough set is used to deal with the identification of common attributes in data sets [13]. It has been broadly and successfully applied to knowledge discovery. The theory provides a powerful foundation to reveal and discover important structures in data and to classify complex objects. An attribute-oriented rough set technique can reduce the computational complexity of learning processes and eliminate the unimportant or irrelevant attributes so that the knowledge can be learned from large databases efficiently.

In an information system, if some categories of objects cannot be distinguished by the available attributes and may be just able to be defined roughly or approximately, the rough set theory is suitable to deals with such information system. The idea of rough sets is based on establishment of equivalence classes on the given data set and supports two approximations called lower approximation and upper approximation. The lower approximation of a concept contains the equivalence classes that are certain to belong to X without ambiguity. The upper approximation of a concept contains the equivalence classes that cannot be described as not belonging to X. Let $S = (U, A)$ denote an information system where $U$ means a non-empty finite set of objects with a non-empty finite set of attributes $A$. For all $a \in A$ and each $B \subseteq A$, we define a binary indiscernible relation $R_a(B)$ in the following.

Definition 1: Let $x, y \in U$, we say that objects $x$ and $y$ are indiscernible if the equivalence relation $R_a(B)$ is satisfied on the set $U$, $R_a(B) = \{(x, y): x, y \in U, \forall a \in B, a(x) = a(y)\}$.

Definition 2: Let $apr = (U, R_a)$ be an approximation space. The object $x \in U$ belongs to one and only one equivalence class. Let $[x]_a$ denote an equivalence class of $R_a(B)$ and $[U]_a$ denote the set of all equivalence classes $[x]_a$ for $x \in U$. We have $[x]_a = \{y \mid xR_a(B)y, x, y \in U\}$ and $[U]_a = \{[x]_a \mid x \in U\}$.

Definition 3: Let $S = (U, A)$, $x_i \in U$ and $B \subseteq A$. The rough entropy of information based on a subset of attributes $B$ is $E(B) = -\sum_{i=1}^{n} \frac{1}{|[x_i]_B|} \log_2 \frac{1}{|[x_i]_B|}$, where $|[x_i]_B|$ denotes the cardinality of the equivalence class of $[x_i]_B$ and $n$ is the number of objects in $U$.

Theorem 1: Let $S = (U, A)$, $B \subseteq A$ and $B' \subseteq A$. If $[x_i]_B \subseteq [x_i]_{B'}$, then $E(B) \leq E(B')$.

Corollary 1: Let $S = (U, A)$, $B \subseteq A$ and $B' \subseteq A$. If $B' \subseteq B$, then $E(B) \leq E(B')$.
the concept is more general while the value of rough entropy is larger as Corollary 1 states.

III. THE ALGORITHMS

A. Notations

The symbols used in our proposed algorithms are defined in the following.

\( U \) : A database with categorical attributes.
\( n \) : The number of objects in \( U \).
\( A \) : The finite set of categorical attributes in \( U \).
\( m \) : The number of attributes in \( A \).
\( a_j \) : The \( j \)-th attribute of \( A \).
\( [x_j]_a \) : The equivalence class of \( x_j \) satisfying \( R_a(B), x_j \in U \).
\(|[x_j]_a|\) : The cardinality of \([x_j]_a, x_j \in U \).
\([U]_a \) : The partition of \( R_a(B) \) on \( U \), \([U]_a = \{[x_j]_a | x_j \in U \}. \)
\( C_k \) : The \( k \)-th concept.

We define the reduced set of attributes in the following.

Assume that \( B = \{a_1, a_2, \ldots, a_n\} \) is a set of attributes. Let \( B_j \) be a reduced set with \( m' \) attributes of \( B \), \( m' < m \). \( B_j = \{b_1, b_2, \ldots, b_{m'}\} \) is a subset of \( B \), where \( b_i, b_j \in B \) and \( b_i \neq b_j \) for \( 1 \leq i, j \leq m' \). Let \( P_m(B) \) denote the set of all possible reduced sets with \( m' \) attributes of \( B \). Thus, the number of attributes in \( B_j = |B_j| = m' \) and \( |P_m(B)| = \binom{m}{m'} \).

B. The Concept Clustering and Refinement Algorithms

After the definitions, we describe the algorithms for clustering hierarchical concept with respect to a given database.

Algorithm: The hierarchical concept clustering
Input: A database \( U \) with the set of attributes \( A \).
Output: A concept hierarchy \( H \) for \( S = (U, A) \).
Step 1: Initialize the values of \( m' = m, k = 1 \) and \( B = A \).
Step 2: Generate the concepts \( C_k \) of lowest level \( H_m \) with the set of attributes \( A \). That is, \( H_m = [U]_A \).
Step 3: Let \( |B| \) be the number of attributes in \( B \), we set \( m' = |B| - 1 \) and find \( P_m(B) \).
Step 4: Compute the rough entropy \( E(B) \) for all \( B_j \in P_m(B) \).
Step 5: Generate a higher concept level \( H_m' \) by the following two sub-steps:

5.1) Let \( B' \) be the reduced set \( B_j \) with \( \max \{E(B)\} \), for \( B_j \in P_m(B) \); then we generate new concepts \( C_k \) in \( H_m' \) with the partition \( [U]_B \). The \( C_k \) is the concept that merge the concepts \( C_i \) in the lower level \( H_{m'+1} \) belonging to equivalence class \( [x]_B \).\)

5.2) For the others \( C_i \), in the lower level \( H_{m'+1} \), if \( C_i \equiv [x]_B \), the \( C_i \) is unchanged and added into \( H_m' \).
Step 6: Let \( B = B' \), if \( B \neq \emptyset \), go to Step 3; otherwise, output all concept levels \( H = \{H_0, \ldots, H_m\} \).

The time complexity of the proposed algorithm mainly depends on the partitioning of equivalence class \([U]_a\) and the number of reduced sets of attributes of each processing iteration. Assume that the time complexity of partitioning \( U \) is \( T(n) \). In Step 3 of the algorithm needs \( O(m) \) to generate \( P_m(B) \). Then, we have to spend \( O(mT(n)) \) to compute the values of rough entropy \( E(B) \) for all \( B_j \in P_m(B) \) in Step 4. In Step 5, the time for generating a single concept level is less than \( n \). Finally, the number of attributes is decreased when a new concept level is generated. The overall time complexity is \( O(m^2 T(n)) \). The time complexity of finding partitions is obviously bounded by the sorting problem. Hence, the time complexity of the algorithm is no more than \( O(m^2 n \log n) \). We give an example to help readers to understand the above algorithm.

**TABLE I.**

<table>
<thead>
<tr>
<th>data</th>
<th>gender</th>
<th>income</th>
<th>insurance</th>
<th>status</th>
<th>car</th>
</tr>
</thead>
<tbody>
<tr>
<td>x_1</td>
<td>male</td>
<td>Low</td>
<td>no</td>
<td>single</td>
<td>no</td>
</tr>
<tr>
<td>x_2</td>
<td>male</td>
<td>Low</td>
<td>yes</td>
<td>divorce</td>
<td>no</td>
</tr>
<tr>
<td>x_3</td>
<td>female</td>
<td>High</td>
<td>yes</td>
<td>divorce</td>
<td>yes</td>
</tr>
<tr>
<td>x_4</td>
<td>female</td>
<td>High</td>
<td>yes</td>
<td>married</td>
<td>yes</td>
</tr>
<tr>
<td>x_5</td>
<td>boy</td>
<td>Low</td>
<td>yes</td>
<td>yet</td>
<td>no</td>
</tr>
<tr>
<td>x_6</td>
<td>girl</td>
<td>Low</td>
<td>yes</td>
<td>yet</td>
<td>no</td>
</tr>
<tr>
<td>x_7</td>
<td>female</td>
<td>Low</td>
<td>no</td>
<td>married</td>
<td>yes</td>
</tr>
<tr>
<td>x_8</td>
<td>male</td>
<td>High</td>
<td>yes</td>
<td>single</td>
<td>yes</td>
</tr>
</tbody>
</table>

Example 1: TABLE I shows a database containing eight objects \( U = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8\ \} \) and five attributes \( A = \{\text{gender}, \text{income}, \text{insurance}, \text{status}, \text{car}\} \). The symbolic categories in the five attributes, respectively, are: \( \text{gender} : \{\text{male}(m), \text{female}(f), \text{boy}(b), \text{girl}(g)\} \), \( \text{income} : \{\text{high}(h), \text{low}(l)\} \), \( \text{insure} : \{\text{yes}(y), \text{no}(n)\} \), \( \text{status} : \{\text{single}, \text{married}, \text{divorce}, \text{yet}, \text{no}\} \) and \( \text{car} : \{\text{yes}, \text{no}\} \). The hierarchical result of clustering is shown in Fig. 3. The steps are executed as follows:

Input: The dataset as TABLE I shows.
Output: The concept hierarchy \( H \).
Step 1: Initialize the values of \( m' = m = 5, k = 1 \) and \( B = A = \{\text{gender}, \text{income}, \text{insurance}, \text{status}, \text{car}\} \).
Step 2: Generate the concepts of lowest level \( H_m \) with the set of attributes \( A \).
\( H_5 = [U]_A = [U]_{\{\text{gender}, \text{income}, \text{insurance}, \text{status}, \text{car}\}} = \{C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8\} \)
\( = \{[x_1]_{E}[x_2]_{E}[x_3]_{E}[x_4]_{E}[x_5]_{E}[x_6]_{E}[x_7]_{E}[x_8]_{E}\} \).
Step 3: We set \( m' = |B| - 1 = 4 \). Find \( P_4(B) \), which is the set of all possible reduced sets with 4 attributes of \( B \).
\( P_4(B) = \{B_j | B_j \subseteq B, |B_j| = m'\} = \{B_1, B_2, B_3, B_4\} \), where \( B_1 = \{\text{income, insurance, status, car}\} \), \( B_2 = \{\text{gender, insurance, status, car}\} \), \( B_3 = \{\text{gender, income, status, car}\} \), \( B_4 = \{\text{gender, income, insurance, car}\} \), \( B_5 = \{\text{gender, income, insurance, status}\} \).
Step 4: Compute the values of \( E(B) \) for all \( B_j \in P_4(B) \).
\( E(B_1) = E(\{\text{income, insurance, status, car}\}) = 4 \), \( E(B_2) = E(\{\text{gender, insurance, status, car}\}) = 0 \),
Algorithm: The concept refinement algorithm.

Input: A concept hierarchy $H$ for $S = (U, A)$.
Output: A refined concept hierarchy $H^r$ for $S = (U, A)$.
Step 1: Let $j = 1$, $B_j$ be the reduced set of attribute used in $H_j$, and $C_k$ is the top concept in $H_0$. We set $k = k + 1$.
Step 2: For all the reduced sets of attributes $B \in \mathcal{P}(B_{j-1})$ and $B \neq B_j$, repeat Step 3 to Step 5 to refine the concept level $H_j$.
Step 3: Find the partition of $[U]_B = \{[x]_B | x \in U\}$.
Step 4: If there is $[x]_B$ existing objects that belong to distinct concepts in $H_j$, go to Step 5; otherwise, go to Step 2.
Step 5: Generate the new concept $C_i$ in concept level of $H_j$, $C_i = [x]_B$. The $[x]_B$ is the equivalence class satisfying the condition in Step 4.
Step 6: If $j < m$, $j = j + 1$, go to Step 2 for refining the higher concept level; otherwise, the algorithm is halt.

Example 2: We continue to explain the concept refinement algorithm following the result of Example 1. The result of refinement is shown in Fig. 4. The steps are explained as the following example.

Input: A concept hierarchy $H$ in Fig. 3.
Output: A refined concept hierarchy $H^r$ for $S = (U, A)$.
Step 1: $j = 1$, $k = 16$ and $B_1 = \{income\}$, $H_1 = \{C_{14}, C_{12}\}$.
Step 2: $P(B_{j-1}) = P(B_2) = \{\{income\}, \{car\}\}$, for $B \in \mathcal{P}(B_2)$ and $B \neq B_1 = \{income\}$; that is $B = \{car\}$, execute Step 3 to Step 5 to refine the concept level $H_1$.
Step 3: Find partition $[U]_{\{car\}} = \{C_i | C_i \in \mathcal{C}_{12}\}$.
Step 4: Because there is an object $x_7$ in $C_i$ belonging to distinct concepts $C_{14}$ and $C_{12}$ in $H_1$ at the same time, the next step goes to Step 5.
Step 5: Generate the new concept $C_16$ in concept level of $H_1$, $C_{16} = C_i \cup C_{12}$.
Step 6: $j = 1 < m = 5$, $j = j + 1 = 2$, the algorithm repeats Step 2 for refining the higher concept level $H_j$ until $j = 5$.

It is easy to know that the time complexity of the concept refinement algorithm is still the same to the concept clustering algorithm. While the detailed concept we need is only the part of whole hierarchy, our scheme can help to find the useful concept levels without wasting time to generate all detailed concept hierarchy.

IV. EXPERIMENTAL RESULTS

To evaluate the performance of the proposed scheme, we make some experiments and comparisons using several test data sets. The experiments are done by using Pentium III 800MHz CPU with 128MB RAM. Test datasets are selected from UCI Machine Learning Repository [2] summarized in TABLE II.
TABLE II.
SUMMARY OF SELECTED DATASETS

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Attributes</th>
<th>Instances</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>BCW</td>
<td>9</td>
<td>683</td>
<td>2</td>
</tr>
<tr>
<td>CLEVE</td>
<td>13</td>
<td>296</td>
<td>2</td>
</tr>
<tr>
<td>CRX</td>
<td>15</td>
<td>653</td>
<td>2</td>
</tr>
<tr>
<td>GLASS</td>
<td>9</td>
<td>214</td>
<td>7</td>
</tr>
<tr>
<td>HORSE</td>
<td>12</td>
<td>326</td>
<td>2</td>
</tr>
<tr>
<td>PIMA</td>
<td>8</td>
<td>768</td>
<td>2</td>
</tr>
</tbody>
</table>

For observing the hierarchy structure generated by the proposed concept hierarchy generating algorithm, all numeric features were discretized by the functions in MLC++ [10] to fit the requirements of the proposed scheme and performed the tasks of test. In order to evaluate the effectiveness of the concept level, we perform learning and prediction processes. The decision attribute is used only at prediction, but hidden during the learning process. After the learning process, we use the following measurement function and a threshold \( \alpha \) to label the concept in the concept hierarchy to predict the test data. The function is

\[
\mu^X_B(x) = \frac{|[x]_B \cap X|}{|[x]_B|},
\]

where \( X \) denotes the concept of decision attribute, \( B \) is the reduced set of attributes at that concept level. We select the nodes with \( \mu^X_B(x) > \alpha \) as the concept we predict. TABLE III shows predictive accuracies and number of nodes examined in prediction for original hierarchy and refined hierarchy. The experiments are done with 10-fold cross validation. Considering the best accuracies on each dataset among the methods, the method with refinement is better than without refinement. We compare the results of the generated concept hierarchy with refinement with GCF [16] and COBWEB [3] methods. TABLE IV shows the comparison including the average accuracy and number of nodes examined to make the predictions with the standard deviations. The average number of nodes examined can be viewed as a measure of the complexity of the hierarchy. Results show that none of the systems is the best for every domain, thus suggesting that each one may be better suited for a particular type of problem. Considering only the best accuracies for three systems on each data set, our algorithm needs to examine fewer nodes on average to attain the same accuracy or the better accuracy. COBWEB uses an objective function that implicitly weights features and it is intended to produce predictive clusters. GCF model uses similarity metrics which are sensitive to the choice of features and may need some information about feature weights to be more robust. This may the reason for COBWEB performing much better than GCF and our approach in some domains. Experimental results confirm that our system traverses fewer nodes than previous work for making good prediction in average.

V. CONCLUSIONS

Knowledge discovery from a large database is an important research topic of late years. The concept hierarchy is an explicit representation of knowledge and can be widely used in many applications. In this paper, we present a new scheme for generating concept hierarchies based on rough set and rough entropy. The scheme contains the concept clustering algorithm and the concept refinement algorithm. We can fast construct a meaningful concept hierarchy by the concept clustering algorithm first, and then refine the concept level by the concept refinement algorithm as users’ needed. The experimental results demonstrate that the proposed scheme is efficient and the generated concept hierarchy is effective.
the future, we are going to extend the research to cope with the data with missing values.

ACKNOWLEDGEMENT

This work was supported in part by the National Science Council of Taiwan, R.O.C. under grant no. NSC-92-2213-E-214-005

REFERENCES


<table>
<thead>
<tr>
<th>Data sets</th>
<th>GCF Accuracy (0.34)</th>
<th>#Nodes 0.46</th>
<th>COBWEB Accuracy (0.17)</th>
<th>#Nodes 0.31</th>
<th>$H^p$ of proposed scheme Accuracy (0.04)</th>
<th>#Nodes 0.34</th>
</tr>
</thead>
<tbody>
<tr>
<td>BCW</td>
<td>92.36 (0.34)</td>
<td>5.20(0.46)</td>
<td>94.76(0.17)</td>
<td>5.17(0.31)</td>
<td>92.97(0.04)</td>
<td>4.66(0.34)</td>
</tr>
<tr>
<td>CRX</td>
<td>92.36 (0.42)</td>
<td>7.19(0.41)</td>
<td>95.18(0.74)</td>
<td>8.64(0.52)</td>
<td>91.51(0.04)</td>
<td>5.75(0.10)</td>
</tr>
<tr>
<td>CLEVE</td>
<td>82.42 (0.92)</td>
<td>6.39(0.57)</td>
<td>82.48(1.92)</td>
<td>9.11(0.29)</td>
<td>81.32(0.04)</td>
<td>9.28(0.88)</td>
</tr>
<tr>
<td>GLASS</td>
<td>82.16 (0.94)</td>
<td>5.66(0.92)</td>
<td>80.84(1.91)</td>
<td>5.81(0.26)</td>
<td>80.09(0.03)</td>
<td>9.65(0.07)</td>
</tr>
<tr>
<td>HORSE</td>
<td>81.30 (1.20)</td>
<td>2.89(0.69)</td>
<td>78.77(3.05)</td>
<td>2.97(0.09)</td>
<td>76.26(0.05)</td>
<td>10.2(0.06)</td>
</tr>
<tr>
<td>HORSE</td>
<td>81.46 (0.67)</td>
<td>2.05(0.05)</td>
<td>81.70(1.51)</td>
<td>2.02(0.06)</td>
<td>76.69(0.08)</td>
<td>7.8(0.19)</td>
</tr>
<tr>
<td>CRX</td>
<td>82.06 (1.41)</td>
<td>4.75(0.78)</td>
<td>81.21(2.37)</td>
<td>5.09(0.30)</td>
<td>73.99(0.10)</td>
<td>4.27(0.74)</td>
</tr>
<tr>
<td>GLASS</td>
<td>81.80 (1.12)</td>
<td>7.79(1.30)</td>
<td>80.69(1.78)</td>
<td>8.71(0.33)</td>
<td>73.31(0.07)</td>
<td>7.05(0.08)</td>
</tr>
<tr>
<td>PIMA</td>
<td>59.08 (1.04)</td>
<td>10.6(0.24)</td>
<td>57.73(2.70)</td>
<td>8.29(0.43)</td>
<td>56.07(0.10)</td>
<td>3.53(0.29)</td>
</tr>
<tr>
<td>PIMA</td>
<td>58.01 (1.40)</td>
<td>5.82(0.19)</td>
<td>56.19(3.22)</td>
<td>5.07(0.30)</td>
<td>54.67(0.08)</td>
<td>3.95(0.15)</td>
</tr>
<tr>
<td>GLASS</td>
<td>44.89 (1.59)</td>
<td>2.64(0.24)</td>
<td>42.68(3.92)</td>
<td>2.22(0.34)</td>
<td>54.67(0.10)</td>
<td>4.16(0.08)</td>
</tr>
<tr>
<td>HORE</td>
<td>72.56 (2.09)</td>
<td>6.82(1.01)</td>
<td>75.99(2.62)</td>
<td>8.72(0.41)</td>
<td>77.91(0.20)</td>
<td>4.86(0.69)</td>
</tr>
<tr>
<td>HORSE</td>
<td>71.61 (3.11)</td>
<td>4.97(0.86)</td>
<td>75.88(2.16)</td>
<td>5.14(0.26)</td>
<td>76.99(0.29)</td>
<td>6.7(0.14)</td>
</tr>
<tr>
<td>HORSE</td>
<td>66.68 (3.01)</td>
<td>2.57(0.24)</td>
<td>63.56(2.21)</td>
<td>2.05(0.07)</td>
<td>73.31(0.28)</td>
<td>6.86(0.13)</td>
</tr>
<tr>
<td>PIMA</td>
<td>68.58 (1.29)</td>
<td>7.80(0.90)</td>
<td>67.64(1.86)</td>
<td>8.87(0.57)</td>
<td>67.84(0.06)</td>
<td>4.76(0.08)</td>
</tr>
<tr>
<td>PIMA</td>
<td>67.63 (1.06)</td>
<td>5.16(0.77)</td>
<td>66.05(1.81)</td>
<td>5.70(0.54)</td>
<td>67.32(0.06)</td>
<td>4.81(0.06)</td>
</tr>
<tr>
<td>PIMA</td>
<td>65.90 (0.86)</td>
<td>2.62(0.21)</td>
<td>64.72(1.39)</td>
<td>2.66(0.37)</td>
<td>67.32(0.05)</td>
<td>4.82(0.06)</td>
</tr>
</tbody>
</table>