

Cascade Classifiers for Hierarchical Decision Systems

Zbigniew W. Raś^{1,3,4}, Agnieszka Dardzińska², and Wenxin Jiang¹

¹ Univ. of North Carolina, Dept. of Comp. Science, Charlotte, NC 28223, USA

² Białystok Technical Univ., Dept. of Computer Science, 15-351 Białystok, Poland

³ Polish-Japanese Institute of Information Technology, 02-008 Warsaw, Poland

⁴ Polish Academy of Sciences, Inst. of Comp. Science, 01-237 Warsaw, Poland

Abstract

Hierarchical classifiers are usually defined as methods of classifying inputs into defined output categories. The classification occurs first on a low-level with highly specific pieces of input data. The classifications of the individual pieces of data are then combined systematically and classified on a higher level iteratively until one output is produced. This final output is the overall classification of the data. In this paper we follow a controlled devise type of approach. The initial group of classifiers is trained using all objects in an information system S partitioned by values of the decision attribute d at its all granularity levels (one classifier per level). Only values of the highest granularity level (corresponding granules are the largest) are used to split S into information sub-systems where each one is built by selecting objects in S of the same decision value. These sub-systems are used for training new classifiers at all granularity levels of its decision attribute. Next, we split each sub-system further by sub-values of its decision value. The obtained tree-structure with groups of classifiers assigned to each of its nodes is called a cascade classifier.

Given an incomplete information system with a hierarchical decision attribute d , we consider the problem of training classifiers describing values of d at its lowest granularity level. Taking *MIRAI* database of music instrument sounds (Raś et al., 2007b), as an example, we show that the confidence of such classifiers can be lower than the confidence of cascade classifiers.

Keywords: data mining, hierarchical information systems, music information retrieval

1 Introduction

One of the main goals in data mining area is to describe knowledge hidden in data sets by means of classifiers. Clearly there is a need for classifiers which are easy and quick to build, accurate and suitable for different type of data. If data are incomplete, then null-value imputation techniques, including Chase (Dardzińska and Raś, 2005), (Dardzińska and Raś, 2003), can be used before the knowledge extraction algorithms are applied.

A hierarchical classifier is usually defined as agglomerative method of classifying inputs into defined output categories (Haskell, 1989), (Lu and Drew, 2001). The classification occurs first on a low-level with highly specific pieces of input data. The classifications of the individual pieces of data are then combined systematically and classified on a higher level iteratively until one output is produced. This final output is the overall classification of the data.

Automatic indexing of music by instruments and their types is taken as the application and testing area for our research. In (Zhang et al., 2008), a multi-hierarchical decision system S with a large number of descriptors built for describing music sound objects was described. The decision attributes in S are hierarchical and they include Hornbostel-Sachs classification and classification of instruments with respect to a playing method. The information richness hidden in these descriptors has strong implication on the confidence of classifiers built from S and used as a tool by the content-based Automatic Indexing Systems (*AIS*). Because decision attributes are hierarchical, then the indexing can be done with respect to different granularity levels of music instrument classes. This way, we identify not only the instruments playing in a given music piece but also classes of instruments. The quality of *AIS* was verified in (Zhang et al., 2008) using precision and recall based on two interpretations: user and system-based (Raś et al., 2007a). *AIS* engine follows system-based interpretation.

In this paper we propose a methodology of building cascade classifiers for regularly incomplete data sets. A data set is called regularly incomplete if it can be partitioned into non-singular subsets described by attributes which are either complete or entirely empty.

The initial group of classifiers is trained using all objects in an information system S partitioned by values of the decision attribute d at its all granularity levels (one classifier per level). Only values of the highest granularity level (corresponding granules are the largest) are used to split S into information sub-systems where each one is built by selecting objects in S of the same decision value. Attributes with no-values assigned to all objects of a sub-system of S are removed from that sub-system. These sub-systems are used for training new classifiers at all granularity levels of its decision attribute. Next, we split each sub-system further by sub-values of its decision value. The obtained tree-type structure with groups of classifiers assigned to each of its nodes is called a cascade classifier. There is some similarity between our approach and the work presented in (Grzymala-Busse, 2007), (Novotny, 1998).

Our cascade classifier has been tested on two classes of musical instrument sounds: one labelled as 4F and the other as 3B. The analysis shows a significant improvement in the precision of *AIS*, if cascade classifiers are used instead of single classifiers.

2 Multi-Hierarchical Decision System and Query Language

In this section we introduce the notion of a multi-hierarchical decision system S and the query language associated with S , built from values of decision attributes. Classifier-based semantics and standard semantics of queries in S was given in

TABLE 1: Decision System

	a	b	c	d
x_1		1	2	$d[1, 1]$
x_2		1	3	$d[1, 1]$
x_3	1	1	0	$d[1, 2]$
x_4	1	1	3	$d[1, 2]$
x_5	2		2	$d[2, 1]$
x_6	2		3	$d[2, 1]$
x_7		1	1	$d[1, 1]$
x_8		1	1	$d[1, 1]$
x_9	2		1	$d[2, 1]$
x_{10}	2		0	$d[2, 1]$
x_{11}	1	1	2	$d[2, 2]$
x_{12}	1	1	1	$d[2, 2]$

(Raś et al., 2007a). The set of objects X in S forms the interpretation domain for both semantics. Standard semantics identifies all correct objects in X which should be retrieved by a query. Classifier-based semantics gives weighted set of objects which actually are retrieved by a query. The notion of precision and recall of the query answering system (QAS) in the proposed setting was also introduced in (Raś et al., 2007a). By improving the confidence and support of classifiers trained by S , we also improve the precision and recall of QAS .

Multi-hierarchical decision systems (Raś et al., 2007a) can be seen as a subclass of decision systems (Pawlak, 1991) and they are mostly used for representing data which are incomplete. If a multi-hierarchical decision system contains only one decision attribute, then it is called a hierarchical decision system.

By a decision system we mean a pair $S = (U, A \cup \{d\}, V)$, where:

- U is a nonempty, finite set of objects,
- $A \cup \{d\}$ is a nonempty, finite set of attributes i.e. $a : U \rightarrow V_a$ is a partial function for any $a \in A$, where V_a is the domain of a ,
- elements in A are called classification attributes and d is a distinguished attribute called the decision. We assume that the decision is a total function,
- $V = \bigcup \{V_a : a \in A \cup \{d\}\}$.

As an example of a decision table we take $S = (\{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}\}, \{a, c\} \cup \{b\} \cup \{d\}, V)$ represented by Table 1. Decision attribute d is hierarchical with $V_d = \{d[1], d[2], d[1, 1], d[1, 2], d[2, 1], d[2, 2]\}$. Term $d[i, j]$ is a child of $d[i]$, for $i, j \in \{1, 2\}$.

By a multi-hierarchical decision system we mean a triple $S = (X, A \cup \{d[1], d[2], \dots, d[k]\}, V)$, where X is a nonempty, finite set of objects, A is a nonempty finite set of classification attributes, $\{d[1], d[2], \dots, d[k]\}$ is a set of hierarchical decision attributes and $V = \bigcup \{V_a : a \in A \cup \{d[1], d[2], \dots, d[k]\}\}$ is a set of their values. We assume that:

- V_a, V_b are disjoint for any $a, b \in [A \cup \{d[1], d[2], \dots, d[k]\}]$, such that $a \neq b$,
- $a : X \rightarrow V_a$ is a partial function for every $a \in A \cup \{d[1], d[2], \dots, d[k]\}$.

By a set of decision queries (d-queries) for S we mean a least set T_D such that:

- $0, 1 \in T_D$,
- if $w \in \bigcup \{V_a : a \in \{d[1], d[2], \dots, d[k]\}\}$, then $w, \sim w \in T_D$,
- if $t_1, t_2 \in T_D$, then $(t_1 + t_2), (t_1 * t_2) \in T_D$.

Decision query t is called simple if $t = t_1 * t_2 * \dots * t_n$ and $(\forall j \in \{1, 2, \dots, n\})(t_j \in \bigcup \{V_a : a \in \{d[1], d[2], \dots, d[k]\}\}) \vee (t_j = \sim w \wedge w \in \bigcup \{V_a : a \in \{d[1], d[2], \dots, d[k]\}\})$.

By a set of classification terms (c-terms) for S we mean a least set T_C such that:

- $0, 1 \in T_C$,
- if $w \in \bigcup \{V_a : a \in A\}$, then $w, \sim w \in T_C$,
- if $t_1, t_2 \in T_C$, then $(t_1 + t_2), (t_1 * t_2) \in T_C$.

Classification term t is called simple if $t = t_1 * t_2 * \dots * t_n$ and $(\forall j \in \{1, 2, \dots, n\})(t_j \in \bigcup \{V_a : a \in A\}) \vee (t_j = \sim w \wedge w \in \bigcup \{V_a : a \in A\})$.

By a classification rule we mean any expression of the form $[t_1 \longrightarrow t_2]$, where t_1 is a simple classification term and t_2 is a simple decision query.

Semantics M_S of c-terms in $S = (X, A \cup \{d[1], d[2], \dots, d[k]\}, V)$, is defined in a standard way as follows:

- $M_S(0) = 0, M_S(1) = X$,
- $M_S(w) = \{x \in X : w = a(x)\}$ for any $w \in V_a, a \in A$,
- $M_S(\sim w) = \{x \in X : (\exists v \in V_a)[(v = a(x)) \wedge (v \neq w)]\}$ for any $w \in V_a, a \in A$,
- if t_1, t_2 are terms, then $M_S(t_1 + t_2) = M_S(t_1) \cup M_S(t_2)$, $M_S(t_1 * t_2) = M_S(t_1) \cap M_S(t_2)$.

Now, we introduce the notation we are going to use in this paper for values of decision attributes. Assume that $d[i]$ is a hierarchical decision attribute which is also interpreted as its first granularity level. The set $\{d[i, 1], d[i, 2], d[i, 3], \dots\}$ represents the values of attribute $d[i]$ at its second granularity level. The set $\{d[i, 1, 1], d[i, 1, 2], \dots, d[i, 1, n_i]\}$ represents the values of attribute d at its third granularity level, right below the node $d[i, 1]$. We assume here that the value $d[i, 1]$ can be refined to any value from $\{d[i, 1, 1], d[i, 1, 2], \dots, d[i, 1, n_i]\}$, if necessary. Similarly, the set $\{d[i, 3, 1, 3, 1], d[i, 3, 1, 3, 2], d[i, 3, 1, 3, 3], d[i, 3, 1, 3, 4]\}$ represents the values of attribute d at its fourth granularity level which are finer than the value $d[i, 3, 1, 3]$.

Now, let us assume that a rule-based classifier (for instance one of the modules in systems *RSES* or *WEKA*) was used to extract rules describing simple decision queries in S . We denote this classifier by *RC*. The definition of semantics of c-terms does not depend on a classifier but the definition of semantics M_S of d-queries is a classifier dependent.

Classifier-based semantics M_S of d-queries in $S = (X, A \cup \{d[1], d[2], \dots, d[k]\}, V)$, is defined as follows:

if t is a simple d -query in S and $\{r_j = [t_j \longrightarrow t] : j \in J_t\}$ is a set of all rules defining t which are extracted from S by classifier RC , then $M_S(t) = \{(x, p_x) : (\exists j \in J_t)(x \in M_S(t_j)[p_x = \Sigma\{conf(j) \cdot sup(j) : x \in M_S(t_j) \wedge j \in J_t\} / \Sigma\{sup(j) : x \in M_S(t_j) \wedge j \in J_t\}])\}$, where $conf(j)$, $sup(j)$ denote the confidence and the support of $[t_j \longrightarrow t]$, correspondingly.

Attribute value $d[j_1, j_2, \dots, j_n]$ in $S = (X, A \cup \{d[1], d[2], \dots, d[k]\}, V)$ is dependent on $d[i_1, i_2, \dots, i_k]$ in S , if one of the following conditions hold: (1) $n \leq k \wedge (\forall m \leq n)[i_m = j_m]$, (2) $n > k \wedge (\forall m \leq k)[i_m = j_m]$. Otherwise, $d[j_1, j_2, \dots, j_n]$ is called independent from $d[i_1, i_2, \dots, i_k]$ in S .

Example. The attribute value $d[2, 3, 1, 2]$ is dependent on the attribute value $d[2, 3, 1, 2, 5, 3]$. Also, $d[2, 3, 1, 2, 5, 3, 2, 4]$ is dependent on $d[2, 3, 1, 2, 5, 3]$.

Let $S = (X, A \cup \{d[1], d[2], \dots, d[k]\}, V)$, $w \in V_{d[i]}$, and $IV_{d[i]}$ be the set of all attribute values in $V_{d[i]}$ which are independent from w . Standard semantics N_S of d -queries in S is defined as follows:

- $N_S(0) = 0$, $N_S(1) = X$,
- if $w \in V_{d[i]}$, then $N_S(w) = \{x \in X : d[i](x) = w\}$, for any $1 \leq i \leq k$
- if $w \in V_{d[i]}$, then $N_S(\sim w) = \{x \in X : (\exists v \in IV_{d[i]})[d[i](x) = v]\}$, for any $1 \leq i \leq k$
- if t_1, t_2 are terms, then $N_S(t_1 + t_2) = N_S(t_1) \cup N_S(t_2)$, $N_S(t_1 * t_2) = N_S(t_1) \cap N_S(t_2)$.

Let $S = (X, A \cup \{d[1], d[2], \dots, d[k]\}, V)$, t is a d -query in S , $N_S(t)$ is its meaning under standard semantics, and $M_S(t)$ is its meaning under classifier-based semantics. Assume that $N_S(t) = X_1 \cup Y_1$, where $X_1 = \{x_i, i \in I_1\}$, $Y_1 = \{y_i, i \in I_2\}$. Assume also that $M_S(t) = \{(x_i, p_i) : i \in I_1\} \cup \{(z_i, q_i) : i \in I_3\}$ and $\{y_i, i \in I_2\} \cap \{z_i, i \in I_3\} = \emptyset$.

By precision of a classifier-based semantics M_S on a d -query t , we mean $Prec(M_S, t) = [\sum\{p_i : i \in I_1\} + \sum\{(1 - q_i) : i \in I_3\}] / [card(I_1) + card(I_3)]$.

By recall of a classifier-based semantics M_S on a d -query t , we mean $Rec(M_S, t) = [\sum\{p_i : i \in I_1\}] / [card(I_1) + card(I_2)]$.

Clearly, the precision and recall of a classifier-based semantics can be improved by using classifiers of higher confidence. In the remaining part of the paper, we show how to construct the cascade λ -representation of a multi-hierarchical decision system and how it is used to construct cascade classifiers which confidence is usually higher than the confidence of standard classifiers.

3 Cascade λ -Representation of Hierarchical Decision Systems

For simplicity reason, we only consider multi-hierarchical decision systems with one decision attribute. Such systems are called hierarchical decision systems.

Let $S(d) = (X, A \cup \{d\}, V)$ is a regularly incomplete decision system, where d is a hierarchical attribute. Its values are represented following the notation proposed in the previous section. Figure 1 shows an example of a hierarchical decision attribute.

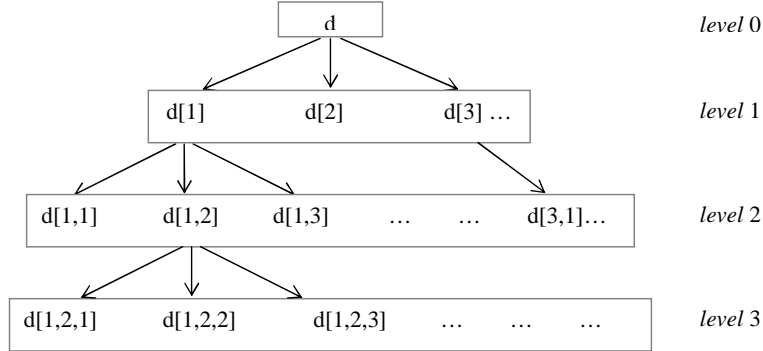


FIGURE 1: Example of a decision attribute

Let $\{d[1], d[2], \dots, d[k]\}$ is a set of all values of the attribute d at level 1 of its tree representation. Let $X_i = \{x \in X : d(x) = d[i]\}$ and $S_i[d_i] = (X_i, A \cup \{d[i]\}, V)$, for any $1 \leq i \leq n$. Now, assume that $CR(S)$ denotes a tree of high one. System S is its root and $S_i(d[i])$, ($1 \leq i \leq n$), are its children. The outgoing edge from S to $S_i(d[i])$ is labelled by $d[i]$, for any $1 \leq i \leq n$.

Cascade λ -representation of $S(d)$, where $\lambda \in [0, 1]$, is a tree with $S(d)$ defined as its root and all its descendants being built by executing the instruction [if $card(V_d) > 1$, then replace $S(d)$ by $CR(S(d))$] recursively, starting from the root and then repeating for all leaves of a constructed tree. The final step of this construction process is the removal of all λ -incomplete attributes in all decision subsystems of $S(d)$ representing descendants of $S(d)$. We say that an attribute a is λ -incomplete in S , if $a(x)$ is defined for $[1 - \lambda] * 100$ percent of objects in S .

Let us go back to the example of a decision system $S(d)$ represented as Table 1. Its attributes $\{a, b\}$ are 4/12-incomplete. To build a cascade λ -representation of $S(d)$, where $\lambda = 0$, we take its subsystems: $S^*(d) = (\{x_i : 1 \leq i \leq 12\}, \{c, d\}, V)$, $S_{[1]}(d[1]) = (\{x_i : i = 1, 2, 3, 4, 7, 8\}, \{b, c, d\}, V)$, $S_{[2]}(d[2]) = (\{x_i : i = 5, 6, 9, 10, 11, 12\}, \{a, c, d\}, V)$, $S_{[1,1]}(d[1, 1]) = (\{x_i : i = 1, 2, 7, 8\}, \{b, c, d\}, V)$, $S_{[1,2]}(d[1, 2]) = (\{x_i : i = 3, 4\}, \{a, b, c, d\}, V)$, $S_{[2,1]}(d[2, 1]) = (\{x_i : i = 5, 6, 9, 10\}, \{a, c, d\}, V)$, $S_{[2,2]}(d[2, 2]) = (\{x_i : i = 11, 12\}, \{a, b, c, d\}, V)$.

Now, the corresponding cascade λ -representation of $S(d)$, denoted as $[(\{S^*(d)\} \cup \{S_k(d) : k \in J\}, \prec)$, where $J = \{[1], [2], [1, 1], [1, 2], [2, 1], [2, 2]\}$ and " \prec " means *parent-child* relation, is represented in Figure 2.

The partition of objects in $S(d)$ can be driven by an optimization function or it can be predefined, as it is done in *MIRAI* (Raś et al., 2007b), by following either Hornbostel-Sachs classification or classification of instruments with respect to a playing method.

4 Cascade Classifiers

In this section, we show how to use the cascade λ -representation of $S(d)$ to build cascade classifiers for $S(d)$.

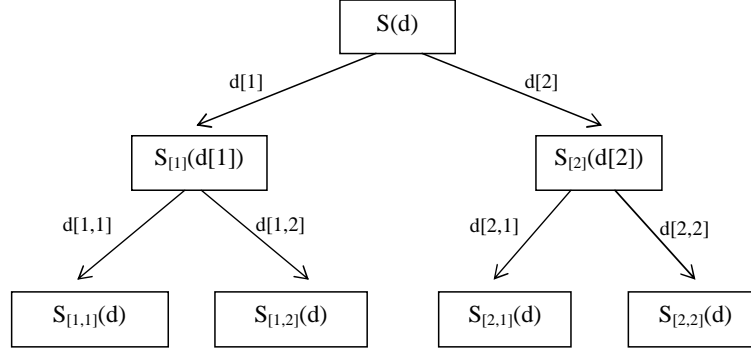


FIGURE 2: Cascade λ -representation of $S(d)$

Let $S(d) = (X, A \cup \{d\}, V)$ is a regularly incomplete decision system, where d is a hierarchical attribute. We follow the notation proposed in the previous section to represent its values, with $d[i]$ referring to a child of d and $d[i, j]$ to its grandchild. Also, we assume that $(Casc(S(d)), \prec)$, where $Casc(S(d)) = \{S_k(d) : k \in J\}$ is a cascade λ -representation of $S(d)$. Let $p_1 = [i_1, i_2, \dots, i_k]$ and $p_2 = [j_1, j_2, \dots, j_n]$. Relation \prec is defined as: $S_{p_1}(d) \prec S_{p_2}(d)$ iff $(k \leq n)$ and $(\forall m \leq k)[i_m = j_m]$. In all other cases \prec is undefined. Clearly, if d_1 is a descendent of d , then $(Casc(S(d_1)), \prec)$ is a cascade λ -representation of $S(d_1)$.

Let us assume that the height of $(Casc(S(d)), \prec)$ is n and $S_{[i_1, i_2, \dots, i_k]}(d) \in Casc(S(d))$. Clearly, $d[i_1, i_2, \dots, i_k]$ is the root of $S_{[i_1, i_2, \dots, i_k]}(d)$.

By $(S, d[i_1, i_2, \dots, i_k], m)$, where $k + 1 \leq m \leq n$, we denote a subtree of S with $d[i_1, i_2, \dots, i_k]$ as its root and all descendent of $d[i_1, i_2, \dots, i_k]$ at all levels between $k + 1$ and m .

Assume now that $class(S, d[i_1, i_2, \dots, i_k], m)$ denotes a classifier trained by $S(d[i_1, i_2, \dots, i_k])$ with the decision attribute $d[i_1, i_2, \dots, i_k]$ and its values restricted to level m of its tree representation. For example, $\{d[1, 1, 1], d[1, 1, 2], \dots, d[1, 3, 3]\}$ is the set of values for $S(d[1])$ at level 3 (see Figure 1).

By a cascade classifier of type λ for $S(d)$ we mean $(class(Casc(S(d))), \prec)$, where $class(Casc(S(d))) = \{class((S, d[i_1, i_2, \dots, i_k], m)) : [k + 1 \leq m \leq n] \wedge [[i_1, i_2, \dots, i_k] \in J]\}$ and $Casc(S(d)) = \{S(d[i_1, i_2, \dots, i_k]) : [i_1, i_2, \dots, i_k] \in J\}$ is a cascade λ -representation of $S(d)$. A sample representation structure for a cascade classifier is given in Figure 3.

So, three classifiers are associated with the root level of the tree represented by Figure 3. The first one (with $i=1$) is trained by S with values of the decision attribute defined as the largest granules. The last one (with $i=3$) is based on attribute values defined as the smallest granules.

5 Application Domain and Testing Results

Music instrument identification is one of the important subtasks of a content-based automatic indexing, for which authors built a multi-hierarchical decision

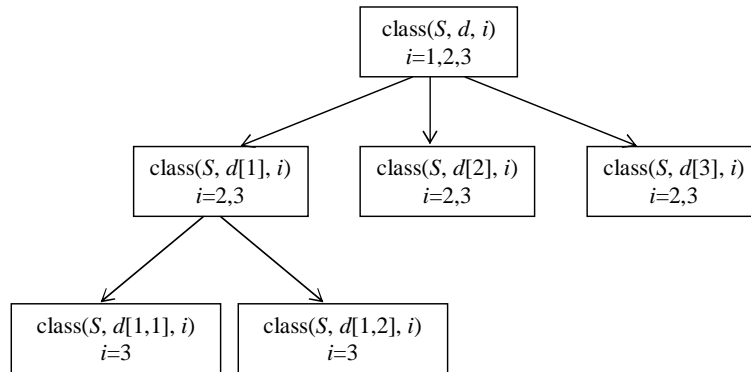
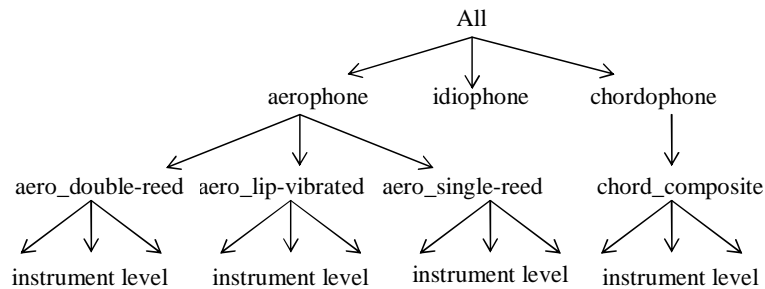
FIGURE 3: Cascade classifier for $S(d)$ 

FIGURE 4: Hornbostel-Sachs classification of instruments

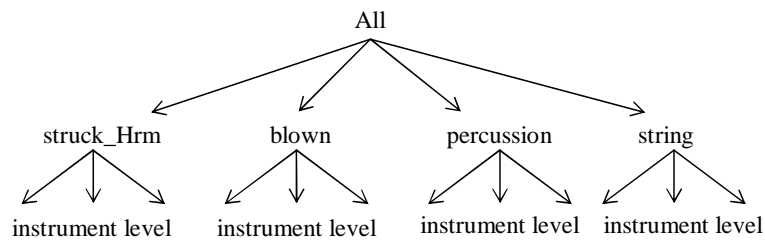


FIGURE 5: Classification of instruments with respect to playing method

system S describing 58 music instruments by more than one thousand attributes containing all low-level MPEG7 descriptors as well as other popular descriptors for describing music sound objects (see <http://www.mir.uncc.edu>). There is close to 1,000,000 objects in S . The decision attributes in S are hierarchical and they include Hornbostel-Sachs classification (see Figure 4) and classification of instruments with respect to playing method (see Figure 5).

The information richness hidden in descriptors has strong implication on the confidence of classifiers built from S . Rule-based classifiers give us approximate definitions of values of decision attributes and they are used as a tool by content-

TABLE 2: Modules of cascade classifier for classification of instruments with respect to playing method and their confidence

<i>root</i>	<i>classname</i>	<i>classifier</i>	<i>pos_support</i>	<i>confidence</i>
<i>d</i>	<i>all_instruments</i>	<i>class(S, d, 2)</i>	730	91.80%
<i>d</i>	<i>all_instruments</i>	<i>class(S, d, 1)</i>	750	94.26%
<i>d[1]</i>	<i>blown</i>	<i>class(S, d[1], 2)</i>	265	96.84%
<i>d[2]</i>	<i>string</i>	<i>class(S, d[2], 2)</i>	390	94.68%
<i>d[3]</i>	<i>struck_Hrn</i>	<i>class(S, d[3], 2)</i>	69	98.84%

TABLE 3: Modules of cascade classifier for Hombostel-Sachs classification of instruments and their confidence

<i>root</i>	<i>classname</i>	<i>classifier</i>	<i>pos_support</i>	<i>confidence</i>
<i>d</i>	<i>all_instruments</i>	<i>class(S, d, 1)</i>	771	96.97%
<i>d</i>	<i>all_instruments</i>	<i>class(S, d, 2)</i>	764	96.02%
<i>d</i>	<i>all_instruments</i>	<i>class(S, d, 3)</i>	730	91.80%
<i>d[1]</i>	<i>aerophone</i>	<i>class(S, d[1], 2)</i>	269	98.26%
<i>d[1]</i>	<i>aerophone</i>	<i>class(S, d[1], 3)</i>	265	96.84%
<i>d[2]</i>	<i>chordophone</i>	<i>class(S, d[2], 2)</i>	497	98.83%
<i>d[2]</i>	<i>chordophone</i>	<i>class(S, d[2], 3)</i>	466	92.75%
<i>d[3]</i>	<i>idiophone</i>	<i>class(S, d[3], 2)</i>	19	95.95%
<i>d[3]</i>	<i>idiophone</i>	<i>class(S, d[3], 3)</i>	19	95.95%
<i>d[1, 1]</i>	<i>aero_double_reed</i>	<i>class(S, d[1, 1], 3)</i>	70	98.94%
<i>d[1, 2]</i>	<i>aero_lip_vibrated</i>	<i>class(S, d[1, 2], 3)</i>	113	95.66%
<i>d[1, 3]</i>	<i>aero_side</i>	<i>class(S, d[1, 3], 3)</i>	10	90.91%
<i>d[1, 4]</i>	<i>aero_single_reed</i>	<i>class(S, d[1, 4], 3)</i>	72	99.54%
<i>d[2, 1]</i>	<i>chrd_composite</i>	<i>class(S, d[2, 1], 3)</i>	410	93.18%

based Automatic Indexing Systems (*AIS*) (Raś et al., 2007b). Hierarchical decision attributes allow us to have the indexing done on different granularity levels of classes of music instruments. We can identify not only the instruments playing in a given music piece but also classes of instruments if the instrument level identification fails. The quality of *AIS* can be verified using precision and recall based on two interpretations: user and system-based (Raś et al., 2007a). *AIS* engine follows system-based interpretation.

In this section we show that cascade classifiers outperform standard classifiers in the music information retrieval domain. The first step in the process of recognizing a dominating musical instrument in a musical piece is the identification of its pitch. If the pitch is found, then a pitch-dedicated classifier is used to identify this instrument. The testing was done for music instrument sounds of pitch 3B. The results are shown in Table 3 and Table 4. The confidence of a standard classifier $class(S, d, 3)$ for Hombostel-Sachs classification of instruments is 91.50%. However, we can get much better results by following

the cascade approach. For instance, if we use the classifier $class(S, d, 2)$ followed by the classifier $class(S, d[1, 1], 3)$, then its precision in recognizing musical instruments in *aero_double_reed* class is equal to $96.02\% * 98.94\% = 95.00\%$. Also, its precision in recognizing instruments in *aero_single_reed* class is equal to $96.02\% * 99.54\% = 95.57\%$. It has to be noted that this improvement in confidence is obtained without increasing the number of attributes in the subsystems of S used to build the cascade classifier replacing S . Clearly, if we increase the number of attributes in these subsystems then the resulting classifiers forming the cascade classifier may easily have higher confidence and the same the confidence of the cascade classifier will get increased.

6 Acknowledgement

This research was supported by the National Science Foundation under grant IIS-0414815.

References

- A. DARDZIŃSKA and Z.W. RAŚ (2005), CHASE-2: Rule based chase algorithm for information systems of type lambda, in *Postproceedings of the Second International Workshop on Active Mining (AM'2003)*, LNAI, No. 3430, Springer, pp. 258–270.
- A. DARDZIŃSKA and Z.W. RAŚ (2003), Chasing Unknown Values in Incomplete Information Systems, in *Proceedings of ICDM'03 Workshop on Foundations and New Directions of Data Mining*, Melbourne, Florida, IEEE Computer Society, pp. 24–30.
- J.W. GRZYMALA-BUSSE(2007), Mining numerical data - a rough set approach, in *Proceedings of RSEISP 2007*, LNAI, Vol. 4585, Springer, pp. 12-21.
- R.-E. HASKELL (1989), Design of hierarchical classifiers, in *Proceedings of the The First Great Lakes Computer Science Conference on Computing in the 90's*, LNCS, Vol. 507, pp. 118–124.
- C. LU and M.S. DREW (2001), Construction of a hierarchical classifier schema using a combination of text-based and image-based approaches, in *SIGIR'01 Proceedings*, ACM Publications, pp. 331–336.
- M. NOVOTNY(1998), Dependence spaces of information systems, in *Incomplete Information: Rough Set Analysis*, Physica-Verlag, pp. 193–246.
- Z. PAWLAK (1991), Information systems - theoretical foundations, in *Information Systems Journal*, Vol. 6, pp. 205-218.
- Z.W. RAŚ, A. DARDZIŃSKA, and X. ZHANG (2007), Cooperative Answering of Queries based on Hierarchical Decision Attributes, in *CAMES Journal*, Polish Academy of Sciences, Institute of Fundamental Technological Research, Vol. 14, No. 4, pp. 729–736.
- Z.W. RAŚ, X. ZHANG, and R. LEWIS (2007), MIRAI: Multi-hierarchical, FS-tree based music information retrieval system, in *Proceedings of RSEISP 2007*, LNAI, Vol. 4585, Springer, pp. 80–89.
- X. ZHANG, Z.W. RAŚ, and A. DARDZIŃSKA (2008), Discriminant Feature Analysis for Music Timbre Recognition and Automatic Indexing, in *Mining Complex Data*, Post-Proceedings of 2007 ECML/PKDD Third International Workshop (MCD 2007), LNAI, Vol. 4944, Springer, pp. 104-115.