Analogical Classification: A Rule-Based View

Myriam Bounhas, Henri Prade, Gilles Richard

Analogical Classification: A Rule-Based View

Myriam Bounhas[1], Henri Prade[2], and Gilles Richard[2]

[1] LARODEC Laboratory, ISG de Tunis, 41 rue de la Liberté, 2000 Le Bardo, Tunisia
& Emirates College of Technology, P.O. Box: 41009, Abu Dhabi, United Arab Emirates
[2] IRIT – CNRS, 118, route de Narbonne, Toulouse, France
myriam_bounhas@yahoo.fr, {prade,richard}@irit.fr

Abstract. Analogical proportion-based classification methods have been introduced a few years ago. They look in the training set for suitable triples of examples that are in an analogical proportion with the item to be classified, on a maximal set of attributes. This can be viewed as a lazy classification technique since, like k-nn algorithms, there is no static model built from the set of examples. The amazing results (at least in terms of accuracy) that have been obtained from such techniques are not easy to justify from a theoretical viewpoint. In this paper, we show that there exists an alternative method to build analogical proportion-based learners by statically building a set of inference rules during a preliminary training step. This gives birth to a new classification algorithm that deals with pairs rather than with triples of examples. Experiments on classical benchmarks of the UC Irvine repository are reported, showing that we get comparable results.

Introduction

Comparing objects or situations and identifying in what respects they are identical (or similar) and in what respects they are different, is a basic type of operations at the core of many intelligent activities. A more elaborate operation is the comparison between pairs of objects or situations, where a comparison has already been done inside the pairs. This corresponds to the idea of analogical proportions, i.e. statements of the form "A is to B as C is to D", denoted $A : B :: C : D$, expressing the fact "A differs from B as C differs from D", as well as "B differs from A as D differs from C" [5].

Analogical reasoning has been recognized for a long time as a powerful heuristic tool for solving problems. In fact, analogical proportions are not explicitly used in general. Compound situations identified as analogous are rather put in parallel, leading to the plausible conclusion that what holds in one case should also hold in the other case (up to suitable transpositions). However, analogical reasoning can also be directly based on analogical proportions. This requires a formalized view of these proportions. Such a modeling has been only recently developed in algebraic or logical settings [2,8,5,6]. Then analogical proportions turn to be a powerful tool in classification tasks [4].

We assume that the objects or situations A, B, C, D are represented by vectors of attribute values, denoted $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{d}$. The analogical proportion-based approach to classification relies on the idea that the unknown class $x = cl(\boldsymbol{d})$ of a new instance $\boldsymbol{d}$, may be predicted as the solution x of an equation expressing that the analogical proportion $cl(\boldsymbol{a}) : cl(\boldsymbol{b}) :: cl(\boldsymbol{c}) : x$ holds between the classes. This is done on the basis of triples

of examples $\boldsymbol{a}$, $\boldsymbol{b}$ and $\boldsymbol{c}$ of the training set that are such that th
$\boldsymbol{a} : \boldsymbol{b} :: \boldsymbol{c} : \boldsymbol{d}$ holds on vector components for all, or at least or
attributes describing the items. This approach has been tested o
results competitive with the ones of classical machine learning
tained. These good results have remained largely unexplaine
why this analogical proportion-based approach may be so effec
merits of analogy. In this paper, we investigate a new type of
induction of particular rules induced from pairs of examples, w
to analogical proportions, thus providing some light on the unde

The paper is organized as follows. First a background on a
provided, emphasizing noticeable properties important for appl
before discussing how they can be applied to this task. Then
proach is contrasted with the original triples-based approach. A
and their results on machine learning benchmarks are reported

A Short Background on Analogical Proportions

Analogical proportions are statements of the form "A is to B as
been supposed to continue to hold when the pairs (A, B) and $(C$
when the terms B and C are permuted, just like numerical pro
time; see, e.g., [7]. Thus, $A : B :: C : D$ is equivalent to $C : I$
and $A : B :: C : D$ is equivalent to $A : C :: B : D$ (central perm
symmetry and permutation, this leads to 8 equivalent forms.

In this paper, A, B, C, D are represented by Boolean vecto
ponent of such a vector $\boldsymbol{a}$. Then an analogical proportion be
be expressed componentwise, in a logical manner under vario
One remarkable expression of the analogical proportion is g
$a : b :: c : d = (a \wedge \neg b \equiv c \wedge \neg d) \wedge (\neg a \wedge b \equiv \neg c \wedge d)$.

As can be seen, this expression of the analogical proportion
and could be informally read as *what is true for a and not for
for c and not for d, and vice versa*. This logical expression mak
proportion $\boldsymbol{a} : \boldsymbol{b} :: \boldsymbol{c} : \boldsymbol{d}$ that $\boldsymbol{a}$ differs from $\boldsymbol{b}$ as $\boldsymbol{c}$ differs fr
differs from $\boldsymbol{a}$ as $\boldsymbol{d}$ differs from $\boldsymbol{c}$. The 6 cases (among $2^4 = 16$
the above Boolean expression is *true* are given in the truth Tabl

Table 1. When an analogical proportion is tru

a	b	c	d	$a : b :: c : d$
0	0	0	0	1
1	1	1	1	1
0	0	1	1	1
1	1	0	0	1
0	1	0	1	1
1	0	1	0	1

It can be easily checked on the above truth Table 1 that the logical expression of the analogical proportion indeed satisfies symmetry and central permutation. Assuming that an analogical proportion holds between four binary items, three of them being known, then one may try to infer the value of the fourth one. The problem can be stated as follows. Given a triple (a, b, c) of Boolean values, does there exist a Boolean value x such that $a : b :: c : x = 1$, and in that case, is this value unique? It is easy to see that there are cases where the equation has no solution since the triple a, b, c may take $2^3 = 8$ values, while A is true only for 6 distinct 4-tuples. Indeed, the equations $1 : 0 :: 0 : x = 1$ and $0 : 1 :: 1 : x = 1$ have no solution. It is easy to prove that the analogical equation $a : b :: c : x = 1$ is solvable iff $(a \equiv b) \vee (a \equiv c)$ holds true. In that case, the unique solution is given by $x = a \equiv (b \equiv c)$. Note that due to symmetry and permutation properties, there is no need to consider the equations $x : b :: c : d = 1$, $a : x :: c : d = 1$, and $a : b :: x : d = 1$ that can be handled in an equivalent way.

Analogical Proportions and Classification

Numerical proportions are closely related to the ideas of extrapolation and of linear regression, i.e., to the idea of predicting a new value on the ground of existing values, and to the idea of inducing general laws from data. Analogical proportions may serve similar purposes. The equation solving property recalled above is at the root of a brute force method for classification. It is based on a kind of proportional continuity principle: if the binary-valued attributes of 4 objects are componentwise in analogical proportion, then this should still be the case for their classes. More precisely, having a 2-class classification problem, and 4 Boolean objects $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{d}$ over $\mathbb{B}^n$, 3 in the training set with known classes $cl(\boldsymbol{a}), cl(\boldsymbol{b}), cl(\boldsymbol{c})$, the 4th being the object to be classified in one of the 2 classes, i.e. $cl(\boldsymbol{d}) \in \mathbb{B}$ is unknown, this principle can be stated as:

$$\frac{\forall i \in [1, n], a_i : b_i :: c_i : d_i = 1}{cl(\boldsymbol{a}) : cl(\boldsymbol{b}) :: cl(\boldsymbol{c}) : cl(\boldsymbol{d}) = 1}$$

Then, if the equation $cl(\boldsymbol{a}) : cl(\boldsymbol{b}) :: cl(\boldsymbol{c}) : x = 1$ is solvable, we can allocate its solution to $cl(\boldsymbol{d})$. This principle can lead to diverse implementations; see next section. The case of attributes on discrete domains and of a number of classes larger than 2 can be handled as easily as the binary case. Indeed, consider a finite attribute domain $\{v_1, \cdots, v_m\}$. Note that the attribute may also be the *class itself*. This attribute (or the class), say $\mathcal{A}$, can be straightforwardly binarized by means of the m properties "having value v_i, or not". Consider the partial description of objects $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{c}$, and $\boldsymbol{d}$ wrt $\mathcal{A}$. Assume, for instance, that objects $\boldsymbol{a}$ and $\boldsymbol{c}$ have value v_1, while objects $\boldsymbol{b}$ and $\boldsymbol{d}$ have value v_2. This situation is summarized in Table 2 where the respective truth-values of the four objects wrt each binary property "having value v_i" are indicated. As can be seen on this table, an analogical proportion holds true between the four objects for each binary property, and in the example, can be more compactly encoded as an analogical proportion between the attribute values themselves, namely here: $v_1 : v_2 :: v_1 : v_2$. More generally, x and y denoting possible values of a considered attribute $\mathcal{A}$, the analogical proportion between objects $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{c}$, and $\boldsymbol{d}$ holds for $\mathcal{A}$ iff the 4-tuple $(\mathcal{A}(\boldsymbol{a}), \mathcal{A}(\boldsymbol{b}), \mathcal{A}(\boldsymbol{c}), \mathcal{A}(\boldsymbol{d}))$ is equal to one 4-tuple having one of the three forms (s, s, s, s), (s, t, s, t), or (s, s, t, t).

Table 2. Handling non binary attributes

	v_1	v_2	v_3	$\cdots$	v_m	
$\boldsymbol{a}$	1	0	0	$\cdots$	0	v_1
$\boldsymbol{b}$	0	1	0	$\cdots$	0	v_2
$\boldsymbol{c}$	1	0	0	$\cdots$	0	v_1
$\boldsymbol{d}$	0	1	0	$\cdots$	0	v_2

A training set TS of examples $\boldsymbol{x^k} = ({x^k}_1, ..., {x^k}_i, ..., {x^k}_n)$ together with their class $cl(\boldsymbol{x^k})$, with $k = 1, t$ may also be read in an analogical proportion style: "$\boldsymbol{x^1}$ is to $cl(\boldsymbol{x^1})$ as $\boldsymbol{x^2}$ is to $cl(\boldsymbol{x^2})$ as $\cdots$ as $\boldsymbol{x^t}$ is to $cl(\boldsymbol{x^t})$". However note that $\boldsymbol{x^k}$ and $cl(\boldsymbol{x^t})$ are vectors of different dimensions. This may still be written (abusively) as $\boldsymbol{x^1} : cl(\boldsymbol{x^1}) :: \boldsymbol{x^2} : cl(\boldsymbol{x^2}) :: \cdots :: \boldsymbol{x^t} : cl(\boldsymbol{x^t})$. Note that this view exactly fits with the idea that in a classification problem there exists a classification *function* that associates a unique class with each object, which is unknown, but exemplified by the training set. Indeed $\boldsymbol{x^k} : cl(\boldsymbol{x^k}) :: \boldsymbol{x^k} : cl'(\boldsymbol{x^k})$ with $cl(\boldsymbol{x^k}) \neq cl'(\boldsymbol{x^k})$ is forbidden, since it cannot hold as a generalized analogical proportion obeying to a pattern of the form (s, t, s, t) where s and t belong to different spaces.

Postulating the central permutation property, the informal analogical proportion $\boldsymbol{x^i} : cl(\boldsymbol{x^i}) :: \boldsymbol{x^j} : cl(\boldsymbol{x^j})$ linking examples $\boldsymbol{x^i}$ and $\boldsymbol{x^j}$ can also be rewritten as $\boldsymbol{x^i} : \boldsymbol{x^j} :: cl(\boldsymbol{x^i}) : cl(\boldsymbol{x^j})$ (still informally as we deal with vectors of different dimensions). This suggests a new reading of the training set, based on pairs. Namely, the ways vectors $\boldsymbol{x^i}$ and $\boldsymbol{x^j}$ are similar / dissimilar should be related to the identity or the difference of classes $cl(\boldsymbol{x^i})$ and $cl(\boldsymbol{x^j})$. Given a pair of vectors $\boldsymbol{x^i}$ and $\boldsymbol{x^j}$, one can compute the set of attributes $A(\boldsymbol{x^i}, \boldsymbol{x^j})$ where they agree (i.e. they are equal) and the set of attributes $D(\boldsymbol{x^i}, \boldsymbol{x^j})$ where they disagree (i.e. they are not equal). Suppose, we have in the training set TS, both the pair $(\boldsymbol{x^i}, \boldsymbol{x^j})$, and the example $\boldsymbol{x^k}$ which once paired with $\boldsymbol{x^0}$ has exactly the *same disagreement set* as $D(\boldsymbol{x^i}, \boldsymbol{x^j})$ and moreover *with the changes oriented in the same way*. Note that although $A(\boldsymbol{x^i}, \boldsymbol{x^j}) = A(\boldsymbol{x^k}, \boldsymbol{x^0})$, the 4 vectors are not everywhere equal on this subset of attributes. Then we have a perfect analogical proportion componentwise, between the 4 vectors (of the form (s, s, s, s) or (s, s, t, t) on the agreement part of the components, and of the form (s, t, s, t) on the disagreement set). Indeed, the above view straightforwardly extends from binary-valued attributes to attributes with finite domains. Thus, working with pairs, we can implicitly reconstitute 4-tuples of vectors that form an analogical proportion as in the triple-based brute force approach to classification. We now discuss the algorithmic aspects of this approach.

Analogical Classification: The Standard View

Before introducing the analogical classifiers, let us restate the classification problem. Let T be a data set where each vector $\boldsymbol{x} = (x_1, ..., x_i, ..., x_n) \in T$ is a set of n feature values representing a piece of data. Each vector $\boldsymbol{x}$ is assumed to belong to a unique class $cl(\boldsymbol{x}) \in C = \{c_1, ..., c_l\}$, where C is finite and covered through the data set (in the binary class case, $l = 2$). If we suppose that cl is known on a subset $TS \subset T$,

given a new vector $\boldsymbol{y} = (y_1, ..., y_i, ..., y_n) \notin TS$, the classification problem amounts to assign a plausible value $cl(\boldsymbol{y})$ on the basis of the examples stored in TS.

Learning by analogy, as developed in [1], is a lazy learning technique which uses a measure of *analogical dissimilarity* between 4 objects. It estimates how far 4 situations are from being in analogical proportion. Roughly speaking, the analogical dissimilarity ad between 4 Boolean values is the minimum number of bits that have to be switched to get a proper analogy. Thus $\texttt{ad}(1,0,1,0) = 0$, $\texttt{ad}(1,0,1,1) = 1$ and $\texttt{ad}(1,0,0,1) = 2$. Thus, $A(a,b,c,d)$ holds if and only if $\texttt{ad}(a,b,c,d) = 0$. Moreover ad differentiates two types of cases where analogy does not hold, namely the 8 cases with an odd number of 0 and an odd number of 1 among the 4 Boolean values, such as $\texttt{ad}(0,0,0,1) = 1$ or $\texttt{ad}(0,1,1,1) = 1$, and the two cases $\texttt{ad}(0,1,1,0) = \texttt{ad}(1,0,0,1) = 2$. When, instead of having 4 Boolean values, we deal with 4 Boolean vectors in $\mathbb{B}^n$, we add the ad evaluations componentwise to get the analogical dissimilarity between the 4 vectors, which leads to an integer belonging to the interval $[0, 2n]$. This number estimates how far the 4 vectors are from building, componentwise, a complete analogy. It is used in [1] in the implementation of a classification algorithm where the input is a set S of classified items, a new item $\boldsymbol{d}$ to be classified, and an integer k. It proceeds as follows:

Step 1: Compute the analogical dissimilarity ad between $\boldsymbol{d}$ and all the triples in S^3 that produce a solution for the class of $\boldsymbol{d}$.

Step 2: Sort these n triples by the increasing value of ad wrt with $\boldsymbol{d}$.

Step 3: Let p be the value of ad for the k-th triple, then find k' as being the greatest integer such that the k'-th triple has the value p.

Step 4: Solve the k' analogical equations on the label of the class. Take the winner of the k' votes and allocate this winner as the class of $\boldsymbol{d}$.

This approach provides remarkable results and, in several cases, outperforms the best known algorithms [4]. Another equivalent approach [6] does not use a dissimilarity measure but just applies the previous continuity principle, adding flexibility by allowing to have some components where analogy does not hold. A majority vote is still applied among the candidate voters. Any triple $\mathbf{a}, \mathbf{b}, \mathbf{c}$, such that the cardinal of the set $\{i \in [1,n] | A(a_i, b_i, c_i, d_i)$ holds and $A(cl(\mathbf{a}), cl(\mathbf{b}), cl(\mathbf{c}), cl(\mathbf{d}))$ is solvable$\}$ is maximal, belongs to the candidate voters.

Analogical Classification: A Rule-Based View

We claim here that analogical classifiers behave as if a set of rules was build inductively during a pre-processing stage. To support intuition, we use an example inspired from the Golf data set (UCI repository [3]). This data set involves 4 multiple-valued attributes:

1: Outlook: sunny or overcast or rainy. ; *2: Temperature: hot or mild or cool* ;
3: Humidity: high or normal. ; *4: Windy: true or false*.

Two labels are available: 'Yes' (play) or 'No' (don't play).

Main Assumptions. Starting from a finite set of examples, 2 main assumptions are made regarding the behavior of the function cl:

– Since the target relation cl is assumed to be a function, when 2 distinct vectors $\boldsymbol{x}$, $\boldsymbol{y}$ have different labels ($cl(\boldsymbol{x}) \neq cl(\boldsymbol{y})$), the cause of the label switch is to be found in the switches of the attributes that differ. Take $\boldsymbol{x}$ and $\boldsymbol{y}$ in the Golf data set, as:
$\boldsymbol{x} = (overcast, mild, high, false)$ and $cl(\boldsymbol{x}) = Yes$
$\boldsymbol{y} = (overcast, cool, normal, false)$ and $cl(\boldsymbol{y}) = No$
then the switch in attributes 2 and 3 is viewed as the cause of the 'Yes'-'No' switch.
– When 2 distinct $\boldsymbol{x}$ and $\boldsymbol{y}$ are such that $cl(\boldsymbol{x}) = cl(\boldsymbol{y})$, this means that cl does not preserve distinctness, i.e. cl is not injective. We may then consider that the label stability is linked to the particular value arrangement of the attributes that differ.

Patterns. Let us now formalize these ideas. Given 2 distinct vectors $\boldsymbol{x}$ and $\boldsymbol{y}$, they define a partition of $[1, n]$ as $A(\boldsymbol{x}, \boldsymbol{y}) = \{i \in [1, n] | x_i = y_i\}$ and $D(\boldsymbol{x}, \boldsymbol{y}) = [1, n] \backslash A(\boldsymbol{x}, \boldsymbol{y}) = \{i \in [1, n] | x_i \neq y_i\}$. Given $J \subseteq [1, n]$, let us denote $x|_J$ the subvector of x made of the x_j, $j \in J$. Obviously, $\boldsymbol{x}|_{A(\boldsymbol{x},\boldsymbol{y})} = \boldsymbol{y}|_{A(\boldsymbol{x},\boldsymbol{y})}$ and, in the binary case, when we know $\boldsymbol{x}|_{D(\boldsymbol{x},\boldsymbol{y})}$, we can compute $\boldsymbol{y}|_{D(\boldsymbol{x},\boldsymbol{y})}$. In the binary case, the pair $(\boldsymbol{x}, \boldsymbol{y})$ allows us to build up a *disagreement pattern* $Dis(\boldsymbol{x}, \boldsymbol{y})$ as a list of pairs $(value, index)$ where the 2 vectors differ. with $n = 6, \boldsymbol{x} = (1, 0, 1, 1, 0, 0), \boldsymbol{y} = (1, 1, 1, 0, 1, 0)$, $Dis(\boldsymbol{x}, \boldsymbol{y}) = (0_2, 1_4, 0_5)$. It is obvious that having a disagreement pattern $Dis(\boldsymbol{x}, \boldsymbol{y})$ and a vector $\boldsymbol{x}$ (resp. $\boldsymbol{y}$), we can get $\boldsymbol{y}$ (resp. $\boldsymbol{x}$). In the same way, the disagreement pattern $Dis(\boldsymbol{y}, \boldsymbol{x})$ is deducible from $Dis(\boldsymbol{x}, \boldsymbol{y})$. For the previous example, $Dis(\boldsymbol{y}, \boldsymbol{x}) = (1_2, 0_4, 1_5)$.

In the categorical case, the disagreement pattern is a bit more sophisticated as we have to store the changing values. Then the disagreement pattern $Dis(\boldsymbol{x}, \boldsymbol{y})$ becomes a list of triple $(value1, value2, index)$ where the 2 vectors differ, with $value1$ being the attribute value for x and $value2$ being the attribute value for y. For instance, with the previously described Golf dataset, for the pair of given examples $\boldsymbol{x}$ and $\boldsymbol{y}$, $Dis(\boldsymbol{x}, \boldsymbol{y})$ is $\{(mild, cool)_2, (high, normal)_3\}$. Then we have two situations:

1. $\boldsymbol{x}$ and $\boldsymbol{y}$ have different labels, i.e. $cl(\boldsymbol{x}) \neq cl(\boldsymbol{y})$. Their disagreement pattern $Dis(\boldsymbol{x}, \boldsymbol{y})$ is called a *change pattern*. Then $Dis(\boldsymbol{y}, \boldsymbol{x})$ is also a change pattern.
2. $\boldsymbol{x}$ and $\boldsymbol{y}$ have the same label $cl(\boldsymbol{x}) = cl(\boldsymbol{y})$. Their disagreement pattern $Dis(\boldsymbol{x}, \boldsymbol{y})$ is called a *no-change pattern*. Then $Dis(\boldsymbol{y}, \boldsymbol{x})$ is also a no-change pattern.

To build up a change (resp. no-change) pattern, we have to consider all the pairs $(\boldsymbol{x}, \boldsymbol{y})$ such that $cl(\boldsymbol{x}) \neq cl(\boldsymbol{y})$ (resp. such that $cl(\boldsymbol{x}) = cl(\boldsymbol{y})$). We then build 2 sets of patterns P_{ch} and P_{noch}, each time keeping only one of the 2 patterns $Dis(\boldsymbol{x}, \boldsymbol{y})$ and $Dis(\boldsymbol{y}, \boldsymbol{x})$ to avoid redundancy. As exemplified below, these 2 sets are not disjoint in general. Take $n = 6$, and assume we have the 4 binary vectors $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \boldsymbol{t}$ in TS:

- $\boldsymbol{x} = (1, 0, 1, 1, 0, 0), \boldsymbol{y} = (1, 1, 1, 0, 1, 0)$ with $cl(\boldsymbol{x}) = 1$ and $cl(\boldsymbol{y}) = 0$. Then, for $(\boldsymbol{x}, \boldsymbol{y})$, the disagreement pattern is a change pattern, i.e., $(0_2, 1_4, 0_5) \in P_{ch}$.

- $\boldsymbol{z} = (0, 0, 1, 1, 0, 1), \boldsymbol{t} = (0, 1, 1, 0, 1, 1)$ with $cl(\boldsymbol{z}) = cl(\boldsymbol{t})$. They have the same disagreement pattern as $\boldsymbol{x}$ and $\boldsymbol{y}$, which is now a no-change pattern $(0_2, 1_4, 0_5) \in P_{noch}$.

Now, given an element $\boldsymbol{x}$ in TS whose label is known, and a new element to be classified $\boldsymbol{y}$, if the disagreement pattern $Dis(x, y)$ belongs to $P_{ch} \cap P_{noch}$, we do not get any hint regarding the label of $\boldsymbol{y}$. Then we remove the patterns in $P_{ch} \cap P_{noch}$: the remaining patterns are the *valid patterns* (still keeping the same notations for the resulting sets).

Rules. Thanks to the concept of pattern, it is an easy game to provide a formal definition of the 2 above principles. We get 2 general classification rules, corresponding to dual situations, for a new element $\boldsymbol{y}$ to be classified:

$$\text{Change Rule: } \frac{\exists \boldsymbol{x} \in TS, \exists D \in P_{ch} | (Dis(\boldsymbol{x}, \boldsymbol{y}) = D) \vee (Dis(\boldsymbol{y}, \boldsymbol{x}) = D)}{cl(\boldsymbol{y}) \neq cl(\boldsymbol{x})}$$

$$\text{NoChange Rule: } \frac{\exists \boldsymbol{x} \in TS, \exists D \in P_{noch} | (Dis(\boldsymbol{x}, \boldsymbol{y}) = D) \vee (Dis(\boldsymbol{y}, \boldsymbol{x}) = D)}{cl(\boldsymbol{y}) = cl(\boldsymbol{x})}$$

NoChange rules tell us when a new item y to be classified should get the class of its associated example x, and Change rules tell the opposite. Let us note that if there is no valid pattern, then we cannot build up any rule, then we cannot predict anything! This has never been the case for the considered benchmarks.

Implementation

It is straightforward to implement the previous ideas.

1. Construct from TS the sets P_{ch} and P_{noch} of all disagreement patterns.
2. Remove from P_{ch} and from P_{noch} the patterns belonging to $P_{ch} \cap P_{noch}$ to get the set of valid patterns.

The remaining change patterns in P_{ch} and no-change patterns in P_{noch} are used to build up respectively the *Change Rule Set* R_{ch} and *No-Change Rule Set* R_{noch}. In this context, we have implemented two different classifiers: the *Change Rule based Classifier* (CRC) and the *No Change Rule based Classifier* ($NCRC$), which have the same principles in all respect. The only difference is in the classification phase where the CRC only uses the set P_{ch} of pattern and applies the Change rules, whereas the second classifier $NCRC$ uses the no-change patterns P_{noch} and applies the No-Change rules to classify new items.

Classification. The classification process for CRC and $NCRC$ are detailed in the following algorithms 1 and 2, where the Boolean function $Analogy(x, x', y)$ is true if and only if $card(\{cl(x), cl(x'), cl(y)\}) \leq 2$. For the NCRC, the $Analogy(x, x', y)$ always has a solution since classes associated to any No-Change rule r in R_{noch} are homogeneous. In terms of complexity, the algorithms are still cubic in the size of TS since the disagreement pattern sets have a maximum of n^2 elements and we still have to check every element of TS to build up a relevant pair with $\boldsymbol{y}$.

With our approach, contrary to k-nn approaches, we always deal with pairs of examples: i) to build up the rules, ii) to classify a new item, we just associate to this item another one to build a pair in order to trigger a rule. Moreover, the two pairs of items involved in an analogical proportion are not necessarily much similar as pairs, beyond the fact they should exhibit the same dissimilarity. An analogical view of the nearest neighbor principle could be "close/far instances are likely to have the same/possibly different class", making an assumption that the similarity of the classes is related to the similarity of the instances. This does not fit, e.g., our No-Change rules where the similarity of the classes is associated with dissimilarities of the instances. More generally, while

Algorithm 1. Change Rule Classifier

Given a new instance $\boldsymbol{y'} \notin TS$ to be classified.
$CandidateRules(c_j) = 0$, for each $j \in [1, l]$ (in the binary class case, $l = 2$).
for each $\boldsymbol{y}$ in TS **do**
  Construct the disagreement patterns $D(\boldsymbol{y}, \boldsymbol{y'})$ and $D(\boldsymbol{y'}, \boldsymbol{y})$
  for each change rule $r \in R_{ch}$ // r has a pattern $D(\boldsymbol{x}, \boldsymbol{x'})$ **do**
    if $Analogy(x, x', y)$ AND ($D(\boldsymbol{y}, \boldsymbol{y'}) = D(\boldsymbol{x}, \boldsymbol{x'})$ OR $D(\boldsymbol{y'}, \boldsymbol{y}) = D(\boldsymbol{x}, \boldsymbol{x'})$) **then**
      if $(cl(x) = cl(y))$ **then** $c^* = cl(x')$ **else** $c^* = cl(x)$ **end if**
      $CandidateRules(c^*) + +$.
    end if
  end for
end for
$cl(y') = arg\max_{c_j} CandidateRules(c_j)$

Algorithm 2. No Change Rule Classifier

Given a new instance $\boldsymbol{y'} \notin TS$ to be classified.
$CandidateRules(c_j) = 0$, for each $j \in Dom(c_j)$.
for each $\boldsymbol{y}$ in TS **do**
  Construct the disagreement patterns $D(\boldsymbol{y}, \boldsymbol{y'})$ and $D(\boldsymbol{y'}, \boldsymbol{y})$
  for each no change rule $r \in R_{noch}$ // r has a pattern $D(\boldsymbol{x}, \boldsymbol{x'})$ **do**
    if $Analogy(x, x', y)$ AND ($D(\boldsymbol{y}, \boldsymbol{y'}) = D(\boldsymbol{x}, \boldsymbol{x'})$ OR $D(\boldsymbol{y'}, \boldsymbol{y}) = D(\boldsymbol{x}, \boldsymbol{x'})$) **then**
      $c^* = cl(y)$
      $CandidateRules(c^*) + +$.
    end if
  end for
end for
$cl(y') = arg\max_{c_j} CandidateRules(c_j)$

k-nn-like classifiers focus on the neighborhood of the target item, analogical classifiers "take inspiration" of information possibly far from the immediate neighborhood.

Example. Let's continue with the previous Golf example to show the classification process in Algorithm1. Given three change rules r_1, r_2 and r_3:

$$r_{1(Yes-No)} = \{(sunny, overcast)_1, (false, true)_4\}$$
$$r_{2(No-Yes)} = \{(cool, mild)_2, (high, normal)_3\}$$
$$r_{3(No-Yes)} = \{(rainy, overcast)_1, (false, true)_4\},$$

and a new instance y' to be classified: $y' : overcast, mild, normal, true, \rightarrow ?$
Assume that there are three training examples y_1, y_2 and y_3 in T_s:

$$y_1 : sunny, mild, normal, false, \rightarrow Yes$$
$$y_2 : overcast, cool, high, true, \rightarrow No$$
$$y_3 : rainy, mild, normal, false, \rightarrow No$$

We note that disagreement patterns p_1, p_2 and p_3 corresponding respectively to the pairs (y_1, y'), (y_2, y') and (y_3, y') match respectively the change rules r_1, r_2 and r_3. Inferring the first rule predict a first candidate class "No" for y'. In the same manner

the second rule predict a class "Yes" and the third one also predict "Yes". The rule-based inference produces the following set of candidate classes for y': $Candidate = \{No, Yes, Yes\}$. So the most plausible class for y' is "Yes".

Experimental Results and Comparison

This section provides experimental results for the two analogical proportion-based classifiers. The experimental study is based on several data sets selected from the U.C.I. machine learning repository [3]. A brief description of these data sets is given in Table 3. We note that for all classification results given in the following, only half of the

Table 3. Description of datasets

Datasets	Instances	Attributes	Classes
Breast cancer	286	9	2
Balance	625	4	3
Tic tac toe	958	9	2
Car	743	7	4
Monk1	432	6	2
Monk 2	432	6	2
Monk3	432	6	2

training set is used to extract patterns. We ensured that all class labels are represented in this data set. The classification results for the CRC or NCRC are summarized in the first and second columns of Table 4. We also tested a hybrid version of these classifiers called *Hybrid Analogical Classifier (HAC)* based on the following process. Given an instance y' to classify,

1. Merge the two rule subsets R_{ch} and R_{noch} into a single rule set R_{chnoch}.
2. Assign to y' the class label with the highest number of candidate rules in R_{chnoch}.

Classification results for HAC are given in Table 4, where we also give the mean number of Change (MeanCh) and No-Change rules (MeanNoCh) generated for each data set.

In order to compare analogical classifiers with other classification approaches, Table 5 includes classification results of some machine learning algorithms (the SVM, k-nearest neighbors IBK with k=10 and the propositional rule learner JRip) obtained by using the Weka software. By analyzing classification performance in Table 4 we can see that:

• Overall, the analogical classifiers show good performance to classify test examples (at least for one of CRC and NCRC), especially NCRC.

• If we compare classification results for the two analogical classifiers, CRC and NCRC, we see that NCRC seems to be more efficient than CRC for almost all data sets, except the case of "Tic tac toe" where the two classifiers have the same accuracy.

Table 4. Classification accuracies: mean and standard deviation of 10 cross-validations

Datasets	CRC	NCRC	HAC	MeanCh	MeanNoCh
Breast cancer	50.03 ± 8.03	**74.03±7.48**	73.39±8.44	6243.4	8738.5
Balance	82.82 ±5.8	**91.02±4.44**	90.51 ± 4.27	31736.2	20805.4
Tic tac toe	**98.3±5.11**	**98.3±5.11**	**98.3±5.11**	74391.9	86394.2
Car	79.54±4.23	**95.02± 2.16**	92.6 ±2.69	36526.6	20706.1
Monk1	90.52±6.16	**100±0**	99.54 ±1.4	9001.2	8644.6
Monk2	78.02 ±4.71	**100±0**	94.68 ± 4.38	7245.9	10607.8
Monk3	91.93±7.04	**97.93±1.91**	**97.93±1.91**	10588.0	10131.7

Table 5. Classification results of some known machine learning algorithms

Datasets	SVM	IBK(k=10)	JRip
Breast cancer	69.58	73.07	70.97
Balance	90.24	83.84	71.68
Tic tac	98.32	98.64	97.80
Car	91.65	91.92	87.88
Monk1	75.0	95.60	94.44
Monk2	67.12	62.96	66.43
Monk3	100	98.37	98.61

• HAC shows good performance if compared to CRC and very close accuracies to NCRC for "Balance, Tic tac toe, Monk1 and Monk3". For the remaining datasets, the lower classification accuracy of Change rules may affect the efficiency of HAC.

• In general, analogical classifiers (especially NCRC) show very good performance when compared to some of existing algorithms. NCRC significantly outperforms all other classifiers for all tested data sets (bold results in Table 5) except to some extent for "Monk3" and SVM. We see that NCRC is largely better than other classifiers, in particular for data sets "Monk1", "Monk2" and "Car".

• The classification success of NCRC for "Monks" datasets with noisy data and "Balance" and "Car" (which have multiple classes) demonstrates its ability to deal with noisy and multiple class data sets.

• The analogy-based classifiers seem to be very efficient when classifying data sets with a limited number of attribute values and seems to have more difficulties for classifying data sets with a large number of attribute values. In order to evaluate analogical classifiers such a dataset, we tested CRC and NCRC on "Cancer" (9 attributes, each of them having 10 different labels). From this additional test, we note that analogical classifiers are significantly less efficient on "Cancer" when compared to the state of the art algorithms. By contrast, if we look at the 3 "Monks" and "Balance" data sets, we note that these data sets have a smaller number of attributes and more importantly all attributes have a reduced number of possible values (the maximum number of possible attribute values in "Balance" and "Monks" is 5, and most of attributes have only 3 possible labels). This clearly departs from the "Cancer" situation. So we may say that this latter dataset is closer to a data set with numerical rather than categorical data. The proposed

classifiers are basically designed for handling categorical attributes. We plan to extend analogical rule-based classifiers in order to support numerical data in future.

• In Table 4 we see that a huge number of rules of the two kinds are generated. We may wonder if a reduced subset of rules could lead to the same accuracy. This would mean that there are some redundancy among each subset of rules, raising the question of how to detect it. We might even wonder if all the rules have the same "relevance", which may also mean that some rules have little value in terms of prediction, and should be identified and removed. This might also contribute to explain why CRC has results poorer than NCRC in most cases.

• In the case of NCRC, we come apparently close to the principle of a k-nn classifier, since we use nearest neighbors for voting, but here some nearest neighbors are disqualified because there is no NoChange rule (having the same disagreement pattern) that supports them.

Concluding Remarks

This paper has shown that analogical classification can rely on a rule-based technique, which contrasts with the existing implementations which are mainly lazy techniques. In the proposed approach, the rules are built at compile time, offline with respect to the classification process itself, where this set of rules is applied to new unclassified items in order to predict their class. This view brings new highlights in the understanding of analogical classification and may make this kind of learner more amenable to be mixed with logical ones like the ones coming from Inductive Logic Programming.

References

1. Bayoudh, S., Miclet, L., Delhay, A.: Learning by analogy: A classification rule for binary and nominal data. In: Proc. Inter. Conf. on Artificial Intelligence, IJCAI 2007, pp. 678–683 (2007)
2. Lepage, Y.: Analogy and formal languages. Electr. Notes Theor. Comput. Sci. 53 (2001)
3. Mertz, J., Murphy, P.: Uci repository of machine learning databases, `ftp://ftp.ics.uci.edu/pub/machine-learning-databases`
4. Miclet, L., Bayoudh, S., Delhay, A.: Analogical dissimilarity: definition, algorithms and two experiments in machine learning. JAIR 32, 793–824 (2008)
5. Miclet, L., Prade, H.: Handling analogical proportions in classical logic and fuzzy logics settings. In: Sossai, C., Chemello, G. (eds.) ECSQARU 2009. LNCS, vol. 5590, pp. 638–650. Springer, Heidelberg (2009)
6. Prade, H., Richard, G.: Reasoning with logical proportions. In: Lin, F.Z., Sattler, U., Truszczynski, M. (eds.) Proc. 12th Int. Conf. on Principles of Knowledge Representation and Reasoning, KR 2010, Toronto, May 9-13, pp. 545–555. AAAI Press (2010)
7. Prade, H., Richard, G.: From analogical proportion to logical proportions. Logica Universalis 7(4), 441–505 (2013)
8. Stroppa, N., Yvon, F.: Du quatrième de proportion comme principe inductif: une proposition et son application à l'apprentissage de la morphologie. Traitement Automatique des Langues 47(2), 1–27 (2006)