Representing and Learning Variations

Fadi Badra
INSERM, U1142, LIMICS, F-75006, Paris, France;
Sorbonne Universités, UPMC Univ Paris 06, UMR_S 1142, LIMICS, F-75006, Paris, France;
Université Paris 13, Sorbonne Paris Cité, LIMICS, (UMR_S 1142), F-93430, Villetaneuse, France.
badra@univ-paris13.fr

***Abstract*—In machine learning, objects are usually grouped according to similarities found in the objects descriptions. Recent works, however, suggest that representing the differences between object descriptions is also pertinent in many learning tasks. But not much study has been made on how to represent and learn from differences. This paper proposes a qualitative representation of inter-object variations that can be used as input of a learning task. The main idea is to define inter-objects variations as attributes of repetitions of objects, so that machine learning methods will be able to manipulate them in the same way as they manipulate object attributes. The approach is tested on both classification and a numerical value prediction tasks and shows encouraging results.**

***Keywords*-machine learning; knowledge representation; qualitative representation of variations;**

INTRODUCTION

Machine learning is a scientific field which purpose is to design computer programs that improve with experience. Traditionally, machine learning methods aim at grouping objects based on *similarities* found in the objects descriptions. However, some recent work on analogy [1], case-based reasoning [2], [3], or semantic relationship learning [4] seem to take an orthogonal approach: they take advantage of the *differences* between object descriptions to achieve a learning task. More precisely, these approaches manipulate explicit representations of some inter-object variations.

In the common language, the term variation usually expresses a difference between two or more states, be it a difference between species (genetic variation), a perturbation of the orbit (astronomical variation), an alteration of the form during a repetition (musical variation), or a measure of the codomain of a function or measure (mathematical variation). The notion of variation is thus closely related to the notion of difference. But *difference* is a complex concept to be defined. As noted by G. Bateson in [5], the concept of a difference is inherently relational:

> "*Obviously the difference between the paper and the wood is not in the paper; it is obviously not in the wood; it is obviously not in the space between them, and it is obviously not in the time between them. (Difference which occurs across time is what we call "change.")*"

Difference implies *repetition*, that is the reproduction of an object with an optional variation in its form. So difference (and thus, variation) somehow characterizes what happens between an object and its reproduction, i.e., between the different repetitions of a same object.

This paper studies how to represent a special kind of variation, that expresses some differences between the values taken by a property φ when this property is applied to describe different objects. Such a variation might express, e.g., that "black morels are edible whereas false morels aren't" (φ: mushroom edibility), or that the movie "*Gladiator* is more violent than *The Blair Witch Project*" (φ: degree of violence of a movie). To our knowledge, no unifying framework exist for the qualitative representation of variations. In [2] for example, the variation of a property φ between two elements x and y of a set $\mathcal{X}$ is represented by three properties φ^-, φ^+, and $\varphi^=$, to represent whether the property is lost, gained, or preserved from x to y. This representation is applied to learning adaptation knowledge for case-based reasoning. In [4], the goal is to learn semantic relationships such as "more violent than" from the Web. The elements of $\mathcal{X}$ are compared with respect to each property φ by introducing a total order on the set $\mathcal{X}$. Such an order results from the projection of the elements of $\mathcal{X}$ on a particular dimension of a conceptual space $\mathbb{R}^n$. Another approach is [3], which assumes the existence of a function $- : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$ that quantifies the difference between two elements x and y of $\mathcal{X}$. Taking the metaphor of differential calculus, the variation from x to y is then represented by the quantity $d_x = -(x, y) = y - x$. Differences are also represented in analogical reasoning: [1] represents the variations of a single property in extension, by the means of analogical proportions among objects of a set $\mathcal{X}$. Given four elements x, y, z, and t of $\mathcal{X}$, an analogical propertion models situations where "x is to y what z is to t" [6]. Another domain where differences have been explicitely represented for learning is pattern recognition [7]. In this domain, pairwise dissimilarities between objects are represented to compare structural objects. A dissimilarity representation is a mapping $\mathcal{X} \longrightarrow \mathbb{R}^n$ from a set of objects $\mathcal{X}$ to a dissimilarity space $\mathbb{R}^n$. This mapping is used to associate to each object a set of pairwise dissimilarities that represent the differences between this object and an object prototype. Finally, explicitely representing variations for a learning task might be very useful (though not much explored, to our knowledge) for gene expression data analysis, and in particular searching for biclusters with coherent evolutions [8]. Such biclusters represent co-evolutions of gene expression when experimental conditions vary.

In this paper, inter-objects variations are defined as attributes of repetitions of objects, so that machine learning methods can manipulate them as they manipulate object attributes. The proposed representation is applied to machine learning to perform classical tasks such as classification or numerical value prediction.

The paper is organized as follows. The next section studies how to represent variations between elements of a simple value space, such as $\mathbb{R}$, $\mathbb{B}$, or an enumerated set. Then, Sec. II shows how this representation can be used to represent variations between objects in machine learning. Sec. III defines variation learning, which is the task of learning such variations from a set of training examples. Sec. IV shows that that variation learning can be used to predict the value of a property for a given object. Sec. V gives implementation details, and then Sec. VI provides experimental results. The last section concludes the paper and gives future work.

I. Representing Variations

This section studies how to represent variations between elements of a simple value space.

A. Variations over Simple Value Spaces

Let $\mathcal{U}$ be a non-empty set of values. The set $\mathcal{U}$ is assumed to be either $\mathbb{R}$ (the set of real numbers), $\mathbb{Z}$ (the set of relative numbers), $\mathbb{B} = \{\texttt{true}, \texttt{false}\}$ (the set of Booleans), an enumerated set, e.g., $\{\texttt{red}, \texttt{green}, \texttt{blue}\}$, or the powerset 2^Ω of a set Ω.

The notion of repetition can be represented by an ordered collection of elements of $\mathcal{U}$. More formally:

Definition 1: Given a set $\mathcal{U}$ and an integer $n \geq 2$, a *repetition* $\mathbf{u}$ of elements of $\mathcal{U}$ is an element of the cartesian product $\mathcal{U}^n$.

To represent variations, the idea to view a variation as an attribute of a repetition of elements of $\mathcal{U}$. Therefore, a variation Δ is defined as a function that associates to a repetition $\mathbf{u} \in \mathcal{U}^n$ a value $\Delta(\mathbf{u})$ taken in a predefined set of values $\mathcal{V}$.

Definition 2: A *variation* Δ over a simple set of values $\mathcal{U}$ is a partial function $\Delta : \mathcal{U}^n \to \mathcal{V}$.

As an attribute, a variation can take different types of values, depending on the chosen set $\mathcal{V}$. If $\mathcal{V}$ is the set of Boolean values $\mathbb{B} = \{\texttt{true}, \texttt{false}\}$, the resulting variation Δ is a binary attribute. It is the indicator function of a n-ary relation r_Δ over $\mathcal{U}$. If $\mathcal{V}$ is the set of real numbers $\mathbb{R}$, the resulting variation is a real-value attribute. It associates real values to repetitions of elements of $\mathcal{U}$[1].

[1]Note that the given definition of a variation is quite general. Any distance function defined on $\mathcal{U}$ is a variation according to this definition.

B. Examples

For a set of values $\mathcal{U}$, a repetition $\mathbf{u}$ of elements of $\mathcal{U}$ is an element of the cartesian product $\mathcal{U}^n$. Therefore, a repetition $\mathbf{u}$ takes the form $\mathbf{u} = (u_1, u_2, \ldots, u_n)$ with $u_i \in \mathcal{U}$ for all i. For example, if $\mathcal{U}$ is the enumerated set $\mathcal{U} = \{\texttt{edible}, \texttt{poisonous}\}$, an example of repetition would be $\mathbf{u} = (\texttt{edible}, \texttt{poisonous})$. If $\mathcal{U} = \mathbb{Z}$, an example of repetition $\mathbf{u}$ of elements of $\mathcal{U}$ is $\mathbf{u} = (4, -8, 1, 1, 0)$.

Variations are functions $\Delta : \mathcal{U}^n \to \mathcal{V}$ that associate to a repetition $\mathbf{u} \in \mathcal{U}^n$ a value $\Delta(\mathbf{u})$ taken in a predefined set $\mathcal{V}$. If $\mathcal{U}$ is the enumerated set $\mathcal{U} = \{\texttt{edible}, \texttt{poisonous}\}$ and $\mathcal{V} = \mathbb{B}$, an example of (Boolean-value) variation is the variation $\texttt{edible}^=$ defined by:

$$\texttt{edible}^=(\mathbf{u}) = \begin{cases} \texttt{true} & \text{if } \forall i, u_i = \texttt{edible} \\ \texttt{false} & \text{otherwise} \end{cases}$$

Δ	$\Delta((u,v)) = \texttt{true}$ iff :
$\top_{\{\texttt{a,b,c}\}}$	always true
$=$	$u = v$
$\neq$	$u \neq v$
$\texttt{a}^=$	$u = v = \texttt{a}$
$\texttt{a} \to ?$	$u = \texttt{a}$
$? \to \texttt{b}$	$v = \texttt{b}$
$\texttt{a} \to \texttt{b}$	$u = \texttt{a}$ and $v = \texttt{b}$

Table I
Examples of variations when $\mathcal{U}$ is the enumerated set $\{\texttt{a}, \texttt{b}, \texttt{c}\}$.

Δ	$\Delta((u,v)) = \texttt{true}$ iff :
$\top_{\mathbb{Z}}$	always true
$=$	$u = v$
$\neq$	$u \neq v$
$\leq$	$u \leq v$
$<$	$u < v$
$\geq$	$u \geq v$
$>$	$u > v$

Table II
Examples of variations when $\mathcal{U}$ is the set $\mathbb{Z}$.

Table I gives some examples of Boolean-value variations that can be defined for $n = 2$ when $\mathcal{U}$ is an enumerated set such as $\{\texttt{a}, \texttt{b}, \texttt{c}\}$. The set of Booleans $\mathbb{B}$ can be considered as an enumerated set containing only two values $\texttt{true}$ and $\texttt{false}$. Table II gives some examples of Boolean-value variations that can be defined when $\mathcal{U}$ is the set of relative numbers $\mathbb{Z}$. Similar variations can be defined when $\mathcal{U}$ is the set of natural numbers $\mathbb{N}$ or the set of real numbers $\mathbb{R}$. Table III gives some examples of Boolean-value variations that can be defined when $\mathcal{U}$ is the powerset 2^Ω of a set Ω.

Δ	$\Delta((A,B)) = \texttt{true}$ iff :
$\top_{2^\Omega}$	always true
$\subseteq$	$A \subseteq B$
$=$	$A = B$
$\neq$	$A \neq B$
$\texttt{p}^=$	$\texttt{p} \in A$ and $\texttt{p} \in B$
$\texttt{p}^-$	$\texttt{p} \in A$ and $\texttt{p} \notin B$
$\texttt{p}^+$	$\texttt{p} \notin A$ and $\texttt{p} \in B$
$\texttt{p}^0$	$\texttt{p} \notin A$ and $\texttt{p} \notin B$

Table III
EXAMPLES OF VARIATIONS WHEN $\mathcal{U}$ IS THE SET 2^Ω.

When $\mathcal{V} = \mathbb{R}$, variations associate a real value to a repetition $\mathbf{u} \in \mathcal{U}^n$. An example of real-value variation that can be defined for $n = 2$ and $\mathcal{U} = \mathbb{R}$ is the variation $-$ (minus) that computes the difference between the two elements of a pair $\mathbf{u} = (u_1, u_2)$:

$$-(\mathbf{u}) = u_2 - u_1$$

So far, we have only represented variations between elements of a simple value space such as $\mathbb{B}$, $\mathbb{R}$, or an enumerate set of values. Can the same principles be applied to represent variations between objects in machine learning?

II. APPLICATION TO MACHINE LEARNING

This section studies how to represent variations between objects in machine learning.

A. Idea of the method

Machine learning methods usually start with a set of objects (also called items, or examplars), and attempt to find regularities in their descriptions. By learning variations between objects, we take an orthogonal approach to learning: instead of characterizing what in the description of an object makes it belong to a particular class, we want to characterize what in the differences between two (or more) objects makes these object belong to different classes. Similarly, we may want to learn what in the differences between two object descriptions makes the second object have a greater/lesser value for a property φ. To represent variations between objects, the idea is to consider a single property φ that transforms each object into a value in a simple value space, and then to reuse the variations previously defined between the values of a simple value space.

B. Definitions

Let $\mathcal{X}$ be a set of objects. A repetition $\mathbf{x}$ of elements of $\mathcal{X}$ is an element of the cartesian product $\mathcal{X}^n$. To define variations between objects, we'll assume the existence of a property φ that takes a value $\varphi(x)$ for some objects x of a set $\mathcal{X}$. The property φ can be seen as a partial function on $\mathcal{X}$ which associates to each element x of its domain Dom_φ (with $Dom_\varphi \subseteq \mathcal{X}$) an element $\varphi(x)$ of a set $\mathcal{U}$ (its codomain). A variation φ^Δ over a set of objects $\mathcal{X}$ is constructed from a pair (φ, Δ) consisting of a property φ together a variation between the values of the codomain $\mathcal{U}$ of φ.

Definition 3: Given a set $\mathcal{X}$, a partial function $\varphi : \mathcal{X} \to \mathcal{U}$ and a variation Δ over the set of values $\mathcal{U}$, a variation φ^Δ between objects is a function that associates to a repetition $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ of $\mathcal{X}^n$ the value:

$$\varphi^\Delta(\mathbf{x}) = \Delta((\varphi(x_1), \varphi(x_2), \ldots, \varphi(x_n)))$$

For example, if φ is a function that associates to each object an element of the enumerated set $\mathcal{U} = \{\texttt{edible}, \texttt{poisonous}\}$, and if $\Delta = \texttt{edible}^=$ defined in Sec. I-B, the variation $\varphi^{\texttt{edible}^=}$ is defined by:

$$\varphi^{\texttt{edible}^=}(\mathbf{x}) = \begin{cases} \texttt{true} & \text{if } \forall i, \varphi(x_i) = \texttt{edible} \\ \texttt{false} & \text{otherwise} \end{cases}$$

The following sections give examples of variations that can be defined when the objects are represented in an attribute-value formalism or in a logical language.

C. Variations in Attribute-Value Languages

An attribute-value language is interpreted on a cartesian product $\mathcal{U}_1 \times \ldots \times \mathcal{U}_m$, where each $\mathcal{U}_i$ is a set of values taken by an attribute. An attribute $\texttt{a}_i$ ($i \in \{1, 2, \ldots, m\}$) is a function defined on $\mathcal{X}$ and that returns the i^{th} projection on $\mathcal{U}_i$: for $x \in \mathcal{X}$, $\texttt{a}_i(x) = u_i \in \mathcal{U}_i$. When objects are represented in an attribute-value language, each attribute $\texttt{a}_i$ may constitute a property φ of interest from which we may want to define *per-attribute* variations. For example, if $\mathcal{U}_i = \mathbb{N}$ and $\texttt{a}_i = \texttt{age}$ represents the age of a person, a per-attribute variation $\texttt{age}^<$ could be defined by:

$$\texttt{age}^<((x_1, x_2)) = \begin{cases} \texttt{true} & \text{if } \texttt{age}(x_1) < \texttt{age}(x_2) \\ \texttt{false} & \text{otherwise} \end{cases}$$

The similarity evaluation process $(x_1, x_2) \mapsto d_x$ of [3] involves the computation of a set of real-value per-attribute variations. Each of these variations represents, for the repetition $\mathbf{x} = (x_1, x_2)$ the difference between the two values $d_i(x_1)$ and $d_i(x_2)$ taken on the i$^{\text{th}}$ problem descriptor d_i. The similarity d_x between problems is represented by a sequence $d_x = (d_{x_1}, \ldots, d_{x_n})$ of variations. The similarity d_y between solutions is represented in the same manner. Differential adaptation consists in computing d_y from d_x, i.e., in deriving a collection of variations from another.

The same kind of approach is used to represent dissimilarities in the pattern recognition domain. A set of real-value variations $\varphi^-((x_1, x_2)) = \varphi(x_2) - \varphi(x_1) \in \mathbb{R}$ are computed between an object x of $\mathcal{X}$ and a chosen prototype y.

D. Variations in Logic

Some variations can also be defined when the object representation language is a logical language with model-theoretic semantics. In such a formalism, one starts with a set

Ω (the domain of interpretation of formulas), and a function $\texttt{Mod} : \mathcal{L} \longrightarrow 2^{\Omega}$ that associates to each formula ϕ of the language $\mathcal{L}$ a subset $\texttt{Mod}(\phi)$ of Ω. An interpretation $\mathcal{I}$ is an element of Ω and a model of a formula ϕ is an interpretation such that $\mathcal{I} \in \texttt{Mod}(\phi)$. A formula ψ is a logical consequence of a formula ϕ, denoted $\phi \Vdash \psi$, iff $\texttt{Mod}(\phi) \subseteq \texttt{Mod}(\psi)$.

On such formalism, one can define variations between models by setting $\varphi = \texttt{Mod}$ and stating constraints on the models of the different formulas involved in a repetition $\mathbf{x}$. A simple example is the logical consequence relation $\Vdash$ that reflects the inclusion relation between the models of two formulas. The associated variation is $\texttt{Mod}^{\subseteq}$, which returns, for a repetition $\mathbf{x} = (\phi, \psi) \in \mathcal{L}^2$:

$$\texttt{Mod}^{\subseteq}((\phi,\psi)) = \begin{cases} \texttt{true} & \text{if } \texttt{Mod}(\phi) \subseteq \texttt{Mod}(\psi) \\ \texttt{false} & \text{otherwise} \end{cases}$$

In another example, inspired from [2], $\mathcal{L}$ is the language of propositional logic with m variables, so $\Omega = \mathbb{B}^m$, and $\varphi = \texttt{Mod}$. Each propositional variable p of $\mathcal{L}$ is interpreted as a singleton $\{\texttt{p}\} \in 2^{\Omega}$. For each element $\texttt{p} \in \Omega$, the four variations $\texttt{p}^-$, $\texttt{p}^=$, $\texttt{p}^+$, and $\texttt{p}^0$ defined in Tab. III are introduced. Using these variations, one can define e.g., the variation $\varphi^{\Delta} = \texttt{Mod}^{\texttt{p}^-}$ by:

$$\texttt{Mod}^{\texttt{p}^-}((\phi,\psi)) = \begin{cases} \texttt{true} & \text{if } \texttt{p} \in \texttt{Mod}(\phi) \text{ and } \texttt{p} \notin \texttt{Mod}(\psi) \\ \texttt{false} & \text{otherwise} \end{cases}$$

So if $\phi = \texttt{pie} \wedge \texttt{apple}$ and $\psi = \texttt{pie} \wedge \neg\, \texttt{apple}$, the variation $\texttt{Mod}^{\texttt{apple}^-}$ will associate the value $\texttt{true}$ to the repetition $\mathbf{x} = (\phi, \psi)$ (i.e., $\texttt{Mod}^{\texttt{apple}^-}((\phi,\psi)) = \texttt{true}$) since both $\phi \Vdash \texttt{apple}$ and $\psi \Vdash \neg\, \texttt{apple}$ hold.

These examples show that when objects are represented in an attribute-value or a logic formalism, inter-object variations can be represented as binary or real variables. Can such variations be learned from a set of training examples? For example, how can we learn from a set of film descriptions that "horror films are *more violent* than musicals"?

III. Learning Variations

This section introduces variation learning, an inductive learning task which goal is to learn a variation from a set of training examples.

A. Definition

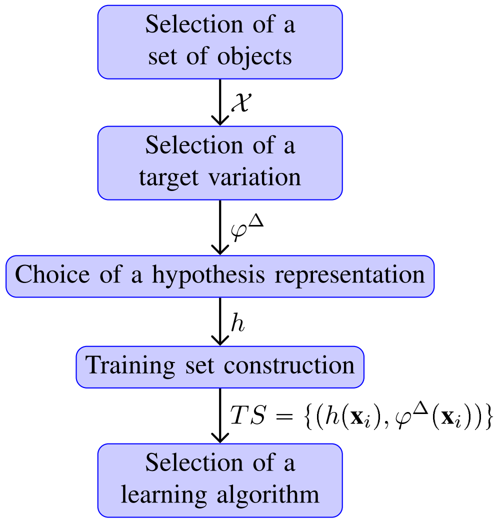
Inductive learning can be stated as the approximation of a target function f by a function f_H learned from a set of training examples. The *inductive learning hypothesis* consists in assuming that f_H will approximate f closely enough when the set of training examples is sufficiently large [9]. Variation learning is defined as an inductive learning task where the target function f is a variation between objects[2].

[2]As the target function $f = \varphi^{\Delta}$ is a function $f : \mathcal{X}^n \to \mathcal{V}$, it can be noted that this definition is consistent with the classical definition of concept learning, which would be obtained by setting $n = 1$ and $\mathcal{V} = \mathbb{B}$.

Definition 4: Variation learning consists in learning a variation φ^{Δ} between objects from a set of examples of its input and its output.

B. Overview of the Learning Process

The following diagram describes the main steps of a variation learning process:


Selection of a set of objects
$\mathcal{X}$
Selection of a target variation
φ^{Δ}
Choice of a hypothesis representation
h
Training set construction
$TS = \{(h(\mathbf{x}_i), \varphi^{\Delta}(\mathbf{x}_i))\}$
Selection of a learning algorithm

Selection of a set $\mathcal{X}$ of objects: The first step consists in selecting a set $\mathcal{X}$ of objects.

Selection of a target variation φ^{Δ}: A target variation φ^{Δ} is chosen by selecting a property φ of interest, and then selecting a variation Δ over the values of its codomain $\mathcal{U}$.

Choice of a hypothesis representation: A common approach to inductive learning is (i) to choose a hypothesis representation h, and then (ii) to search, in the space of hypothesis implied by this representation, for the hypothesis that best fits a set of training examples.

Training set construction: The training set $TS = \{(h(\mathbf{x}_i), \varphi^{\Delta}(\mathbf{x}_i))\}$ consists in a set of pairs $(h(\mathbf{x}_i), \varphi^{\Delta}(\mathbf{x}_i))$, where $\mathbf{x}_i$ is a repetition (i.e., an element of $\mathcal{X}^n$), $h(\mathbf{x}_i)$ is the hypothesis representation of $\mathbf{x}_i$, and $\varphi^{\Delta}(\mathbf{x}_i)$ is the image of $\mathbf{x}_i$ by the target variation φ^{Δ}.

Selection of a learning algorithm: If variations are used as the hypothesis representation language, they play the role that object attributes play in classical learning methods. Therefore, any classical learning algorithm can in principle be applied *as is* to mine the training set.

C. General-to-Specific Ordering of Hypothesis

Inductive learning algorithms often rely on the definition of a general-to-specific relation between hypothesis. A general-to-specific ordering of variations can be obtained whenever a Boolean-value function h_{Δ} can be associated to each variation Δ (and, by extension, to the corresponding inter-object variations φ^{Δ}). We would say that a variation Δ' is more general than a variation Δ, denoted by $h_{\Delta'} \geq_g h_{\Delta}$, iff $h_{\Delta}(x)$ is true whenever $h_{\Delta'}(x)$ is true. The simplest case is when Δ is a Boolean-value variation ($\mathcal{V} = \mathbb{B}$). In this case, it suffices to set $h_\Delta = \Delta$. For example, let us assume that the set $\mathcal{U} = \mathbb{R}$ represents car prices and consider the two variations $<: \mathcal{U}^2 \longrightarrow \mathbb{B}$ and $\neq: \mathcal{U}^2 \longrightarrow \mathbb{B}$ that represent respectively a (strict) price increase and price difference. The variations $<$ and $\neq$ can be ordered by generality ($\neq \geq_g <$) since their domain relations $r_<$ and $r_\neq$ verify $r_< \subseteq r_\neq$: a strict increase in price from a car to another implies a difference in price. When $\mathcal{V}$ is not $\mathbb{B}$, a way to obtain such a Boolean-value function h_Δ is to choose a subset $\mathcal{C}$ of $\mathcal{V}$ and to set h_Δ to be the indicator function $h_\Delta(u) = [\Delta(u) \in \mathcal{C}]$. Fig. 1 and Fig. 2 give some examples of general-to-specific orderings when $\mathcal{U}$ is the enumerated set $\{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ or the set of relative numbers $\mathbb{Z}$.

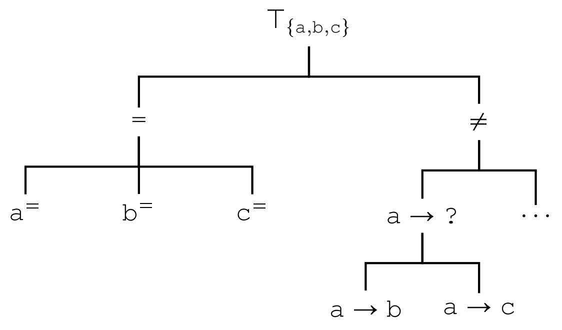


Figure 1. Examples of general-to-specific orderings between variations when $\mathcal{U}$ is the enumerated set $\{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$.

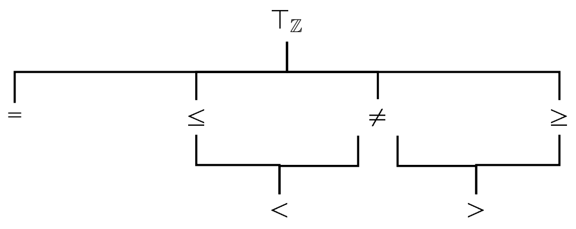


Figure 2. Examples of general-to-specific orderings between variations when $\mathcal{U}$ is the set of relative numbers $\mathbb{Z}$.

D. An Example

In this example, the objects represent students passing an exam and the goal is to be able to predict for two new students which one will have the highest mark.

Selection of a set $\mathcal{X}$ of objects: Let us assume that $\mathcal{X} = \{o_1, o_2, o_3\}$ contains three objects o_1, o_2, and o_3 that represent high-school students (Tab. IV). Each object is described by the values it takes for three attributes: the nominal attribute $\mathtt{s}$ ranges in $\{\mathtt{M}, \mathtt{F}\}$, and represents the student's <u>s</u>ex, and the two numeric attributes $\mathtt{a}$ and $\mathtt{m}$ measure respectively the student's <u>a</u>lcohol consumption, and the <u>m</u>ark obtained at an exam.

	sex (s)	alcohol (a)	mark (m)
o_1	M	2	14
o_2	F	4	6
o_3	F	1	18

Table IV
A SET $\mathcal{X}$ OF OBJECTS REPRESENTING HIGH-SCHOOL STUDENTS.

Selection of a target variation φ^Δ: The target variation is $\mathtt{m}^<$: the property of interest φ is the attribute $\mathtt{m}$, which gives the mark of a student at the exam, and the variation $\Delta =<$ is the function $\mathbb{Z} \times \mathbb{Z} \longrightarrow \mathbb{B}$ defined by $<((u_1, u_2)) = \mathtt{true}$ iff $u_1 < u_2$.

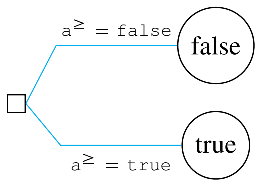
Choice of a hypothesis representation: A hypothesis $h(\mathbf{x}_{ij})$ is constructed for a repetition $\mathbf{x}_{ij} = (o_i, o_j)$ by computing the 9 variations $\mathtt{s}^{\mathtt{M}\to\mathtt{M}}$, $\mathtt{s}^{\mathtt{M}\to\mathtt{F}}$, $\mathtt{s}^{\mathtt{F}\to\mathtt{M}}$, $\mathtt{s}^{\mathtt{F}\to\mathtt{F}}$, $\mathtt{a}^<$, $\mathtt{a}^\leq$, $\mathtt{a}^=$, $\mathtt{a}^\neq$, $\mathtt{a}^\geq$, and $\mathtt{a}^>$. Each variation acts as a binary variable and represents some difference between o_i and o_j for the attribute $\mathtt{s}$ (sex) or the attribute $\mathtt{a}$ (alcohol). For example, the sex of o_1 is $\mathtt{M}$ (i.e., $\mathtt{s}(o_1) = \mathtt{M}$) and the sex of o_2 is $\mathtt{F}$ (i.e., $\mathtt{s}(o_2) = \mathtt{F}$) so $\mathtt{s}^{\mathtt{M}\to\mathtt{F}}(\mathbf{x}_{12}) = \mathtt{true}$.

Training set construction: All pairs (o_i, o_j) of $\mathcal{X}$ are included in the training set: TS contains all pairs $(h(\mathbf{x}_{ij}), \varphi^\Delta(\mathbf{x}_{ij}))$ such that $\mathbf{x}_{ij} = (o_i, o_j)$ and $\varphi^\Delta(\mathbf{x}_{ij}) = \mathtt{true}$ iff $\mathtt{m}(o_i) < \mathtt{m}(o_j)$.

The resulting dataset is (crosses represent the Boolean value $\mathtt{true}$):

	$\mathbf{x}_{11}$	$\mathbf{x}_{12}$	$\mathbf{x}_{13}$	$\mathbf{x}_{21}$	$\mathbf{x}_{22}$	$\mathbf{x}_{23}$	$\mathbf{x}_{31}$	$\mathbf{x}_{32}$	$\mathbf{x}_{33}$
$\mathtt{s}^{\mathtt{M}\to\mathtt{M}}$	×								
$\mathtt{s}^{\mathtt{M}\to\mathtt{F}}$		×	×						
$\mathtt{s}^{\mathtt{F}\to\mathtt{M}}$				×			×		
$\mathtt{s}^{\mathtt{F}\to\mathtt{F}}$					×	×		×	×
$\mathtt{a}^<$		×					×	×	
$\mathtt{a}^\leq$		×					×	×	
$\mathtt{a}^=$	×				×				×
$\mathtt{a}^\neq$		×	×	×		×	×	×	
$\mathtt{a}^\geq$			×	×		×			
$\mathtt{a}^>$			×	×		×			
$\mathtt{m}^<$			×	×		×			

Selection of a learning algorithm: This dataset is given as input to the learning algorithm Id3 [10], taking the attribute $\mathtt{m}^<$ as the class attribute. The following decision tree is obtained:



According to this model of the data, an increase in exam mark can be predicted whenever the alcohol consumption decreases.

IV. Variation-Based Property Prediction

This section shows how learned variations can be used to predict the value of the property φ for a given object.

A. *Position of the Problem*

For a new object y, the goal is to predict the value $\varphi(y)$ that a property φ takes in y.

$$\begin{tikzcd} \varphi(y)? \arrow[d] \\ y \end{tikzcd}$$

Depending on the codomain $\mathcal{U}$ of φ, two prediction problems are considered: class prediction (if $\mathcal{U}$ is an enumerated set), and numerical value prediction (if $\mathcal{U} = \mathbb{R}$).

B. *Idea of the Method*

The idea of the method is to reduce the property prediction problem to a (or say, many) variation prediction problem(s).

$$\begin{tikzcd} \varphi(x) \arrow[d] & \\ x \arrow[r, "\varphi^{\Delta}?"] & y \end{tikzcd}$$

Since $\varphi(x)$ is known for all $x \in \mathcal{X}$, the idea is to learn a well-chosen variation φ^{Δ}, so that $\varphi(y)$ can be deduced from φ^{Δ} and $\varphi(x)$. This operation is repeated for every $x \in \mathcal{X}$, in an ensemble learning approach, and a value is predicted for $\varphi(y)$ in a majority vote.

C. *An Algorithm for Class Prediction*

Let us assume that the set $\mathcal{X}$ is partitioned into two sets A and B (i.e., $\mathcal{X} = A \cup B$ and $A \cap B = \varnothing$). The goal is to predict, for a new object y, whether it belongs to A or B.

Choice of the property φ: Let us set $\mathcal{U} = \{\mathtt{A}, \mathtt{B}\}$ and define the function φ by:

$$\varphi(x) = \begin{cases} \mathtt{A} & \text{if } x \in A \\ \mathtt{B} & \text{if } x \in B \end{cases}$$

The problem now consists in predicting for a new object y whether $\varphi(y) = \mathtt{A}$ or $\varphi(y) = \mathtt{B}$ holds.

Choice of the variation Δ: Let us set $\Delta = \,?\to\mathtt{A}$. According to the definition of this variation (Tab. I), if $\varphi^{?\to\mathtt{A}}(\mathbf{x}) = \mathtt{true}$ for a repetition $\mathbf{x} = (x, y)$, then $\varphi(y) = \mathtt{A}$ holds, so $y \in A$. Otherwise, $y \in B$.

$$\begin{tikzcd} \varphi(x) \arrow[d] & \\ x \arrow[r, "\varphi^{?\to\mathtt{A}}?"] & y \end{tikzcd}$$

A simple variation-based class prediction algorithm, called **VC** (for **V**ariation-based **C**lassifier), is presented in Algo. 1. VC takes as input a learning algorithm cls, a new instance $y \in \mathcal{X}$ to be classified, and a hypothesis representation h. VC predicts a class (A or B) for y. All pairs $(x_i, x_j) \in \mathcal{X}^2$ are considered to construct the training set. For each of them, a repetition $\mathbf{x}_{ij} = (x_i, x_j)$ is formed and its associated hypothesis $h(\mathbf{x}_{ij})$ is computed. The classifier

Algorithm 1 VC

```
Input: $y \notin \mathcal{X}$: a new instance to be classified
Input: $cls$: a classifier
Input: $h$: a hypothesis representation
$vote_A \leftarrow 0$
$vote_B \leftarrow 0$
Form $TS = \{(h(\mathbf{x}_{ij}), \varphi^{?\to\mathtt{A}}(\mathbf{x}_{ij})) \mid \mathbf{x}_{ij} = (x_i, x_j) \in \mathcal{X}^2\}$
Train the learning algorithm $cls$ on $TS$
for all $x \in \mathcal{X}$ do
    Form $\mathbf{x} = (x, y)$
    Predict $\varphi^{?\to\mathtt{A}}(\mathbf{x})$ using $cls$
    if $\varphi^{?\to\mathtt{A}}(\mathbf{x}) = $ true then
        $vote_A \leftarrow vote_A + 1$
    else
        $vote_B \leftarrow vote_B + 1$
    end if
end for
return  $y \in A$ if $vote_A > vote_B$, else $y \in B$
```

cls is trained on a training set TS that includes all hypothesis $h(\mathbf{x}_{ij})$, along with their associated value $\varphi^{?\to\mathtt{A}}(\mathbf{x}_{ij})$ on the variation $\varphi^{?\to\mathtt{A}}$. The class of y is then predicted using an ensemble learning approach. For each instance $x \in \mathcal{X}$, the repetition $\mathbf{x} = (x, y)$ is formed, and cls is used to predict the value for $\varphi^{?\to\mathtt{A}}(\mathbf{x})$. The class of y is predicted using a majority vote.

D. *An Algorithm for Numerical Value Prediction*

The goal is to predict for a new object y a real value $\varphi(y)$.

Choice of the property φ: The property φ that we want to learn is a real-value function $\varphi : \mathcal{X} \longrightarrow \mathbb{R}$.

Choice of the variation Δ: Let us set $\Delta = -$ (minus), defined in Sec. I-B: $\varphi^{-}(\mathbf{x}) = \varphi(y) - \varphi(x)$ for a repetition $\mathbf{x} = (x, y)$, so for any $x \in \mathcal{X}$, $\varphi(y) = \varphi(x) + \varphi^{-}(\mathbf{x})$.

$$\begin{tikzcd} \varphi(x) \arrow[d] & \\ x \arrow[r, "\varphi^{-}?"] & y \end{tikzcd}$$

A variation-based numerical value prediction algorithm, called **VN** (for **V**ariation-based **N**umerical Value Prediction), is presented in Algo. 2. VN takes follows the same principles as VC, but uses the variation φ^{-} to predict a numerical value for each instance $x \in \mathcal{X}$. This value $\varphi^{-}(\mathbf{x})$ represents the difference $\varphi(y) - \varphi(x)$ between the value of the property φ in x and in y. The predicted value for $\varphi(y)$ is the average of these values.

V. Implementation

An implementation of the general representation method for variations was made in Scala, and the property value prediction methods were implemented using the Java API of the Weka package [11].

Algorithm 2 VN

```
Input: $y \notin \mathcal{X}$: a new instance to be classified
Input: $cls$: a classifier
Input: $h$: a hypothesis representation
$V \leftarrow \varnothing$
Form $TS = \{(h(\mathbf{x}_{ij}), \varphi^{-}(\mathbf{x}_{ij})) \mid \mathbf{x}_{ij} = (x_i, x_j) \in \mathcal{X}^2\}$
Train the learning algorithm $cls$ on $TS$
for all $x \in \mathcal{X}$ do
    Form $\mathbf{x} = (x, y)$
    Predict $\varphi^{-}(\mathbf{x})$ using $cls$
    Save estimation of $\varphi(y)$: $V \leftarrow V \cup \{\varphi(x) + \varphi^{-}(\mathbf{x})\}$
end for
return $\frac{1}{\|\mathcal{X}\|} \sum_{v \in V} v$
```

VI. Experiments

This section provides experimental results obtained with the VC and VN algorithms.

A. Class prediction

The VC algorithm was compared to other classifiers on five datasets from the UCI machine learning repository [12]: **MONK 1, 2,** and **3** problems ($MO.1$, $MO.2$, and $MO.3$), **Breast-W** ($Br.$), and **SPECT** ($SP.$) (Tab. V).

	$MO.1$	$MO.2$	$MO.3$	$Br.$	$SP.$
Instances	432	432	432	683	267
Classes	2	2	2	2	2
Nominal Att.	6	6	6	0	22
Numerical Att.	0	0	0	9	0
Nominal Var.	43	43	43	0	88
Numerical Var.	0	0	0	54	0

Table V
Description of datasets.

For each dataset, the hypothesis representation h is an attribute-value language. Each attribute of h is a variation of the form $\mathtt{a}_i^{\Delta}$, where $\mathtt{a}_i$ is one of the (binary or numerical) attributes of the source dataset and Δ is one of the variations defined in Sec. I-B (Tab. I and Tab. II). For example, each pair of instances of the Monks dataset will be represented using the nominal variations $\mathtt{a}_1{}^{=}$, $\mathtt{a}_1{}^{\neq}$, $\mathtt{a}_1{}^{1^{=}}$ $\mathtt{a}_1{}^{1\rightarrow 2}$, etc.

Table VI provides recognition rates obtained using a 5-fold cross validation, and for different classifiers cls: the decision tree algorithms Id3 and PART, the nearest-neighbor classifier IB1, the multi-layer perceptron (MLP), and the propositional rule learner JRip. Best results are highlighted in bold. These results can be compared to the ones obtained by the original classifiers on the same datasets (Table VII). Several comments arise.

- The results obtained by VC parameterized by a classifier cls is most of the time better than the results obtained with the classifier cls alone, with the exception of the JRip classifier.
- The choice of the classifier cls does not seem to have a great impact on the results. The recognition rate is very similar whatever the classifier used.
- Decision tree classifiers appear to be the best choice on any dataset.

	$MO.1$	$MO.2$	$MO.3$	$Br.$	$SP.$
VC(cls=Id3)	**97** $\pm$ 2	**66** $\pm$ 1	**100** $\pm$ 0	**96** $\pm$ 1	**83** $\pm$ 2
VC(cls=PART)	**96** $\pm$ 4	**67** $\pm$ 3	**100** $\pm$ 0	n/a	**84** $\pm$ 3
VC(cls=IB1)	**99** $\pm$ 1	63 $\pm$ 1	98 $\pm$ 3	89 $\pm$ 4	78 $\pm$ 4
VC(cls=MLP)	91 $\pm$ 13	**67** $\pm$ 3	99 $\pm$ 2	**96** $\pm$ 1	81 $\pm$ 2
VC(cls=JRip)	80 $\pm$ 15	**67** $\pm$ 3	97 $\pm$ 3	n/a	80 $\pm$ 2

Table VI
Recognition rate (means and standard deviation).

	$MO.1$	$MO.2$	$MO.3$	$Br.$	$SP.$
Id3	94	65	100	n/a	71
PART	100	68	100	96	79
IB1	72	61	78	96	79
MLP	100	100	100	95	79
JRip	95	65	99	96	81

Table VII
Comparison with well-known classifiers.

B. Numerical Value Prediction

The VN algorithm was applied to the task of predicting the mark of secondary students at a Maths exam.

The **Student-Mat** dataset [13] gives the results of secondary students at a Maths exam. The set $\mathcal{X}$ contains 395 objects, each of which represent a secondary student. Objects are described in an attribute-value formalism by 30 attributes. The attribute $\mathtt{G1}$ associates to each object a real value representing its mark at the Maths exam. Other attributes are either demographic (e.g., age, family size, mother's education), social (e.g., alcohol consumption, activities), or school-related (e.g., number of past failures, absences).

The target variation is $\varphi^{\Delta} = \mathtt{G1}^{-}$, which represents the variation in the student's exam mark: for $\mathbf{x}_{ij} = (o_i, o_j)$, $\mathtt{G1}^{-}(\mathbf{x}_{ij}) = \mathtt{G1}(o_j) - \mathtt{G1}(o_i)$.

As in the previous experiment, the hypothesis representation h is an attribute-value language. Each attribute of h is a variation of the form $\mathtt{a}_i^{\Delta}$, where $\mathtt{a}_i$ is one of the (binary or numerical) attributes of the source dataset and Δ is one of the variations defined in Sec. I-B (Tab. I and Tab. II). For example, each pair of instances of the dataset will be represented using the nominal variations $\mathtt{age}^{=}$, $\mathtt{age}^{\neq}$, $\mathtt{age}^{\leq}$, $\mathtt{age}^{<}$, etc. In the resulting training set TS, each repetition of objects $\mathbf{x}_{ij} = (o_i, o_j)$ is described by 196 variations: 78 for the 13 numerical attributes of the source dataset, and 118 for the 17 binary attributes.

Table VIII provides root mean squared error rates and standard deviations obtained using a 5-fold cross validation,

	Mean squared error	Std deviation
VN(cls=M5P)	2.48	0.13
VN(cls=LinearRegression)	2.50	0.08
VN(cls=IBk)	2.55	0.23
VN(cls=DecisionTable)	2.58	0.12
M5P	2.51	0.05
LinearRegression	2.53	0.05
IBk	3.18	0.18
DecisionTable	2.71	0.23

Table VIII
MEAN SQUARED ERROR AND STANDARD DEVIATIONS.

and for different classifiers cls: the model tree learner M5P, the k nearest neighbor algorithm IBk, linear regression (LinearRegression), and DecisionTable. Some comments can be made:

- When parameterized with a classifier cls, VN algorithms outperforms each time the classifier cls alone.
- The error rate improvement however, is often small.
- The choice of cls does not seem to have a great impact on the results.

C. Discussion

These first experiments show that it makes sense to learn inter-object variations instead of focusing on characterizing the objects themselves as it is done in most machine learning methods. However, the proposed algorithms are very naive and they suffer a high complexity, both spatial and temporal, which makes them inadequate for real-world scenarios. In particular, all pairs of objects of the source dataset are included in the training set, which quickly leads to an explosion of the training set size. On a current PC, we had memory limitations (n/a values in Tab. VI) when computing the training set for the BreastW dataset, which contains over 500 objects.

CONCLUSION AND FUTURE WORK

This paper studies how to represent inter-object variations for machine learning. A qualitative representation of inter-object variations was proposed. This representation was designed so that variations can be used as input of a learning task to learn what differenciates an object from another.

The main achievement of this study is that variations are represented in the same way that objects are usually described: a variation is simply an attribute of a repetition of objects, and can be manipulated by any learning algorithm in the same way that it manipulates object attributes. Two basic algorithms were proposed that apply variation learning to object classification and numerical value prediction. First results are encouraging: variation-based algorithms, when parameterized by a learning algorithm cls, generally perform better than the algorithm cls alone. However, the proposed algorithms are very naive and suffer major limitations: their complexity, both temporal and spatial, is very high.

Representing variations seems promising in many areas because it enables to symbolically represent dissimilarities between objects in a way that goes beyond computing a set of numerical values (as in pattern recognition) or a set of attribute coefficients (as in classical regression). It enables to represent co-variations and rules that could not be expressed otherwise (e.g., “the quantity of sugar increases when chocolate is replaced by cocoa in a cooking recipe”).

Future work includes applying the approach on a real-world scenario in the medical domain, finding optimizations of the proposed algorithms, and studying how to apply frequent itemset algorithms to mine a set of variations.

REFERENCES

[1] L. Miclet, S. Bayoudh, and A. Delhay, “Analogical Dissimilarity: Definition, Algorithms and Two Experiments in Machine Learning,” vol. 32, pp. 793–824, 2008.

[2] F. Badra, S. Lafrogne, J. Lieber, A. Napoli, and L. Szathmary, “Case Base Mining for Adaptation Knowledge Acquisition,” in *Proc. Int. Conf. Artif. Intell. IJCAI’07*, Hyderabad, India, 2007, pp. 750–755.

[3] B. Fuchs, J. Lieber, A. Mille, and A. Napoli, “Differential adaptation: An operational approach to adaptation for solving numerical problems with CBR,” *Knowledge-Based Syst.*, Apr. 2014.

[4] J. Derrac and S. Schockaert, “Characterising Semantic Relatedness using Interpretable Directions in Conceptual Spaces,” in *ECAI*, 2014.

[5] G. Bateson, *Steps to an Ecology of Mind*. Chandler Publishing Company, 1972, ch. Form, Substance, and Difference.

[6] H. Prade and G. Richard, “Reasoning with Logical Proportions,” in *Int. Conf. Princ. Knowl. Represent. Reason.*, 2010, pp. 545–555.

[7] R. P. W. Duin and E. Pçkalska, “The dissimilarity space: Bridging structural and statistical pattern recognition,” *Pattern Recognit. Lett.*, vol. 33, no. 7, pp. 826–832, 2012.

[8] S. C. Madeira and A. L. Oliveira, “Biclustering algorithms for biological data analysis: a survey.” *IEEE/ACM Trans. Comput. Biol. Bioinform.*, vol. 1, no. 1, pp. 24–45, 2004.

[9] T. M. Mitchell, *Machine Learning*. Elsevier, 1983.

[10] J. R. Quinlan, “Induction of Decision Trees,” *Expert Syst.*, pp. 81–106, 1986.

[11] M. Hall, H. National, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, “The WEKA Data Mining Software : An Update,” *SIGKDD Explor.*, vol. 11, pp. 10–18, 2009.

[12] M. Lichman, “UCI machine learning repository,” 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[13] P. Cortez and A. Silva, “Using Data Mining To Predict Secondary School Student Performance,” in *Proc. 5 th Annu. Futur. Bus. Technol. Conf.*, 2008, pp. 5–12.