Arguing and Explaining Classifications

Leila Amgoud
IRIT – CNRS
118, route de Narbonne
31062, Toulouse, France
amgoud@irit.fr

Mathieu Serrurier
IRIT – CNRS
118, route de Narbonne
31062, Toulouse, France
serrurier@irit.fr

ABSTRACT

Argumentation is a promising approach used by autonomous agents for reasoning about inconsistent knowledge, based on the construction and the comparison of arguments. In this paper, we apply this approach to the classification problem, whose purpose is to construct from a set of training examples a model (or hypothesis) that assigns a class to any new example.

We propose a general formal argumentation-based model that constructs arguments for/against each possible classification of an example, evaluates them, and determines among the conflicting arguments the acceptable ones. Finally, a "valid" classification of the example is suggested. Thus, not only the class of the example is given, but also the reasons behind that classification are provided to the user as well in a form that is easy to grasp.

We show that such an argumentation-based approach for classification offers other advantages, like for instance classifying examples even when the set of training examples is inconsistent, and considering more general preference relations between hypotheses. Moreover, we show that in the particular case of concept learning, the results of version space theory are retrieved in an elegant way in our argumentation framework.

Categories and Subject Descriptors

I.2.3 [**Deduction and Theorem Proving**]: Nonmonotonic reasoning and belief revision
; I.2.11 [**Distributed Artificial Intelligence**]: Intelligent agents

General Terms

Human Factors, Theory

Keywords

Argumentation, Classification

AAMAS'07 May 14–18 2007, Honolulu, Hawai'i, USA.

1. INTRODUCTION

Argumentation has become an Artificial Intelligence keyword for the last fifteen years, especially in sub-fields such as non monotonic reasoning, inconsistency-tolerant reasoning, multiple-source information systems [1, 7, 9, 3]. Argumentation follows basically three steps: i) to construct arguments and counter-arguments for a statement, ii) to select the "acceptable" ones and, finally, iii) to determine whether the statement can be accepted or not.

This paper claims that argumentation can also be used as an alternative approach for the problem of classification. Classification aims at building *models* that describe a *concept* from a set of *training examples*. The models are intended to be sufficiently general in order to be reused on new examples. When the concept to learn is binary, i.e. examples of that concept can be either true or false, the problem is called *concept learning*.

In our argumentation-based approach, the classification problem is reformulated as follows: given a set of examples (the training ones, and/or additional examples) and a set of hypotheses, what should be the class of a given example? To answer this question, arguments are constructed in favor of all the possible classifications of that example. A classification can come either from an hypothesis, or from a training example. The obtained arguments may be conflicting since it may be the case that the same example is affected to different classes. Finally, a "valid" classification of the example is suggested. Thus, not only the class of the example is given, but also the reasons behind that classification are provided to the user as well in a form that is easy to grasp.

We show that such an argumentation-based approach for classification offers other advantages, like for instance classifying examples even when the set of training examples is inconsistent, and considering more general preference relations between hypotheses. Moreover, we show that in the particular case of concept learning, the results of the version space theory developed by Mitchell in [4] are retrieved in an elegant way in our argumentation framework. We show that the acceptability semantics defined in [2] allow us to identify and characterize the version space as well as its lower and upper bounds. In sum, this paper proposes a formal theoretical framework for handling, analysing and explaining the problem of classification. The framework presents the following features that make it original and flexible:

1. it handles i) the case of a consistent set of training examples; ii) the case of an inconsistent set of training examples; and iii) the case of an empty set of training

examples;

2. it allows one to reason directly on the set of hypotheses;

3. examples are classified on the basis of the whole set of hypotheses rather than only one hypothesis as it is the case in standard classification models. Indeed, in the standard approach, a unique hypothesis is chosen, and all the examples are classified on the basis of that hypothesis.

4. it presents several original and intuitive decision criteria for choosing the class of an example.

The paper is organized as follows. We first present the classification problem, then we introduce the basic argumentation framework of Dung [2]. The third section introduces our argumentation-based framework for classification as well as its properties.

2. CLASSIFICATION PROBLEM

The aim of this section is to introduce the classification problem. Let $\mathcal{X}$ denote a *feature space* used for describing examples. Elements of $\mathcal{X}$ may then be pairs (attribute, value), first order facts, etc. This set $\mathcal{X}$ is supposed to be equipped with an equivalence relation $\equiv$. Let $\mathcal{C} = \{c_1, \ldots, c_n\}$ be a *concept space*, or a set of possible distinct classes.

A classification problem takes as input a *hypothesis space* $\mathcal{H}$, and a set $\mathcal{S}$ of m *training examples.*

$$\mathcal{S} = \{(x_i, c_i)_{i=1,\ldots,m} \text{ s.t } x_i \in \mathcal{X} \text{ and } c_i \in \mathcal{C}\}$$

An important notion in classification is that of consistency. In fact, a set of examples is said to be consistent if it does not contain two logically equivalent examples with two different classes. Formally:

DEFINITION 1 (CONSISTENCY). *Let $\mathcal{T} = \{(x_i, c_i)_{i=1,\ldots,n}$ s.t $x_i \in \mathcal{X}$ and $c_i \in \mathcal{C}\}$ be a set of examples. $\mathcal{T}$ is* consistent *iff $\nexists$ (x_1, c_1), $(x_2, c_2) \in \mathcal{T}$ such that $x_1 \equiv x_2$ and $c_1 \neq c_2$. Otherwise, $\mathcal{T}$ is said to be* inconsistent.

Regarding the *hypothesis space* $\mathcal{H}$, it may be, for instance, decision trees, propositional sets of rules, neural nets, etc. An *hypothesis* h is a mapping from $\mathcal{X}$ to $\mathcal{C}$ (i.e. h: $\mathcal{X} \mapsto \mathcal{C}$). Before defining the output of the framework, let us first introduce a key notion, that of *soundness.*

DEFINITION 2 (SOUNDNESS). *Let $h \in \mathcal{H}$. An hypothesis h is* sound *with respect to a training example $(x, c) \in \mathcal{S}$ iff $h(x) = c$. If $\forall (x_i, c_i) \in \mathcal{S}$, h is sound w.r.t (x_i, c_i), then h is said to be* sound *with $\mathcal{S}$.*

The general task of classification is to identify an $h \in \mathcal{H}$ that is sound with respect to the training examples. This hypothesis will be next used for classifying new examples. The most common approach for identifying this hypothesis is to use a greedy exploration of the hypothesis space, guided by a preference relation on hypothesis. This preference relation may be encoded by a utility function or by a syntactic or a semantic relation. Utility functions are generally based on the accuracy of the hypothesis (proportion of well classified examples) weighted by some complexity criteria (number of rules, etc.). Utility functions encode usually a total order. Syntactic relations may be for instance entailment or subsumption in the logical case. In this case it encodes a partial preorder on $\mathcal{H}$.

EXAMPLE 1 (LEARNING THE CONCEPT SUNNY DAY). *In this example, the feature space is a pair (attribute, value). Three attributes are considered: pressure, temperature, and humidity. The concept to learn is supposed to be binary, thus $\mathcal{C} = \{0, 1\}$. Four training examples are given, and are summarized in the Table below. For instance (pressure, low), (temperature, medium), and (humidity, high) is a negative example for the concept a sunny day, whereas the (pressure, medium), (temperature, medium), and (humidity, low) is a positive one.*

pressure	*temperature*	*humidity*	*sunny*
low	*medium*	*high*	*0*
medium	*medium*	*low*	*1*
low	*medium*	*medium*	*0*
medium	*high*	*medium*	*1*

Let us suppose that the hypothesis space $\mathcal{H}$ is the space of constraints *on the values of each attribute. Indeed, the constraints are conjunctions of accepted values of attributes. The special constraint $\emptyset$ (resp. ?) means that no (resp. all) values of attributes are accepted. If a vector of values of attributes match all the constraints, then it is considered as a* positive *example, otherwise it is a* negative *one. The hypotheses $\langle \emptyset, \emptyset, \emptyset \rangle$ and $\langle ?, ?, ? \rangle$ are respectively the lower and the upper bound of the hypothesis space $\mathcal{H}$.*

3. ABSTRACT ARGUMENTATION FRAMEWORK

Argumentation is a reasoning model that follows the following steps:

1. Constructing *arguments* and counter-arguments.

2. Defining the *strengths* of those arguments.

3. Evaluating the *acceptability* of the different arguments.

4. Concluding or defining the *justified conclusions.*

In [2], an argumentation system is defined as follows:

DEFINITION 3 (ARGUMENTATION SYSTEM). *An* argumentation system *(AS) is a pair $\langle \mathcal{A}, \mathcal{R} \rangle$. $\mathcal{A}$ is a set of arguments and $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}$ is a defeat relation. We say that an argument A defeats an argument B iff $(A, B) \in \mathcal{R}$ (or A $\mathcal{R}$ B).*

Note that to each argumentation system is associated an oriented graph whose nodes are the different arguments, and the edges represent the defeasibility relationship between them. Among all the conflicting arguments, it is important to know which arguments to keep for inferring conclusions or for making decisions. In [2], different semantics for the notion of acceptability have been proposed. Let us recall them here.

DEFINITION 4 (CONFLICT-FREE, DEFENCE). *Let $\mathcal{B} \subseteq \mathcal{A}$.*

- *$\mathcal{B}$ is* conflict-free *iff there exist no A_i, $A_j \in \mathcal{B}$ such that A_i $\mathcal{R}$ A_j.*

- $\mathcal{B}$ defends *an argument* A_i *iff for each argument* $A_j \in \mathcal{A}$, *if* $A_j \mathcal{R} A_i$, *then there exists* $A_k \in \mathcal{B}$ *such that* $A_k \mathcal{R} A_j$.

DEFINITION 5 (ACCEPTABILITY SEMANTICS). *Let* $\mathcal{B}$ *be a conflict-free set of arguments, and let* $\mathcal{F}$: $2^{\mathcal{A}} \mapsto 2^{\mathcal{A}}$ *be a function such that* $\mathcal{F}(\mathcal{B}) = \{A \mid \mathcal{B}$ *defends* $A\}$.

- $\mathcal{B}$ *is a* complete extension *iff* $\mathcal{B} = \mathcal{F}(\mathcal{B})$.
- $\mathcal{B}$ *is a* grounded extension *iff it is the minimal (w.r.t. set-inclusion) complete extension.*
- $\mathcal{B}$ *is a* preferred extension *iff it is a maximal (w.r.t. set-inclusion) complete extension.*
- $\mathcal{B}$ *is a* stable extension *iff it is a preferred extension that defeats all arguments in* $\mathcal{A} \backslash \mathcal{B}$.

Let $\{\mathcal{E}_1, \ldots, \mathcal{E}_n\}$ *be the set of all possible extensions under a given semantics.*

Note that there is only one grounded extension which may be empty. It contains all the arguments that are not defeated, and also the arguments which are defended directly or indirectly by non-defeated arguments.
The last step of an argumentation process consists of determining, among all the conclusions of the different arguments, the "good" ones, called *justified conclusions.*

4. AN ARGUMENTATION FRAMEWORK FOR CLASSIFICATION

The aim of this section is to propose an instantiation of the general and abstract framework of Dung that allows the classification of examples. Throughout this section, we will consider a features space $\mathcal{X}$, a concept space $\mathcal{C} = \{c_1, \ldots, c_n\}$, a (maybe *inconsistent*) set $\mathcal{S}$ of $m > 0$ training examples, a hypotheses space $\mathcal{H}$ that is equipped with an arbitrary preference relation $\succeq$. Thus, $\succeq \subseteq \mathcal{H} \times \mathcal{H}$, and $\mathcal{H}$ is supposed to be a *partial preorder.*

In order to instantiate the abstract framework of Dung, one needs to define the set $\mathcal{A}$ of arguments as well as the defeat relation between those arguments.

In our particular application, one needs to argue about particular classifications, thus arguments are constructed in favor of assigning particular classes from $\mathcal{C}$ to an example in $\mathcal{X}$. Indeed, an argument in favor of the pair (x, c) represents the reason for assigning the class c to the example x. Two reasons can be distinguished:

1. (x, c) is a training example in $\mathcal{S}$,
2. there exists a hypothesis $h \in \mathcal{H}$ that classifies x in c.

DEFINITION 6 (ARGUMENT). *An* argument *is a triplet* $A = \langle h, x, c\rangle$ *such that:*

1. $h \in \mathcal{H}$, $x \in \mathcal{X}$, $c \in \mathcal{C}$
2. *If* $h \neq \emptyset$, *then* $c = h(x)$
3. *If* $h = \emptyset$, *then* $(x, c) \in \mathcal{S}$

h *is called the* support *of the argument, and* (x, c) *its* conclusion. *Let* $\texttt{Example}(A) = x$, *and* $\texttt{Class}(A) = c$.
Let $\mathcal{A}$ *be the set of arguments built from* $(\mathcal{H}, \mathcal{X}, \mathcal{C})$.

Note that from the above definition, for any training example $(x_i, c_i) \in \mathcal{S}$, $\exists \langle \emptyset, x_i, c_i\rangle \in \mathcal{A}$. Let $\mathcal{A}_{\mathcal{S}} = \{\langle \emptyset, x, c\rangle \in \mathcal{A}\}$ (i.e. the set of arguments coming from the training examples). Since the set $\mathcal{S}$ of training examples is not empty, then $\mathcal{A}_{\mathcal{S}}$ is not empty as well.

PROPERTY 1. *Let* $\mathcal{S}$ *be a set of training examples.*

- $|\mathcal{S}| = |\mathcal{A}_{\mathcal{S}}|$[1].
- $\mathcal{A}_{\mathcal{S}} \neq \emptyset$.

PROOF. The first point follows from the above definition, and from the fact that an hypothesis h cannot be empty. The second point follows directly from the first property, i.e. $|\mathcal{S}| = |\mathcal{A}_{\mathcal{S}}|$, and the assumption that $\mathcal{S} \neq \emptyset$. □

Let us illustrate the notion of argument through example 1.

EXAMPLE 2. *In example 1, there are exactly four arguments with an empty support, and they correspond to the training examples:* $\mathcal{A}_\emptyset = \{a_1 = \langle \emptyset$, *(pressure, low)* $\wedge$ *(temperature, medium)* $\wedge$ *(humidity,high), 0*$\rangle$,
$a_2 = \langle \emptyset$, *(pressure, medium)* $\wedge$ *(temperature, medium)* $\wedge$ *(humidity, low), 1*$\rangle$,
$a_3 = \langle \emptyset$, *(pressure, low)* $\wedge$ *(temperature, medium)* $\wedge$ *(humidity, medium), 0*$\rangle$,
$a_4 = \langle \emptyset$, *(pressure, medium)* $\wedge$ *(temperature, high)* $\wedge$ *(humidity, medium), 1*$\rangle\}$. *There are also arguments with a non-empty support such as:*
$\langle a_5 = \langle$ *?* , *medium* $\vee$ *high, ?*$\rangle$, *(pressure, low)* $\wedge$ *(temperature, high)* $\wedge$ *(humidity, high), 1*$\rangle\}$,
$a_6 = \langle\langle$ *medium*$\vee$ *high, ?, ?*$\rangle$, *(pressure, low)* $\wedge$ *(temperature, high)* $\wedge$ *(humidity, high), 0*$\rangle$,
$a_7 = \langle\langle$ *medium, medium*$\vee$ *high, ?*$\rangle$, *(pressure, low)* $\wedge$ *(temperature, high)* $\wedge$ *(humidity, high), 0*$\rangle$.

In [1, 7, 9], it has been argued that arguments may have different strengths depending on the quality of information used to construct them. In [9], for instance, arguments built from specific information are stronger than arguments built from more general ones. In our particular application, it is clear that arguments with an empty support are stronger than arguments with a non-empty one. This reflects the fact that classifications given by training examples take precedence over ones given by hypotheses in $\mathcal{H}$. It is also natural to consider that arguments using most preferred hypothesis are stronger than arguments with less preferred ones.

DEFINITION 7 (COMPARING ARGUMENTS). *Let* $\langle h, x, c\rangle$, $\langle h', x', c'\rangle$ *be two arguments of* $\mathcal{A}$. $\langle h, x, c\rangle$ *is* preferred *to* $\langle h', x', c'\rangle$, *denoted by* $\langle h, x, c\rangle$ Pref $\langle h', x', c'\rangle$, *iff:*

- $h = \emptyset$ *and* $h' \neq \emptyset$, *or*
- $h \succeq h'$.

PROPERTY 2. *The relation Pref is a partial preorder.*

PROOF. This is due to the fact that the relation $\succeq$ is a partial preorder. □

Now that the set of arguments is defined, it is possible to define the defeasibility relation $\mathcal{R}$ between arguments in $\mathcal{A}$. Here again, there are two ways in which an argument A can attack another argument B:

[1] $|\cdot|$ denotes the cardinal of a given set

1. by *rebutting* its *conclusion*. This situation occurs when the two arguments have contradictory conclusions, i.e. the same example is classified in different ways.

2. by *undercutting* its *support*. This occurs when the support of B classifies in a different way the example of the conclusion of A. However, this relation is only restricted to training examples. Indeed, only arguments built from training examples are allowed to undercut other arguments. The idea behind this is that training examples are the only, in some sense, certain information one has, and thus cannot be defeated by hypothesis. However, hypothesis have controversial status.

DEFINITION 8 (REBUTTING). *Let $\langle h, x, c\rangle$, $\langle h', x', c'\rangle$ be two arguments of $\mathcal{A}$. $\langle h, x, c\rangle$* rebuts *$\langle h', x', c'\rangle$ iff $x \equiv x'$, $c \neq c'$.*

EXAMPLE 3. *In example 2, we have for instance :* a_5 rebuts a_6, a_5 rebuts a_7, a_6 rebuts a_5, *and* a_7 rebuts a_5.

DEFINITION 9 (UNDERCUTTING). *Let $\langle h, x, c\rangle$, $\langle h', x', c'\rangle$ be two arguments of $\mathcal{A}$. $\langle h, x, c\rangle$* undercuts *$\langle h', x', c'\rangle$ iff $h = \emptyset$ and $h'(x) \neq c$.*

EXAMPLE 4. *In example 2, we have for instance :* a_1 undercuts a_5, a_2 undercuts a_5, a_3 undercuts a_5, *and* a_4 undercuts a_5.

Note that the rebutting and undercutting relations are used in most argumentation systems that handle inconsistency in knowledge bases.

PROPERTY 3. *If $\mathcal{S}$ is consistent, then $\nexists$ A, $B \in \mathcal{A}_\mathcal{S}$ such that A rebuts B, or A undercuts B.*

PROOF. Let $A = \langle \emptyset, x, u\rangle$, $B = \langle \emptyset, x', u'\rangle \in \mathcal{S}$ such that A rebuts B. According to Definition 8, $x \equiv x'$ and $u \neq u'$. This contradicts the fact that $\mathcal{S}$ is consistent. □

The two above conflict relations are brought together in a unique relation, called "Defeat".

DEFINITION 10 (DEFEAT). *Let $A = \langle h, x, c\rangle$, $B = \langle h', x', c'\rangle$ be two arguments of $\mathcal{A}$. A* defeats *B iff:*

1. A rebuts (resp. undercuts) B, and

2. (A Pref B), or (not(A Pref B) and not(B Pref A))

EXAMPLE 5. *With the argument defined in ex. 2 we have for instance :* a_1 defeats a_5, a_2 defeats a_5, a_3 defeats a_5, a_4 defeats a_5, a_5 defeats a_6, a_5 defeats a_7 *and* a_6 defeats a_5.

From the above definition, it is easy to check that an argument with an empty-support cannot be defeated by an argument with a non-empty support.

PROPERTY 4. *$\forall$ $A \in \mathcal{A}_\mathcal{S}$, $\nexists$ $B \in \mathcal{A}\backslash\mathcal{A}_\mathcal{S}$ s.t B defeats A.*

PROOF. Let $A \in \mathcal{A}_\mathcal{S}$ and $B \in \mathcal{A}\backslash\mathcal{A}_\mathcal{S}$ such that B defeats A. This means that B rebuts A (because according to Definition 9, an argument with a non-empty support cannot undercut an argument with an empty one. Moreover, according to Definition 10, we have either B Pref A, or (not(B Pref A) and not(A Pref B)). This is impossible because according to Definition 7, arguments in $\mathcal{A}_\mathcal{S}$ are always preferred to arguments with a non-empty support. □

The argumentation system for classification is then the following:

DEFINITION 11 (ARGUMENTATION SYSTEM). *An* argumentation system *for classification (ASC) is a pair $\langle \mathcal{A}, defeat\rangle$, where $\mathcal{A}$ is the set of arguments (see Definition 6) and defeat is the relation defined in Definition 10.*

Let us now identify the acceptable arguments of the above ASC. It is clear that the arguments that are not defeated at all will be acceptable. Let $\mathcal{U}$ denote that set of undefeated arguments.

PROPOSITION 1. *If $\mathcal{S}$ is consistent, then $\mathcal{A}_\mathcal{S} \subseteq \mathcal{U}$.*

PROOF. Let $A \in \mathcal{A}_\mathcal{S}$. Let us assume that $\exists B \in \mathcal{A}$ such that B defeats A. According to Property 4, $B \notin \mathcal{A}\backslash\mathcal{A}_\mathcal{S}$. Thus, $B \in \mathcal{A}_\mathcal{S}$. Moreover, B defeats A means that B rebuts A. This means then that A classifies a training example in u, and B classifies an equivalent example in $u' \neq u$. This contradicts the fact that the set $\mathcal{S}$ is consistent. □

As said in Section 3, one of the acceptability semantics is the so-called 'grounded extension'. Such an extension is unique and maybe empty. However, we show that when the set $\mathcal{S}$ of training examples is consistent, this grounded extension is not empty.

PROPOSITION 2 (GROUNDED EXTENSION). *If $\mathcal{S}$ is consistent, then the argumentation system $\langle \mathcal{A}, defeat\rangle$ has a non empty grounded extension $\mathcal{E}$.*

PROOF. This is due to the fact that $\mathcal{A}_\mathcal{S} \neq \emptyset$ and $\mathcal{A}_\mathcal{S} \subseteq \mathcal{U}$. □

Note that the system $\langle \mathcal{A}, defeat\rangle$ is not always finite. By finite we mean that each argument is defeated by a finite number of arguments. This is due to the fact that $\mathcal{H}$ and $\mathcal{X}$ are not always finite.

PROPOSITION 3. *If $\mathcal{H}$ and $\mathcal{X}$ are finite, then the system $\langle \mathcal{A}, defeat\rangle$ is finite.*

When an argumentation system is finite, its characteristic function $\mathcal{F}$ is continuous. Consequently, the least fixed point of this function can be defined by an iterative application of $\mathcal{F}$ to the empty set.

PROPOSITION 4. *If the argumentation system $\langle \mathcal{A}, defeat\rangle$ is finite, then the grounded extension $\mathcal{E}$ is:*

$$\mathcal{E} = \bigcup \mathcal{F}^{i\geq 0}(\emptyset) = \mathcal{U} \cup [\bigcup_{i\geq 1} \mathcal{F}^i(\mathcal{U})].$$

Let us now analyze the other acceptability semantics, namely preferred and stable ones. In general, the ASC has at least one preferred extension that may be empty. However, as for the case of grounded extension, we can show that in the particular case of a consistent set of training examples, the ASC has at least one non-empty preferred extension.

PROPOSITION 5. *If $\mathcal{S}$ is consistent, then the ASC $\langle \mathcal{A}, defeat\rangle$ has $n \geq 1$ non-empty preferred extensions.*

PROOF. In [2], it has been shown that the grounded extension is included in very preferred extension. Since the grounded extension is not empty (according to Proposition 2, then there exists at least one non-empty preferred extension). □

In general, the preferred extensions of an argumentation system are not stable. However, we can show that when the set $\mathcal{C}$ contains only two possible classes, this means that the concept to learn is binary, these extensions coincide. This result is due to the fact that the oriented graph associated to the above ASC has no odd length circuits in this case. However, it may contain circuits of even length.

PROPOSITION 6. *If $\mathcal{C} = \{c_1, c_2\}$ with $c_1 \neq c_2$, then:*

- *The graph associated with the system $\langle \mathcal{A}, defeat \rangle$ has no odd length circuits.*
- *The preferred extensions and stable extensions of the system $\langle \mathcal{A}, defeat \rangle$ coincide.*

PROOF SKETCH. **Part 1:** Let A, B, C be three arguments such that A defeats B, B defeats C, and C defeats A.

Case 1: Let us suppose that $A \in \mathcal{A}_\mathcal{S}$.
According to Property 3, $B \in \mathcal{A} \backslash \mathcal{A}_\mathcal{S}$. According to Property 4, C should be in $\mathcal{A} \backslash \mathcal{A}_\mathcal{S}$. Contradiction because according to Property 4, C cannot defeat A, which is in $\mathcal{A}_\mathcal{S}$.

Case 2: Let us suppose that $A, B, C \in \mathcal{A} \backslash \mathcal{A}_\mathcal{S}$. This means that A rebuts B, B rebuts C, and C rebuts A (according to Definition 9). Consequently, $\texttt{Example}(A) \equiv \texttt{Example}(B) \equiv \texttt{Example}(C)$, and $\texttt{Value}(A) \neq \texttt{Value}(B)$, $\texttt{Value}(B) \neq \texttt{Value}(C)$. Due to the fact that $\mathcal{U} = \{0, 1\}$, we have $\texttt{Value}(A) = \texttt{Value}(C)$. This contradicts the assumption that C rebuts A.

Part 2: This is a consequence of the fact that there is no odd circuits in the system. □

Moreover, in this case the intersection of all the preferred (stable) extensions coincides with the grounded extension.

PROPOSITION 7. *Let $\langle \mathcal{A}, defeat \rangle$ be an ASC. Let $\mathcal{E}$ be its grounded extension, and $\mathcal{E}_1, \ldots, \mathcal{E}_n$ its preferred (stable) extensions. If $\mathcal{C} = \{c_1, c_2\}$ with $c_1 \neq c_2$, then $\mathcal{E} = \bigcap_{i=1,\ldots,n} \mathcal{E}_i$.*

The last step of an argumentation process consists of defining the *status* of the conclusions, in our case, the classification of examples. In what follows we present two decision criteria: The first one, called universal vote, consists of accepting those classifications that are in any extension. However, it is clear that this kind of voting may not classify all the examples. Thus, we propose a second criterion, called majority vote, that allows to associate a class with each example. The conclusions here are the ones that are supported by a majority of arguments that appear in the different extensions. Formally:

DEFINITION 12. *Let $\langle \mathcal{A}, defeat \rangle$ be a ASC, and $\mathcal{E}_1, \ldots, \mathcal{E}_n$ its extensions under a given semantics. Let $x \in \mathcal{X}$ and $c \in \mathcal{C}$.*

Universal vote: *x is* universally classified *in c iff $\forall \mathcal{E}_i$, $\exists <h, x, c> \in \mathcal{E}_i$. UV denotes the set of all (x, c), such that x is universally classified in c.*

Majority vote: *x is* majoritarily classified *in c iff $|\{< h, x, c > | \exists \mathcal{E}_i, < h, x, c > \in \mathcal{E}_i\}| \geq |\{< h, x, c' > | c' \neq c, \exists \mathcal{E}_i, < h, x, c' > \in \mathcal{E}_i\}|$. MV denotes the set of all (x, c), such that x is majoritarily classified in c.*

The universally classified examples are those that are supported by arguments in all the extensions. From a classification point of view, these correspond to examples classified by the most preferred hypotheses. It is easy to check that the set of universally classified examples is included in the set of majoritarily classified ones.

PROPERTY 5. *Let $\langle \mathcal{A}, defeat \rangle$ be a ASC, and $\mathcal{E}_1, \ldots, \mathcal{E}_n$ its extensions under a given semantics :*

$$UV \subseteq MV$$

We can show that the above argumentation framework delivers "safe" results, since its sets of conclusions UV, MV are consistent. Formally:

PROPOSITION 8. *Let $\langle \mathcal{A}, defeat \rangle$ be a ASC, and UV, MV its sets of conclusions. The sets UV and MV are consistent.*

5. RETRIEVING VERSION SPACE THEORY

As said before, *concept learning* is a particular case of classification, where the concept to learn is binary. In [4], Mitchell has proposed the famous general and abstract framework, called *version space learning*, for concept learning. That framework takes as input a *consistent* set of *training examples* on the concept to learn. $\mathcal{C}$ contains only two classes, denoted respectively by 0 and 1. Thus, $\mathcal{C} = \{0, 1\}$. The set $\mathcal{H}$ is supposed to be equipped with a "particular" *partial preorder* $\succeq$ that reflects the idea that some hypothesis are more general than others in the sense that they classify positively more examples. This preorder defines a lattice on the hypothesis space. Formally:

DEFINITION 13 (GENERALITY ORDER ON HYPOTHESIS). *Let h_1, $h_2 \in \mathcal{H}$. h_1 is* more general *than h_2, denoted by $h_1 \succeq h_2$, iff $\{x \in \mathcal{X} | h_1(x) = 1\} \supseteq \{x \in \mathcal{X} | h_2(x) = 1\}$.*

The framework identifies the *version space*, which is the set $\mathcal{V}$ of all the hypothesis of $\mathcal{H}$ that are sound with $\mathcal{S}$. The idea is that a "good" hypothesis should at least classify the training examples correctly.

DEFINITION 14 (VERSION SPACE).

$$\mathcal{V} = \{h \in \mathcal{H} |\ h \textit{ is sound with } \mathcal{S}\}$$

Version space learning aims at identifying the *upper* and the *lower* bounds of this version space $\mathcal{V}$. The upper bound will contain the most general hypothesis, i.e the ones that classify more examples, whereas the lower bound will contain the most specific ones, i.e the hypothesis that classify less examples.

DEFINITION 15 (GENERAL HYPOTHESES). *The set of* general hypothesis *is $\mathcal{V}_G = \{h \in \mathcal{H} \mid h$ is sound with $\mathcal{S}$ and $\nexists$ $h' \in \mathcal{H}$ with h' sound with $\mathcal{S}$, and $h' \succeq h\}$.*

DEFINITION 16 (SPECIFIC HYPOTHESES). *The set of* specific hypothesis *is $\mathcal{V}_S = \{h \in \mathcal{H} \mid h$ is sound with $\mathcal{S}$ and $\nexists$ $h' \in \mathcal{H}$ with h' sound with $\mathcal{S}$, and $h \succeq h'\}$.*

From the above definition, we have the following simple property characterizing the elements of $\mathcal{V}$.

PROPERTY 6. *[4]*

$$\mathcal{V} = \{h \in \mathcal{H} | \exists h_1 \in \mathcal{V}_S, \exists h_2 \in \mathcal{V}_G, h_2 \succeq h \succeq h_1\}$$

In [4], an algorithm that computes the version space $\mathcal{V}$ by identifying its upper and lower bounds $\mathcal{V}_S$ and $\mathcal{V}_G$ has been proposed.

The above framework has some limits. First, finding the version space is not sufficient for classifying examples out of the training set. This is due to possible conflicts between hypothesis. Second, it has been shown that the complexity of the algorithm that identifies $\mathcal{V}_S$ and $\mathcal{V}_G$ is very high. In order to palliate that limit, learning algorithms try in general to reach only one hypothesis in the version space by using heuristical exploration of $\mathcal{H}$ (from general to specific exploration, for instance FOIL [8], or from specific to general exploration, for instance PROGOL [6]). That hypothesis is then used for classifying new objects. Moreover, it is obvious that this framework does not support inconsistent set of examples:

PROPERTY 7. *[4] If the set $\mathcal{S}$ is inconsistent, then the version space $\mathcal{V} = \emptyset$.*

A consequence of the above result is that no concept can be learned. This problem may appear in the case of noisy training data set.
Let us now show how the above ASC can retrieve the results of the version space learning, namely the version space and its lower and upper bounds. Before doing that, we start first by introducing some useful notations.
Let Hyp be a function that returns for a given set of arguments, their non empty supports. In other words, this function returns all the hypothesis used to build arguments:

DEFINITION 17. *Let $T \subseteq \mathcal{A}$.*

$$\texttt{Hyp}(T) = \{h \mid \exists \langle h, x, u\rangle \in T \text{ and } h \neq \emptyset\}$$

Now we will show that the argumentation-based model for concept learning computes in an elegant way the version space $\mathcal{V}$.

PROPOSITION 9. *Let $\langle \mathcal{A}, defeat\rangle$ be a ASC. Let $\mathcal{E}$ be its grounded extension, and $\mathcal{E}_1, \ldots, \mathcal{E}_n$ its preferred (stable) extensions. If the set $\mathcal{S}$ is consistent then:*

$$\texttt{Hyp}(\mathcal{E}) = \texttt{Hyp}(\mathcal{E}_1) = \ldots = \texttt{Hyp}(\mathcal{E}_n) = \mathcal{V}$$

where $\mathcal{V}$ is the version space.

PROOF. Let $\mathcal{E}_i$ be an extension under a given semantics.

$\texttt{Hyp}(\mathcal{E}_i) \subseteq \mathcal{V}$: Let $h \in \texttt{Hyp}(\mathcal{E}_i)$, then $\exists \langle h, x, u\rangle \in \mathcal{E}_i$.

Let us assume that $\exists (x_i, u_i) \in \mathcal{S}$ such that $h(x_i) \neq u_i$. This means $\langle \emptyset, x_i, u_i\rangle$ undercuts $\langle h, x, u\rangle$ (according to Definition 9). Consequently, $\langle \emptyset, x_i, u_i\rangle$ defeats $\langle h, x, u\rangle$. However, according to Property 1, $\langle \emptyset, x_i, u_i\rangle \in \mathcal{A}_\mathcal{S}$, thus $\langle \emptyset, x_i, u_i\rangle \in \mathcal{E}_i$. Contradiction because $\mathcal{E}_i$ is an extension, thus by definition it is conflict-free.

$\mathcal{V} \subseteq \texttt{Hyp}(\mathcal{E}_i)$: Let $h \in \mathcal{V}$, and let us assume that $h \notin \texttt{Hyp}(\mathcal{E}_i)$.
Since $h \in \mathcal{V}$, then $\forall (x_i, u_i) \in \mathcal{S}$, $h(x_i) = u_i$ (1)
Let $(x, u) \in \mathcal{S}$, thus $h(x) = u$ and consequently $\langle h, x, u\rangle \in \mathcal{A}$. Moreover, since $h \notin \texttt{Hyp}(\mathcal{E})$, then $\langle h, x, u\rangle \notin E$. Thus, $\exists \langle h', x', u'\rangle$ that defeats $\langle h, x, u\rangle$.

- Case 1: $h' = \emptyset$. This means that $\langle \emptyset, x', u'\rangle$ undercuts $\langle h, x, u\rangle$ and $h(x') \neq u'$ Contradiction with (1).
- Case 2: $h' \neq \emptyset$. This means that $\langle h', x', u'\rangle$ rebuts $\langle h, x, u\rangle$. Consequently, $x \equiv x'$ and $u \neq u'$. However, since $h \in \mathcal{V}$, then h is sound with $\mathcal{S}$. Thus, $\langle \emptyset, x, u\rangle$ defeats $\langle h', x', u'\rangle$, then $\langle \emptyset, x, u\rangle$ defeats $\langle h, x, u\rangle$. Since $\langle \emptyset, x, u\rangle \in \mathcal{S}$, then $\langle h, x, u\rangle \in \mathcal{F}(\mathcal{C})$ and consequently, $\langle h, x, u\rangle \in \mathcal{E}_i$. Contradiction.

□

The above result is of great importance. It shows that to get the version space, one only needs to compute the grounded extension.
We can also show that if a given argument is in an extension $\mathcal{E}_i$, then any argument based on an hypothesis from the version space that supports the same conclusion is in that extension. Formally:

PROPOSITION 10. *Let $\langle \mathcal{A}, defeat\rangle$ be a ASC, and $\mathcal{E}_1, \ldots, \mathcal{E}_n$ its extensions under a given semantics. If $< h, x, u > \in \mathcal{E}_i$, then $\forall h' \in \mathcal{V}$ s.t. $h' \neq h$ if $h'(x) = u$ then $< h', x, u > \in \mathcal{E}_i$.*

Using the grounded extension, one can characterize the upper and the lower bounds of the version space. The upper bound corresponds to the most preferred arguments (w.r.t Pref) of the grounded extension, whereas the lower bound corresponds to the less preferred ones.

PROPOSITION 11. *Let $\langle \mathcal{A}, defeat\rangle$ be a ASC, and $\mathcal{E}$ its grounded extension.*

- $\mathcal{V}_G = \{h \mid \exists <h, x, u> \in \mathcal{E}$ *s.t* $\forall <h', x', u'> \in \mathcal{E}$, *not* $(<h', x', u'>$ *Pref* $<h, x, u>)\}$.
- $\mathcal{V}_S = \{h \mid \exists <h, x, u> \in \mathcal{E}$ *s.t* $\forall <h', x', u'> \in \mathcal{E}$, *not* $(<h, x, u>$ *Pref* $<h', x', u'>)\}$.

PROOF.

$\mathcal{V}_G = \{h \mid \exists <h, x, u> \in \mathcal{E}$ s.t $\forall <h', x', u'> \in \mathcal{E}$, not $(<h', x', u'>$ Pref $<h, x, u>)\}$.

- Let $h \in \mathcal{V}_G$, thus $h \in \mathcal{V}$, and $\forall h' \in \mathcal{V}$, $h \succeq h'$. Since $h \in \mathcal{V}$, thus, $h \in \texttt{Hyp}(\mathcal{E})$, with $\mathcal{E}$ an extension. Then, $\exists \langle h, x, u\rangle \in \mathcal{E}$. Since $h \succeq h'$ for any $h' \in \mathcal{V}$, then $h \succeq h'$ for any $h' \in \texttt{Hyp}(\mathcal{E})$. Thus, $\langle h, x, u\rangle$ Pref $\langle h', x', u'\rangle$, $\forall \langle h', x', u'\rangle \in \mathcal{E}$.
- Let $\langle h, x, u\rangle \in \mathcal{E}$ such that $\forall \langle h', x', u'\rangle \in \mathcal{E}$, and not$(\langle h', x', u'\rangle Pref \langle h, x, u\rangle)$. Thus, $h \in \texttt{Hyp}(\mathcal{E})$, and $\forall h' \in \texttt{Hyp}(\mathcal{E})$, not$(h' \succeq h)$, thus $h \in \mathcal{V}_G$.

$\mathcal{V}_S = \{h \mid \exists <h, x, u> \in \mathcal{E}$ s.t $\forall <h', x', u'> \in \mathcal{E}$, not $(<h, x, u>$ Pref $<h', x', u'>)\}$.

- Let $h \in \mathcal{V}_S$, thus $\nexists h' \in \mathcal{V}$ such that $h \succeq h'$. Since $h \in \mathcal{V}_S$, then $h \in \mathcal{V}$ and consequently, $h \in \texttt{Hyp}(\mathcal{E})$. This means that $\exists \langle h, x, u\rangle \in \mathcal{E}$. Let us assume that $\exists \langle h', x', u'\rangle \in \mathcal{E}$ such that $\langle h, x, u\rangle$ $Pref$ $\langle h', x', u'\rangle$, thus $h \succeq h'$. Contradiction with the fact that $h \in \mathcal{V}_S$.
- Let $\langle h, x, u\rangle \in \mathcal{E}$ such that $\forall \langle h', x', u'\rangle \in \mathcal{E}$, and not$(\langle h, x, u\rangle Pref \langle h', x', u'\rangle)$, thus not$(h \succeq h')$. Since $h \in \mathcal{V}$, and $\forall h' \in \mathcal{V}$, not$(h \succeq h')$, then $h \in \mathcal{V}_S$.

□

6. CONCLUSION

Recently, some researchers have tried to use argumentation techniques in machine learning [10, 5]. The basic idea behind their work is to improve existing algorithms in learning by providing arguments. However, they don't exploit the whole power of argumentation theory. This paper has proposed, to the best of our knowledge, the first framework for classification that is completely argumentation-based, and that uses Dung's semantics.

This framework considers the classification problem as a process that follows four main steps: it first constructs arguments in favor of classifications of examples from a set of training examples, and a set of hypothesis. Conflicts between arguments may appear when two arguments classify the same example in different classes. Once the arguments identified, it is possible to compare them on the basis of their strengths. The idea is that arguments coming from the set of training examples are stronger than arguments built from the set of hypothesis. Similarly, arguments based on most preferred hypothesis are stronger than arguments built from less preferred hypothesis. We have shown that acceptability semantics of the ASC retrieves and even characterizes the version space and its upper and lower bounds. Thus, the argumentation-based approach gives another interpretation of the version space as well as its two bounds in terms of arguments. We have also shown that when the set of training examples is inconsistent, it is still possible to classify examples. Indeed, in this particular case, the version space is empty as it is the case in the version space learning framework. A last and not least feature of our framework consists of defining the class of each example on the basis of all the hypothesis and not only one, and also to suggest four intuitive decision criteria for that purpose.
A first extension of this framework would be to explore the proof theories in argumentation that test directly whether a given argument is in the grounded extension without computing this last. This means that one may know the class of an example without exploring the whole hypothesis space.

7. REFERENCES

[1] L. Amgoud and C. Cayrol. Inferring from inconsistency in preference-based argumentation frameworks. *Int. Journal of Automated Reasoning*, Volume 29 (2):125–169, 2002.

[2] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77:321–357, 1995.

[3] S. A. Gómez and C. I. Chesñevar. Integrating defeasible argumentation with fuzzy art neural networks for pattern classification. In *Proc. ECML'03*, Dubrovnik, September 2003.

[4] T. Mitchell. Generalization as search. *Artificial intelligence*, 18:203–226, 1982.

[5] M. Mozina, J. Zabkar, and I. Bratko. Argument based rule learning. In *Proc. of the In 17th European Conference on Artificial Intelligence, ECAI'06.*

[6] S. Muggleton. Inverse entailment and Progol. *New Generation Computing*, 13:245–286, 1995.

[7] H. Prakken and G. Sartor. Argument-based extended logic programming with defeasible priorities. *Journal of Applied Non-Classical Logics*, 7:25–75, 1997.

[8] J. R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5:239–266, 1990.

[9] G. R. Simari and R. P. Loui. A mathematical treatment of defeasible reasoning and its implementation. *Artificial Intelligence and Law*, 53:125–157, 1992.

[10] J. Zabkar, M. Mozina, J. Videcnik, and I. Bratko. Argument based machine learning in a medical domain. In I. Press, editor, *Proc. of the 1st International Conference on Computational Models of Argument*, pages 59–70, 2006.