A Hybrid Approach to Pattern Classification Using Neural Networks and Defeasible Argumentation

Sergio Alejandro Gómez
Artificial Intelligence Laboratory
Dept. of Computer Science and Eng.
Universidad Nacional del Sur
Alem 1253 – B8000CPB Bahía Blanca, ARGENTINA
Tel:(+54)(291) 459-5135
Email: `sag@cs.uns.edu.ar`

Carlos Iván Chesñevar
Artificial Intelligence Research Group
Dept. of Computer Science
Universitat de Lleida
C/Jaume II, 69 – E-25001 Lleida, SPAIN
Tel:(+34)(973)70-2764
Email: `cic@eup.udl.es`

KEYWORDS: Defeasible Argumentation, Neural networks, Pattern Classification

Abstract

Many classification systems rely on clustering techniques in which a collection of training examples is provided as an input, and a number of clusters $c_1, \ldots c_m$ modeling some concept C results as an output, such that every cluster c_i is labeled as *positive* or *negative*. In such a setting clusters can overlap, and a new unlabeled instance can be assigned to more than one cluster with conflicting labels. In the literature, such a case is usually solved non-deterministically by making a random choice. This paper introduces a novel, hybrid approach to solve the above problem by combining a neural network N along with a background theory T specified in *defeasible logic programming* (DeLP) which models preference criteria for performing clustering.

Introduction

Many classification systems rely on clustering techniques in which a collection of labeled training examples $\{e_1, e_2, \ldots e_n\}$ (each of them labeled as positive or negative) is provided as an input, and a number of clusters $c_1, \ldots c_m$ modeling some concept C results as an output. Every cluster c_i is labeled as *positive* (resp. *negative*) indicating that those examples in the cluster belong (resp. do not belong) to the concept C. Given a new, unlabeled instance e_{new}, the above classification is used to determine to which particular cluster c_i this new instance belongs. Should the cluster c_i be labeled as positive (negative), then the instance e_{new} is regarded as positive (negative). This approach has been exploited in some applications such as the web document filtering agent Querando! (Gómez & Lanzarini 2001) and in the counter-propagation neural network model (Skapura 1996; Rao & Rao 1995). In such a setting clusters can overlap, and a new unlabeled instance can be assigned to *more than one cluster* with conflicting labels (ie., some clusters are positive whereas others are negative). Such a case is usually solved non-deterministically by making a random choice.

This paper introduces a novel, hybrid approach to solve the above problem by combining a background theory T specified in *defeasible logic programming* (DeLP) (García & Simari 2004) and a neural network N based on the Fuzzy Adaptive Resonance Theory model (Carpenter, Grossberg, & Rosen 1991). Given a new, unlabeled instance e_{new} it will be first analyzed and classified using the neural network N. Should e_{new} belong to one or more conflicting clusters, then defeasible argumentation based on the theory T will be used to make a decision based on preference criteria declaratively specified by the user.

Fuzzy ART Neural Networks: Fundamentals

Fuzzy Adaptive Resonance Theory (ART) (Carpenter, Grossberg, & Rosen 1991; Rao & Rao 1995) is a class of neurally inspired models for clustering and classification of sensory data, and associations between such data for representing concepts. Fuzzy ART performs unsupervised learning of categories under continuous presentation of inputs through a process of 'adaptive resonance' in which the learned patterns adapt only to inputs considered to be relevant. Thus the ART models solve the so-called *stability-plasticity* dilemma where new patterns are learned without forgetting those already learned.

The Fuzzy ART neural network model accepts M-dimensional *analog* patterns $a_1, a_2, \ldots a_n$ (with components in the real interval $[0, 1]$) (Lavoie, Crespo, & Savaria 1999) which are clustered into *categories*. Its behavior can be tuned by three parameters: $\alpha > 0$, learning rate $0 \leq \beta \leq 1$, and vigilance $0 \leq \rho \leq 1$. Each category j is represented by a $2M$-dimensional weight vector $w_j = (u, v^c)$. Input vectors I for the network have the form $I = (a, a^c)$. A choice function T_j=$(|I \wedge w_j|)/(\alpha + |w_j|)$ is computed for every category, and the similarity between w_j and I is computed on the basis of ρ using the criterion $|I \wedge w_J|/|I| \leq \rho$. If such test is passed, resonance occurs and learning takes place.For every input pattern, either an existing category can be selected (and possibly expanded) or a new category is created.

The behavior of a Fuzzy ART lends itself well to a geometrical interpretation of category prototypes w_j as *hyper-rectangles* in the input space with corners u and v. Such rectangles are allowed to overlap each other. Given a set $S = \{e_1, e_2, \ldots, e_n\}$ of positive and negative training instances wrt some concept C, the application of the Fuzzy ART neural network will result in a number of labeled clusters $\{c_1, c_2, \ldots, c_m\}$. A cluster labeled as positive (resp. negative) will group instances belonging (resp. not belonging) to the concept C. In the Fuzzy ART setting, conflict appears when a new unlabeled instance is classified as belonging to *more than one cluster with different labels*. In the literature (Lavoie, Crespo, & Savaria 1999), such situation is usually solved nondeterministically by making a random choice.

Modeling Argumentation in DeLP

Defeasible logic programming (DeLP) (García & Simari 2004) is a particular formalization of defeasible argumentation (Chesñevar, Maguitman, & Loui 2000; Prakken & Vreeswijk 2002) based on logic programming. A defeasible logic program (delp) is a set $K = (\Pi, \Delta)$ of Horn-like clauses, where Π and Δ stand for sets of strict and defeasible knowledge, respectively. The set Π of strict knowledge involves *strict rules* of the form $p \;\leftarrow\; q_1, \ldots, q_k$ and *facts* (strict rules with empty body), and it is assumed to be *non-contradictory*. The set Δ of defeasible knowledge involves *defeasible rules* of the form $p \multimap q_1, \ldots, q_k$, which stands for $q_1, \ldots q_k$ *provide a tentative reason to believe* p. The underlying logical language is that of extended logic programming, enriched with a special symbol "$\multimap$" to denote defeasible rules. Both default and classical negation are allowed (denoted not and $\sim$, resp.). Syntactically, the symbol "$\multimap$" is all what distinguishes a *defeasible* rule $p \multimap q_1, \ldots q_k$ from a *strict* (non-defeasible) rule $p \;\leftarrow\; q_1, \ldots, q_k$. DeLP rules are thus Horn-like clauses to be thought of as *inference rules* rather than implications in the object language. Deriving literals in DeLP results in the construction of *arguments*. An argument $\mathcal{A}$ is a (possibly empty) set of ground defeasible rules that together with the set Π provide a logical proof for a given literal h, satisfying the additional requirements of *non-contradiction* and *minimality*.

Definition 1 (Argument) *Given a DeLP program $\mathcal{P}$, an* argument $\mathcal{A}$ *for a query q, denoted $\langle \mathcal{A}, q \rangle$, is a subset of ground instances of defeasible rules in $\mathcal{P}$, such that:*

1. *there exists a* defeasible derivation *for q from $\Pi \cup \mathcal{A}$,*
2. *$\Pi \cup \mathcal{A}$ is non-contradictory (ie, $\Pi \cup \mathcal{A}$ does not entail two complementary literals p and $\sim p$ (or p and* not *p)), and*
3. *$\mathcal{A}$ is minimal with respect to set inclusion.*

An argument $\langle \mathcal{A}_1, Q_1 \rangle$ is a sub-argument *of another argument $\langle \mathcal{A}_2, Q_2 \rangle$ if $\mathcal{A}_1 \subseteq \mathcal{A}_2$. Given a DeLP program $\mathcal{P}$, $Args(\mathcal{P})$ denotes the set of all possible arguments that can be derived from $\mathcal{P}$.*

The notion of defeasible derivation corresponds to the usual query-driven SLD derivation used in logic programming, performed by backward chaining on both strict and defeasible rules; in this context a negated literal $\sim p$ is treated just as a new predicate name no_p. Minimality imposes a kind of 'Occam's razor principle' (Simari & Loui 1992) on argument construction: any superset $\mathcal{A}'$ of $\mathcal{A}$ can be proven to be 'weaker' than $\mathcal{A}$ itself, as the former relies on more defeasible information. The non-contradiction requirement forbids the use of (ground instances of) defeasible rules in an argument $\mathcal{A}$ whenever $\Pi \cup \mathcal{A}$ entails two complementary literals. It should be noted that non-contradiction captures the two usual approaches to negation in logic programming (viz. default negation and classical negation), both of which are present in DeLP and related to the notion of counterargument, as shown next.

Definition 2 (Counterargument. Defeat) *An argument $\langle \mathcal{A}_1, q_1 \rangle$ is a* counterargument *for an argument $\langle \mathcal{A}_2, q_2 \rangle$ iff*

1. *There is an subargument $\langle \mathcal{A}, q \rangle$ of $\langle \mathcal{A}_2, q_2 \rangle$ such that the set $\Pi \cup \{q_1, q\}$ is contradictory.*
2. *A literal* not q_1 *is present in the body of some rule in $\mathcal{A}_1$.*

An argument $\langle \mathcal{A}_1, q_1 \rangle$ is a defeater *for an argument $\langle \mathcal{A}_2, q_2 \rangle$ if $\langle \mathcal{A}_1, q_1 \rangle$ counterargues $\langle \mathcal{A}_2, q_2 \rangle$, and $\langle \mathcal{A}_1, q_1 \rangle$ is preferred over $\langle \mathcal{A}_2, q_2 \rangle$ wrt a* preference criterion $\preceq$ *on conflicting arguments. Such criterion is defined as a partial order $\preceq \subseteq Args(\mathcal{P}) \times Args(\mathcal{P})$. For cases (1) and (2) above, we distinguish between* proper *and* blocking defeaters *as follows:*

- *In case 1, the argument $\langle \mathcal{A}_1, q_1 \rangle$ will be called a* proper defeater *for $\langle \mathcal{A}_2, q_2 \rangle$ iff $\langle \mathcal{A}_1, q_1 \rangle$ is strictly preferred over $\langle \mathcal{A}, q \rangle$ wrt $\preceq$.*
- *In case 1, if $\langle \mathcal{A}_1, q_1 \rangle$ and $\langle \mathcal{A}, q \rangle$ are unrelated to each other, or in case 2, $\langle \mathcal{A}_1, q_1 \rangle$ will be called a* blocking defeater *for $\langle \mathcal{A}_2, q_2 \rangle$.*

Specificity (Simari & Loui 1992) is typically used as a syntax-based criterion among conflicting arguments, preferring those arguments which are *more informed* or *more direct* (Simari & Loui 1992; Stolzenburg *et al.* 2003). However, other alternative partial orders could also be valid.

Computing Warrant Through Dialectical Analysis

An *argumentation line* starting in an argument $\langle \mathcal{A}_0, Q_0 \rangle$ (denoted $\lambda^{\langle \mathcal{A}_0, q_0 \rangle}$) is a sequence $[\langle \mathcal{A}_0, Q_0 \rangle, \langle \mathcal{A}_1, Q_1 \rangle, \langle \mathcal{A}_2, Q_2 \rangle, \ldots, \langle \mathcal{A}_n, Q_n \rangle \ \ldots]$ that can be thought of as an exchange of arguments between two parties, a *proponent* (evenly-indexed arguments) and an *opponent* (oddly-indexed arguments). Each $\langle \mathcal{A}_i, Q_i \rangle$ is a defeater for the previous argument $\langle \mathcal{A}_{i-1}, Q_{i-1} \rangle$ in the sequence, $i > 0$. In order to avoid *fallacious* reasoning, dialectics imposes additional constraints on such an argument exchange to be considered rationally acceptable:

- **Non-contradiction** Given an argumentation line λ, the set of arguments of the proponent (resp. opponent) should be *non-contradictory* wrt $\mathcal{P}$. Non-contradiction for a set of arguments is defined as follows: a set $S = \bigcup_{i=1}^{n} \{\langle \mathcal{A}_i, Q_i \rangle\}$ is *contradictory* wrt a DeLP program $\mathcal{P}$ iff $\Pi \cup \bigcup_{i=1}^{n} \mathcal{A}_i$ is contradictory.
- **No circular argumentation** No argument $\langle \mathcal{A}_j, Q_j \rangle$ in λ is a sub-argument of an argument $\langle \mathcal{A}_i, Q_i \rangle$ in λ, $i < j$.
- **Progressive argumentation** Every blocking defeater $\langle \mathcal{A}_i, Q_i \rangle$ in λ is defeated by a proper defeater $\langle \mathcal{A}_{i+1}, Q_{i+1} \rangle$ in λ.

The first condition disallows the use of contradictory information on either side (proponent or opponent). The second condition eliminates the *"circulus in demonstrando"* fallacy (circular reasoning). Finally, the last condition enforces the use of a stronger argument to defeat an argument which acts as a blocking defeater. An argumentation line satisfying the above restrictions is called *acceptable*, and can be proven to be finite (García & Simari 2004).

Given a DeLP program $\mathcal{P}$ and an initial argument $\langle \mathcal{A}_0, Q_0 \rangle$, the set of all acceptable argumentation lines starting in $\langle \mathcal{A}_0, Q_0 \rangle$ accounts for a whole dialectical analysis for $\langle \mathcal{A}_0, Q_0 \rangle$ (ie., all possible dialogues about $\langle \mathcal{A}_0, Q_0 \rangle$ between proponent and opponent), formalized as a *dialectical tree*.

Definition 3 (Dialectical Tree) *Let $\mathcal{P}$ be a DeLP program, and let $\mathcal{A}_0$ be an argument for Q_0 in $\mathcal{P}$. A* dialectical tree *for $\langle \mathcal{A}_0, Q_0 \rangle$, denoted $\mathcal{T}_{\langle \mathcal{A}_0, Q_0 \rangle}$, is a tree structure defined as follows:*

1. *The root node of $\mathcal{T}_{\langle \mathcal{A}_0, Q_0 \rangle}$ is $\langle \mathcal{A}_0, Q_0 \rangle$.*
2. *$\langle \mathcal{B}', H' \rangle$ is an immediate child of $\langle \mathcal{B}, H \rangle$ iff there exists an acceptable argumentation line $\lambda^{\langle \mathcal{A}_0, Q_0 \rangle}$ = [$\langle \mathcal{A}_0, Q_0 \rangle$, $\langle \mathcal{A}_1, Q_1 \rangle$, ..., $\langle \mathcal{A}_n, Q_n \rangle$] with two elements $\langle \mathcal{A}_{i+1}, Q_{i+1} \rangle$ = $\langle \mathcal{B}', H' \rangle$ and $\langle \mathcal{A}_i, Q_i \rangle$ = $\langle \mathcal{B}, H \rangle$, for some $i = 0 \ldots n-1$.*

Nodes in a dialectical tree $\mathcal{T}_{\langle \mathcal{A}_0, Q_0 \rangle}$ can be marked as *undefeated* and *defeated* nodes (U-nodes and D-nodes, resp.). A dialectical tree will be marked as an AND-OR tree: all leaves in $\mathcal{T}_{\langle \mathcal{A}_0, Q_0 \rangle}$ will be marked U-nodes (as they have no defeaters), and every inner node is to be marked as *D-node* iff it has at least one U-node as a child, and as *U-node* otherwise. An argument $\langle \mathcal{A}_0, Q_0 \rangle$ is ultimately accepted as valid (or *warranted*) wrt a DeLP program $\mathcal{P}$ iff the root of its associated dialectical tree $\mathcal{T}_{\langle \mathcal{A}_0, Q_0 \rangle}$ is labeled as *U-node*.

Given a DeLP program $\mathcal{P}$, solving a query q wrt $\mathcal{P}$ accounts for determining whether q is supported by a warranted argument. Different doxastic attitudes are distinguished when answering that query q according to the associated status of warrant, in particular:

1. Believe q (resp. $\sim q$) when there is a warranted argument for q (resp. $\sim q$) that follows from $\mathcal{P}$.
2. Believe q is *undecided* whenever neither q nor $\sim q$ are supported by warranted arguments in $\mathcal{P}$.

A Hybrid Approach Combining Fuzzy ART Networks and DeLP

As discussed in the introduction, conflict appears in the Fuzzy ART setting when a new unlabeled instance is classified as belonging to *two or more clusters with different labels*. The proposed hybrid approach involves combining a traditional Fuzzy ART network N with a background theory formalized as a DeLP program $\mathcal{P}$. As the neural network N is fed with a set of training examples, new facts encoding knowledge about such examples as well as the resulting cluster structure are added as part of a DeLP program $\mathcal{P}$. The program $\mathcal{P}$ also models the user's preference criteria to classify new, unlabeled instances belonging to conflicting clusters. This can be encoded by providing appropriate strict and defeasible rules as part of the program $\mathcal{P}$. Several preference criteria among competing clusters are possible, such as:

```
ALGORITHM ClassifyNewInstance
INPUT: Net $N$, DeLP program $\mathcal{P}$, new instance $E$
OUTPUT: $pos$, $neg$, undecided {Classification of $E$}
BEGIN
 Propagate unlabeled instance $E$ through Net $N$
 $CL$ := SetOfClustersContainingNewInstance($E$, $F$ )
 IF every $c_i \in CL$ is $pos$ OR every $c_i \in CL$ is $neg$
   THEN RETURN Label = label of any $c_i \in CL$
   ELSE
     Solve query $is(P, pos)$ using DeLP program $\mathcal{P}$
     IF $is(P, pos)$ is warranted
      THEN RETURN Label=$pos$
      ELSE
        Solve query $is(P, neg)$ using DeLP program $\mathcal{P}$
        IF $is(P, neg)$ is warranted
          THEN RETURN Label=$neg$
          ELSE RETURN Label=undecided
END
```

Figure 1: High-level algorithm for integrating DeLP and the Fuzzy ART model

- The cluster with newer information is preferred over other ones
- The cluster that subsumes more examples is preferred.
- The smallest cluster containing the new instance is preferred.

It must be noted that the above criteria may be also in conflict, making necessary to analyze which one prevails over the other ones. This ultimate decision will be made on the basis of a dialectical analysis performed by the DeLP inference engine.

Figure 1 shows a sketch of an algorithm that combines the use of DeLP and the Fuzzy ART for determining the classification of a new unlabeled instance e_{new} after training the Fuzzy ART network N. The algorithm takes as input a Fuzzy ART neural network, a DeLP program $\mathcal{P}$ (characterizing a set of examples and preference criteria), and the data corresponding to a new unlabeled instance e_{new}. Such an instance e_{new} is first classified using the Fuzzy ART neural network (modifying the cluster structure accordingly if needed). In case that such a classification cannot be solved successfully by the network N, then the program $\mathcal{P}$ is used to perform a dialectical analysis to decide how to label the new instance E. To do so, a distinguished predicate is($<NewInstance>$,$<Label>$) will be considered. The classification will be (1) positive (pos) if the literal $is(E, pos)$ is warranted from $\mathcal{P}$; (2) negative (neg) if the literal $is(E, neg)$ is warranted from $\mathcal{P}$; (3) undecided if neither (1) nor (2) hold. It must be noted that there is a theorem (García & Simari 2004) ensuring that if some argument $\langle \mathcal{A}, h \rangle$ is warranted, then there does not exist a warranted argument for the opposite conclusion, i.e, $\langle \mathcal{B}, \sim h \rangle$. As a consequence, when analyzing the labeling associated with a new instance E, it cannot be the case that both $is(E, pos)$ and $is(E, neg)$ hold, provided that pos and neg are defined as opposite concepts.

A Worked Example

In this section we will discuss an example of how the proposed approach works. First we will describe how the training of the neural network results in new facts added to a DeLP program $\mathcal{P}$. Then we will show how to specify preference criteria in $\mathcal{P}$. Finally we show how to apply the algorithm shown in Fig. 1 for solving a conflicting situation wrt a new unlabeled instance e_{new} and a particular program $\mathcal{P}$.

Encoding Training Information

Suppose that a set $S = \{p_1, p_2, \ldots, p_k\}$ of training instances in a 2-dimensional space are obtained from a particular experiment, each of them having an associated timestamp. Such set S is provided as a training set for a Fuzzy ART neural network N, resulting in three clusters c_1^+, c_2^- and c_3^- being learnt (see Fig. 2). As the network N is trained, new facts corresponding to a DeLP program $\mathcal{P}$ will be generated to encode some of the above information, as shown below:

$$\begin{array}{ll}
point(p_1, neg, 5, coor(x_1, y_1)). & trigger(p_3, c_2).\\
point(p_2, neg, 7, coor(x_2, y_2)). & trigger(p_5, c_1).\\
point(p_3, neg, 9.9, coor(x_3, y_3)). & trigger(p_2, c_3).\\
point(p_4, pos, 10.7, coor(x_4, y_4)). & cluster(c_1, pos).\\
point(p_5, pos, 12.5, coor(x_5, y_5)). & cluster(c_2, neg).\\
\ldots & cluster(c_3, neg).
\end{array}$$

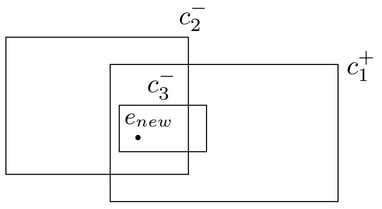

Figure 2: Unlabeled instance e_{new} belonging to conflicting clusters c_1, c_2, and c_3

Note that every new training instance corresponding to a point p labeled as s at time t with coordinates (x, y) results in a fact $point(p, s, t, coor(x, y))$ added to the DeLP program $\mathcal{P}$. When the dynamics of the neural network determines that a new cluster is to be created by occurrence of a point p, a new fact $trigger(p, c)$ is added to $\mathcal{P}$. Analogously, when the network N determines that a cluster c is labeled as positive (resp. negative), a new fact $cluster(c, pos)$ (resp. $cluster(c, neg)$) is also added to $\mathcal{P}$.

Providing Preference Criteria

Fig. 3 presents strict and defeasible rules that characterize possible preference criteria among clusters. Predicate opp indicates that pos and neg are opposite concepts. Predicate $newer(C_1, C_2)$ holds whenever cluster C_1 is newer than C_2. We adopt here one possible criterion, using the timestamp associated with the trigger point for comparing clusters. Predicate $subset(C_1, C_2)$ holds whenever cluster C_1 is subsumed by cluster C_2. This is assumed to be computed elsewhere, based on the data structures of the neural

$$\begin{array}{rcl}
opp(pos, neg). & & \\
opp(neg, pos). & & \\
newer(C_1, C_2) & \leftarrow & trigger(P_1, C_1), point(P_1, \_, T_1, \_),\\
 & & trigger(P_2, C_2), point(P_2, \_, T_2, \_),\\
 & & T_1 > T_2\\
subset(C_1, C_2) & \leftarrow & \text{[computed elsewhere]}\\
activates(P, C) & \leftarrow & \text{[computed elsewhere]}\\
\sim is(P, L_1) & \leftarrow & is(P, L_2), opp(L_1, L_2).\\
is(P, L) & \prec & assume(P, L).\\
assume(P, L) & \prec & belongs(P, C), cluster(C, L).\\
assume(P, L_2) & \prec & newer(C_2, C_1), cluster(C_1, L_1),\\
 & & cluster(C_2, L_2),\\
 & & belongs(P, C_2), belongs(P, C_1).\\
belongs(P, C) & \prec & activates(P, C).\\
\sim belongs(P, C_1) & \prec & subset(C_2, C_1), cluster(C_1, L_1),\\
 & & cluster(C_2, L_2), opp(L_1, L_2),\\
 & & activates(P, C_2).
\end{array}$$

Figure 3: Modeling preference among clusters in DeLP

network N where cluster information is stored. The same applies to predicate $activates(P, C)$, which holds whenever a point P falls within cluster C. The definition of predicate is involves two parts: one the one hand, we specify that if a cluster C is labeled as positive (resp. negative), then it *is not* negative (resp. positive); on the other hand, we also have a defeasible rule indicating that a cluster C gets a label L if we have tentative reasons to assume this to be so. The predicate $assume(P, L)$ defeasibly holds whenever we can assume that a point P gets a label L. First, belonging to a cluster C with label L is a tentative reason to assume that point P gets that label L. If point P belongs to two clusters C_1 and C_2, and C_2 is newer than C_1, this provides a tentative reason to assume that P should be labeled as the newer cluster C_2. If P is found within cluster C (ie. P activates C), then usually P belongs to cluster C. If P belongs to a cluster C_2 which is a subset of another cluster C_1 with a conflicting label, then this is a tentative reason to believe that P does not belong to C_1 (the smaller cluster is preferred over the bigger one).

Performing Dialectical Analysis

Consider a new unlabeled instance e_{new}, as shown in Fig. 2. As discussed before, in the traditional Fuzzy ART setting, such instance would be classified non-deterministically. A DeLP program $\mathcal{P}$ as the one presented before can provide additional, *qualitative* information for making such a decision. As e_{new} belongs to the intersection of clusters c_1, c_2 and c_3, and not all of them have the same label, the algorithm shown in Fig. 1 will start searching for a warranted argument for $is(e_{new}, pos)$, which involves solving the query $is(e_{new}, pos)$ wrt $\mathcal{P}$. The DeLP inference engine will find an argument $\langle \mathcal{A}_1, is(e_{new}, pos)\rangle$, with
$\mathcal{A}_1$={ $(is(e_{new}, pos) \prec assume(e_{new}, pos))$,
$(assume(e_{new}, pos) \prec belongs(e_{new}, c_1), cluster(c_1, pos))$,
$(belongs(e_{new}, c_1) \prec activates(e_{new}, c_1))$}
supporting the fact that e_{new} should be labeled as positive, as it belongs to positive cluster c_1. The DeLP inference engine will search (in a depth-first fashion) for defeaters for $\langle \mathcal{A}_1, is(e_{new}, pos)\rangle$. A blocking defeater $\langle \mathcal{A}_2, is(e_{new}, neg)\rangle$, will be found, stating that e_{new} should

be labeled as negative as it belongs to negative cluster c_2. Here we have

$\mathcal{A}_2$={$(is(e_{new}, neg) \mathrel{-\!\!\prec} assume(e_{new}, neg))$,
$(assume(e_{new}, neg) \mathrel{-\!\!\prec} belongs(e_{new}, c_2)$,$cluster(c_2, neg))$,
$(belongs(e_{new}, c_2) \mathrel{-\!\!\prec} activates(e_{new}, c_2))$}

Note in this case that $\Pi \cup \mathcal{A}_2$ derives the complement of $\mathcal{A}_1$ (i.e. $\sim is(e_{new}, pos)$) via the strict rule $\sim is(P, L_1) \leftarrow is(P, L_2),\ opp(L_1, L_2)$ (see Fig. 3). This second argument is in turn defeated by a more informed argument $\langle \mathcal{A}_3, is(e_{new}, pos)\rangle$: the new instance e_{new} should be labeled as positive as it belongs to clusters c_1 and c_2, but positive cluster c_1 is newer than negative cluster c_2. Here we have:

$\mathcal{A}_3$={$(is(e_{new}, pos) \mathrel{-\!\!\prec} assume(e_{new}, pos))$,
$(assume(e_{new}, pos) \mathrel{-\!\!\prec} newer(c_1, c_2),\ cluster(c_1, pos)$,
$cluster(c_2, neg),\ belongs(e_{new}, c_2)$,$belongs(e_{new}, c_1))$,
$(belongs(e_{new}, c_1) \mathrel{-\!\!\prec} activates(e_{new}, c_1))$
$(belongs(e_{new}, c_2) \mathrel{-\!\!\prec} activates(e_{new}, c_2))$

Note that $\langle \mathcal{A}_1, is(e_{new}, pos)\rangle$ could not be used once again to defeat $\langle \mathcal{A}_2, is(e_{new}, neg)\rangle$, as it would be a fallacious, circular reasoning, which is disallowed in acceptable argumentation lines. However there is a fourth argument $\langle \mathcal{A}_4, \sim belongs(e_{new}, c_1)\rangle$ that can be derived from $\mathcal{P}$ which defeats $\langle \mathcal{A}_3, is(e_{new}, pos)\rangle$, providing a more informed argument about the notion of membership for an instance: e_{new} does not belong to cluster c_1 because that cluster subsumes c_3, and e_{new} belongs to c_3. Here we have:
$\mathcal{A}_4$={$\sim belongs(e_{new}, c_1) \mathrel{-\!\!\prec} subset(c_3, c_1),\ cluster(c_1, pos)$,
$cluster(c_3, neg),\ opp(pos, neg),\ activates(e_{new}, c_3)$ }

Note that the argument $\langle \mathcal{A}_4, \sim belongs(e_{new}, c_1)\rangle$ is also a defeater for the first argument $\langle \mathcal{A}_1, is(e_{new}, pos)\rangle$. This completes the computation of the dialectical tree rooted in $\langle \mathcal{A}_1, is(e_{new}, pos)\rangle$, as there are no more arguments to consider as acceptable defeaters. The dialectical tree can be marked as discussed before: leaves will be marked as undefeated nodes (U-nodes), as they have no defeaters. Every inner node will be marked as a defeated node (D-node) if it has at least one U-node as a child, and as a U-node otherwise. The original argument (the root node) will be a warranted argument iff it is marked as U-node. In the preceding analysis, the resulting marked dialectical tree is shown in Fig. 4(a): nodes are arguments, and branches stand for acceptable argumentation lines. As the root of the tree is marked as D, the original argument $\langle \mathcal{A}_1, is(e_{new}, pos)\rangle$ is not warranted. The DeLP inference engine will start searching automatically for other warranted arguments for $is(e_{new}, pos)$. Fig. 4(b) shows the dialectical tree for $\langle \mathcal{A}_3, is(e_{new}, pos)\rangle$, in which $\langle \mathcal{A}_3, is(e_{new}, pos)\rangle$ is not a warranted argument. There are no other arguments for $is(e_{new}, pos)$ to consider. Following the algorithm shown in Fig. 1, the DeLP inference engine will now start searching for warranted arguments for $is(e_{new}, neg)$. A warranted argument will be found, namely $\langle \mathcal{A}_2, is(e_{new}, neg)\rangle$, whose dialectical tree is shown in Fig. 4(c). Therefore, program $\mathcal{P}$ allows us finally to conclude that the given unlabeled instance e_{new} should be labeled as negative.

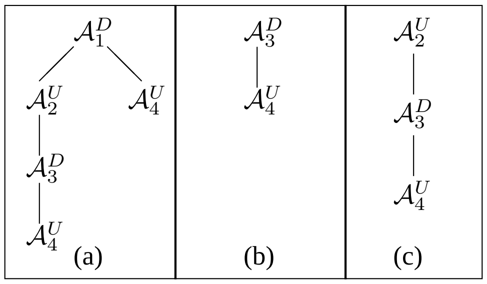

Figure 4: Dialectical analysis for arguments $\langle \mathcal{A}_1, is(e_{new}, pos)\rangle$, $\langle \mathcal{A}_3, is(e_{new}, pos)\rangle$ and $\langle \mathcal{A}_2, is(e_{new}, neg)\rangle$

DeLP: Implementation Issues

Performing defeasible argumentation is a computationally complex task. An abstract machine for an efficient implementation of DeLP has been developed, based on an extension of the WAM (Warren's Abstract Machine) for Prolog. Several features leading to efficient implementations of DeLP have been also recently studied, particularly those related to comparing conflicting arguments by specificity (Stolzenburg *et al.* 2003) and pruning the search space (Chesñevar, Simari, & García 2000). In particular, the search space associated with dialectical trees is reduced by applying $\alpha - \beta$ pruning. Thus, in Fig. 4(a), the right branch of the tree is not even computed, as the root node can be already deemed as ultimately defeated after computing the left branch.

Related Work

The area of clustering algorithms has a wide range of applications which include image processing, information retrieval (Rasmussen 1992), text filtering (Honkela 1997; Gómez & Lanzarini 2001), among others. To the best of our knowledge, in none of these areas argumentation has been used for clustering as described in this paper. In particular, the pitfalls of Fuzzy ART are exploited as an advantage for doing multiple categorization in (Lavoie, Crespo, & Savaria 1999), proposing a variation on the Fuzzy ART model. In early work for combining neural networks and rule sets (Shavlik & Towell 1989), rules are used to initialize the neural network weights, whereas we use defeasible rules for revising a neural network classification *a posteriori*. Other approaches (Johnston & Governatori 2003) involve algorithms for inducing a defeasible theory from a set of training examples. In our case, the defeasible logic theory is assumed to be given. In (Inoue & Kudoh 1997), a method to generate non-monotonic rules with exceptions from positive/negative examples and background knowledge is developed. Such a method induces a defeasible theory from examples; in contrast, the proposed approach uses a defeasible theory for improving an incremental categorization. Another hybrid approach includes an agent collaboration protocol for database initialization of a memory-based reasoning algorithm (Lashkari, Metral, & Maes 1997), using rules for improving learning speed. In contrast, the proposal presented in this paper is aimed to improve learning precision.

Conclusions and Future Work

The growing success of argumentation-based approaches has caused a rich cross-breeding with interesting results in several disciplines, such as legal reasoning (Prakken & Sartor 2002), text classification (Hunter 2001) and decision support systems (Carbogim, Robertson, & Lee 2000). As we have shown in this paper, frameworks for defeasible argumentation can be also integrated with clustering techniques, making them more attractive and suitable for solving real-world applications. Argumentation provides a sound *qualitative* setting for commonsense reasoning, complementing thus the pattern classification process, which relies on *quantitative* aspects of the data involved (such as numeric attributes or probabilities).

Recent research in information technology is focused on developing *argument assistance systems* (Verheij 2004), i.e. systems that can assist users along the argumentation process. We think that such assistance systems could be integrated with the approach outlined in this paper, complementing existing visual tools for clustering and pattern classification (Davidson 2002).

The algorithm presented in this paper has been implemented and tested successfully on several representative problems with different competing criteria for clustering. Part of our current research involves to test it with respect to some benchmark standard collections.[1]

Acknowledgments

This research was partially supported by Project CICYT TIC2001-1577-C03-03 and Ramón y Cajal Program funded by the Ministerio de Ciencia y Tecnología (Spain). The authors would like to thank anonymous reviewers for helpful comments to improve the final version of this paper.

References

Carbogim, D.; Robertson, D.; and Lee, J. 2000. Argument-based applications to knowledge engineering. *The Knowledge Engineering Review* 15(2):119–149.

Carpenter, G.; Grossberg, S.; and Rosen, D. 1991. Fuzzy art: Fast stable learning and categorization of analog patterns by an adaptive resonance system. *Neural Networks* 4:759–771.

Chesñevar, C. I.; Maguitman, A.; and Loui, R. 2000. Logical Models of Argument. *ACM Computing Surveys* 32(4):337–383.

Chesñevar, C. I.; Simari, G. R.; and García, A. 2000. Pruning Search Space in Defeasible Argumentation. In *Proc. of the Workshop on Advances and Trends in AI*, 46–55. XX Intl. Conf. of the SCCC, Santiago, Chile.

Davidson, I. 2002. Visualizing Clustering Results. In *Proc. of 2nd SIAM International Conference on Data Mining, Arlington VA, USA.* SIAM.

García, A. J., and Simari, G. R. 2004. Defeasible Logic Programming an Argumentative Approach. *Theory and Practice of Logic Programming* 4(1):95–138.

Gómez, S. A., and Lanzarini, L. 2001. Querando!: Un agente de filtrado de documentos web. *Procs. of the VII Argentinean Conf. in Computer Science (CACIC)* 1205–1217.

Honkela, T. 1997. *Self-organizing maps in Natural Language Processing.* Ph.D. Dissertation, Helsinky University.

Hunter, A. 2001. Hybrid argumentation systems for structured news reports. *Knowledge Engineering Review* (16):295–329.

Inoue, K., and Kudoh, Y. 1997. Learning Extended Logic Programs. In *Proc. of the 15th IJCAI (vol.1)*, 176–181. Morgan Kaufmann.

Johnston, B., and Governatori, G. 2003. An algorithm for the induction of defeasible logic theories from databases. In *Proc. of the 14th Australasian Database Conference (ADC2003)*, 75–83.

Lashkari, Y.; Metral, M.; and Maes, P. 1997. Collaborative interface agents. In *Readings in Agents*. Morgan Kaufmann. 111–116.

Lavoie, P.; Crespo, J.; and Savaria, Y. 1999. Generalization, discrimination, and multiple categorization using adaptive resonance theory. *IEEE Transactions on Neural Networks* 10(4):757–767.

Prakken, H., and Sartor, G. 2002. The role of logic in computational models of legal argument - a critical survey. In Kakas, A., and Sadri, F., eds., *Computational Logic: Logic Programming and Beyond*. Springer. 342–380.

Prakken, H., and Vreeswijk, G. 2002. Logical Systems for Defeasible Argumentation. In Gabbay, D., and F.Guenther., eds., *Handbook of Philosophical Logic*. Kluwer Academic Publishers. 219–318.

Rao, V., and Rao, H. 1995. *C++ Neural Networks and Fuzzy Logic, Second Edition.* MIS Press.

Rasmussen, E. 1992. Clustering algorithms. In Frakes, W., and Baeza-Yates, R., eds., *Information Retrieval*. Prentice Hall. 419–442.

Shavlik, J., and Towell, G. 1989. An approach to combining explanation-based and neural learning algorithms. *Connection Science* 1(3):233–255.

Simari, G. R., and Loui, R. P. 1992. A Mathematical Treatment of Defeasible Reasoning and its Implementation. *Artificial Intelligence* 53:125–157.

Skapura, D. 1996. *Building Neural Networks.* ACM Press, Addison-Wesley.

Stolzenburg, F.; García, A.; Chesñevar, C. I.; and Simari, G. R. 2003. Computing Generalized Specificity. *Journal of Non-Classical Logics* 13(1):87–113.

Verheij, B. 2004. Artificial argument assistants for defeasible argumentation. *Artificial Intelligence Journal (to appear).*

[1]E.g. http://www.ics.uci.edu/~mlearn/MLRepository.html