FUZZY METHODS FOR SIMPLIFYING A BOOLEAN FORMULA INFERRED FROM EXAMPLES

B. Apolloni, D. Malchiodi, C. Orovas, A. M. Zanaboni

Dipartimento di Scienze dell'Informazione – Università degli Studi di Milano-ITALY

ABSTRACT

In order to state a symbolic description of observed data we learn a *minimal* monotone DNF (Disjunctive Normal Form) formula consistent with them. Then, in the idea that a short formula – i.e. made up of few monomials, each represented by the product of few literals – is better understandable by the user than a longer one, we propose here an algorithm to simplify in this direction the formula learnt from examples. We obtain concise formulas by violating their consistency on a part of the observed examples in some regions of the sample space that we consider as fuzzy borders of the formulas. Special membership functions to these regions allow us to manage the balance between conciseness and description power of the final formula as an optimisation problem that is solved via a simulated annealing procedure on a set of artificially constructed benchmarks.

1. INTRODUCTION

The jump from examples to concepts, for instance from fire observations to the settlement of the thermodynamics laws, might be very long, taking a complex sequence of hierarchical steps, like human kind did over the course of the centuries. In view of capturing some features of this abstraction process, we raised up an agnostic learning procedure [1] that generates a *minimal* function consistent with the examples, in the sense that no formula whose support is included in this function can maintain the consistency. In turn, consistency means that, once we have a set of examples labeled by a flag telling us whether the example must belong to the final formula (positive example) or not (negative one), the formula we find must respect the flags. This kind of minimality however does not guarantee an analogous minimality on the formula description length [2]. Thus, we propose here an algorithm to reduce the number of literals in the formula still preserving its descriptional power.

A shortening may happen for free, when for instance we discover a new monomial still consistent with the examples and including two smaller ones belonging to the formula. However, we accept to sacrifice some accuracy of the formula to get it shorter. It is the typical attitude with which we interpolate a set of experimental points with a line: we don't assume that a true linear relation exists between the point coordinates, but such relation helps us to understand the bulk of the phenomena underlying the points. In our case we may see a consistency violation of the concise formula either as a consequence of a fuzzy definition [3] of its support's contour or as denoting a non clear classification of the missed point flags. Here we prefer the first perspective and use a special membership function to the fuzzy support of the formula to characterize a thickness of the fuzzy border. This allows us to translate the above problem of balancing complexity versus approximation of the final formula into the analogous balancing of its descriptional length and the mean thickness of the borders of the constituting monomials.

We tested numerically the suitability of this perspective on a set of artificially constructed benchmarks, where we drew labeled examples from the support of an original DNF and its complement to the assignment space. The value of the DNF we learn is appreciated through the exact measure of the symmetric difference between its support and the original formula's one.

2. INFERRING A DNF

Given a set $L = \{x_1, \ldots, x_n\}$ of *literals*, a literal being an affirmed or a negated propositional variable of a Boolean formula, we denote by $\mathbf{X}_n=\{0,1\}^n$ the space of the Boolean vectors $\mathbf{x}$ of size n, that can be assigned to the literals of L.

We denote by C the class of such formulas, called *concepts* elsewhere, and, for $g \in C$, by set(g) the set of all the literals occurring in the formula g. A formula is *monotone* if only affirmed variables appear in it. Given a set $M = \{\mathsf{m}_1, \ldots, \mathsf{m}_m\}$ of m monomials, built as a conjunction of literals in L, a formula $g=\bigvee_i \mathsf{m}_i$ is a *DNF*.

By abuse, we will use the same notation for g and support(g), i.e. the set of points $\mathbf{x} \in \mathbf{X}_n$ such that $g(\mathbf{x})=1$. For a given concept g, $E^+ \subseteq \mathbf{X}_n$ and $E^- \subseteq \mathbf{X}_n$ are the sets of positive and negative examples respectively, constituted by assignments belonging the former and not belonging the latter to g.

Definition 2.1: Let us represent a monomial μ by the vector $\mu = (b_1, \ldots, b_n)$ where $b_i = 1$ if $x_i \in \mu$ and $b_i = 0$ otherwise; each binary value b_i is the *crisp membership* value of x_i to μ. With a similar notation, we represent a k-term_DNF $g = \mu_1 \cup \ldots \cup \mu_k$ by a vector B of length nk obtained by concatenating the representations of the joined monomials.

We say that $g = \bigcup_i \mu_i$ is a *DNF crisp description* of $E = E^+ \cup E^-$ if g is consistent with all the positive and negative points of E.

Given a set of examples E, we simply compute a minimal DNF crisp monotone description through the algorithm of Table 1.

*Table 1.*Construction of aDNF from examples.

Construction (E^+, E^-)
BEGIN
  DNF $= \varnothing$
  FOR_ALL $\boldsymbol{u} \in E^+$
    $\mu = \bigcap_{i\,|\,u_i = 1} x_i$
    DNF = **DNF** $\cup \{\mu\}$
  Return **DNF**
END

It is easy to check that μ is the smallest monotone monomial containing the positive sampled point on whose basis it has been created.

3. SYMPLIFYING AN INFERRED DNF

In order to reduce the length of the generated DNF formula g, we work on the single monomials according to a set of simplifying actions, with a consequent broadening of their supports. We want to balance a desirable shortening of the formula with the undesirable loss of description power (in terms of negative points included and positive ones excluded) [4]. We account for this trade-off through a cost function that linearly adds i) the contributions of the single monomials and ii) a cost of a dummy monomial that softly accounts for the combined effect of the single monomials not taken into account in i). In particular, the individual costs are based on a local membership function to a fuzzy relaxation of the original monomials.
The optimization of the single contribution is pursued in conjunction with the minimization of the dummy monomial cost through a simulated annealing algorithm [5].

A first component of the cost of any monomial μ is its length, whose shortening is trivially obtainable through two actions:

- removal of a literal from set(μ)
- removal of the entire monomial

Both actions may corrupt the accuracy of the formula, since the former annexes new points to the support of the relaxed monomial (let us call it μ'), while the second may leave some points in support(μ) missing a description in the DNF.

We manage the first draw back via the introduction of a fuzzy border for the monomial to which annexed points belong according to a fuzzy membership function based on the frequency with which sample points appear therein. On the other hand, points left without a description are attributed to a dummy monomial with an analogous cost.

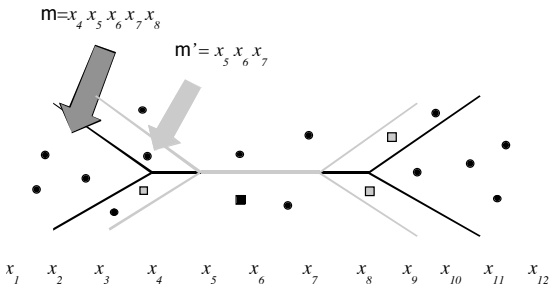


Figure 1. The geometry of simplification of a monomial μ into a monomial μ' on the literals x_i, i = 1, ..., 12. The binary tree representing the assignments is squeezed in correspondence of the literals in μ. Circles and squares denote respectively positive and negative points. Gray squares denote false positive points included in μ' after the simplification.

3.1 A fuzzy border for a monomial.

Starting from a crisp monomial μ, we fuzzy enlarge its contour by annexing boundary regions of the formula. Namely, the removal of a literal from set(μ) gives rise to a new monomial μ' including μ (see Figure 1) [6]. We can start by removing any literal from set(μ), thus obtaining different enlargements.

Whatever the starting point, any subsequent removals of literals give rise to a monotone enlargement of the monomial support. We consider this enlargement as a thick fuzzy border of the monomial that we describe through a fuzzy membership function that decreases from 1 in the crisp monomial to 0 outside its enlargement (see Figure 2). All the points belonging to the same enlargement slice have the same membership value to the fuzzy border. Therefore we associate this value directly to the literal whose removal generated that enlargement slice. In a similar way, by abuse we identify the fuzzy border with the ordered sequence of literals that have been removed from set(μ) to obtain it.

Our construction of the membership function is based on the estimation of the probability that a point falls in a

considered enlargement. Since positive and negative points play different roles in the identification of a concept, in the construction of the membership function we will distinguish between E^+ and E^-. For the same reason, we want to keep the fuzzy border thikness as small as possible. The membership function to an enlarged monomial is given as follows.

Definition 3.1: Given a monomial $\mathbf{m}$, for an ordered sequence $\mathbf{d}=(d_1,\ldots,d_s)$ of length s of literals from set($\mathbf{m}$), let us denote by $\mathbf{d}^k$ its prefix of length k. Let $\mathbf{m}_{\mathbf{d}^0}=\mathbf{m}$, and $\mathbf{m}_{\mathbf{d}^k}$ denote the monomial obtained by flipping from 1 to 0 the crisp membership value of literal d_k in $\mathbf{m}_{\mathbf{d}^{k-1}}$. Let us denote $\eta(\mathbf{m}_{\mathbf{d}^k})$ the cardinality of the E subset belonging to $\mathbf{m}_{\mathbf{d}^k}-\mathbf{m}$. We define the membership function $\mu_{\mathbf{m}_\mathbf{d}}(d_k)$ of a literal d_k in respect to $\mathbf{m}_\mathbf{d}$ as follows:

$$\mu_{\mathbf{m}_\mathbf{d}}(d_k)=1-\frac{\eta(\mathbf{m}_{\mathbf{d}^k})}{\eta(\mathbf{m}_\mathbf{d})} \qquad (1)$$

The monotonicity of the membership function, referred to assignments as points in the unitary hypercube, induces a dummy metrics where points annexed by one literal in $\mathbf{d}$ are more far from the crisp monomial than the ones annexed by previous literals. In a local interpretation of the membership function we can consider $\mu_{\mathbf{m}_\mathbf{d}}(d_k)$ as a probability estimate [7] of finding points that belong to the fuzzy border outside the enlargement induced by d_k, and we can define the radius of the border as the mean value of the distances of points belonging to each enlargement slice from the crisp monomial $\mathbf{m}$ as follows.

Definition 3.2: Given a monomial $\mathbf{m}_i$ and an ordered sequence $\mathbf{d}_i=(d_1,\ldots,d_s)$ of length s of literals from set($\mathbf{m}_i$), we call $\mathbf{m}_\mathbf{d}-\mathbf{m}_i$ the *fuzzy border* of $\mathbf{m}_i$, and

$$\rho_i=\sum_{k=1}^{s}\mu_{\mathbf{m}_\mathbf{d}}(d_k) \qquad (2)$$

its *radius*.

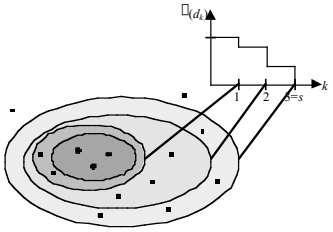


Figure 2. The fuzzy border of a monomial.

Taking into account the labels of the annexed points, we want a positive point close to the crisp region to have a high membership value, while a negative one in analogous position a low value.

This requirement can be met by splitting the membership function as follows

$$\mu^{+}_{\mathbf{m}_\mathbf{d}}(d_k)=\mu_{\mathbf{m}_\mathbf{d}}(d_k)=1-\frac{\eta_+(\mathbf{m}_{\mathbf{d}^k})}{\eta_+(\mathbf{m}_\mathbf{d})}$$

$$\mu^{-}_{\mathbf{m}_\mathbf{d}}(d_k)=\mu_{\mathbf{m}^-_\mathbf{d}}(d_k)=\frac{\eta_-(\mathbf{m}_{\mathbf{d}^k})}{\eta_-(\mathbf{m}_\mathbf{d})} \qquad (3)$$

where $\eta_+(\mathbf{m}_{\mathbf{d}^k})$ and $\eta_-(\mathbf{m}_{\mathbf{d}^k})$ are respectively the total number of positive and negative examples belonging to $\mathbf{m}_{\mathbf{d}^k}-\mathbf{m}$.

Formulas (3) specify the radius notion depending on the labels of the involved points. In particular, we consider also the set $\mathbf{m}^-_\mathbf{d}$ whose membership function $\mu_{\mathbf{m}^-_\mathbf{d}}$ increases monotonically from 0 to 1 when moving from the crisp monomial $\mathbf{m}$ (whose support contains only positive points) to $\mathbf{m}^-$, whose support contains only negative points, and whose radius is $\rho^-_i=\sum_{k=1}^{s}\mu^-_{\mathbf{m}_\mathbf{d}}(d_k)$, accordingly to the radius definition given in (2).

Remark 3.2 Note that the membership functions we defined are not univocal for a given expansion of a monomial from $\mathbf{m}$ to $\mathbf{m}'$. They strictly depend on the history of literal removals we followed. In the next section we further enrich this history by considering also some reinsertion of literals whenever this is suggested by the optimization algorithm. A reinsertion just deletes an item from $\mathbf{d}$ preserving the relative order of the remaining ones.

Remark 3.3 We could define in a similar way the border of the dummy monomial whose crisp support is empty by definition. However, in the following we will consider directly the support of the border as a cost to be minimized and we will estimate it through the intersection of this monomial with E.

3.2 The annealing procedure

The cost function we want to minimize with respect to the DNF f consisting of the m monomials is:

$$O(f,\alpha)=\alpha_1\sum_{i=1}^{m}L_i+\alpha_2\sum_{i=1}^{m}\rho^+_i+\alpha_3\sum_{i=1}^{m}\rho^-_i+\alpha_4\alpha_0 \qquad (4)$$

where:

• $\sum_i L_i$ is the length of the formula, being L_i the number of literals in the i-th monomial

• $\Pi^+_i = \prod_{k=1}^{s} \Pi^+_{m_d}(d_k)$; $\Pi^-_i = \prod_{k=1}^{s} \Pi^-_{m_d}(d_k)$

• β_0 is the percentage of positive examples left out from support(f).

• the free parameter Π guarantees a proper balance between the cost components.

We pursue this goal through a simulated annealing algorithm that drives the search passing through neighboring configurations chosen according to a probability distribution that favors those that involve a lower cost. This probability is however non null over higher-cost ones, in the hope of overcoming local minima. (See Table 2)

Table 2. Pseudo-code for the simulated annealing algorithm.

SA Algorithm

```
CurrentState := InitialState
CurrentTemperature := InitialTemperature
Repeat
  GetTemperature(CoolingSchedule)
  ProposedState := SelectNeighborState
  ProposedCost := EvaluateCost(ProposedState)
  If (Accepted(ProposedState, ProposedCost))
    Then CurrentState := ProposedState
Until StoppingRule
Return(CurrentState)
```

In more detail, we start with a state vector that is the representation, according to Definition 2.1, of the DNF $g = \sum_i m_i$ constructed from the examples as in section 2.

Every move brings the vector in a new state whose Hamming distance from the previous one is 1; only changes involving literals belonging to set(m_i) for i in $\{1,\ldots,k\}$ are allowed. Thus, given a state $\boldsymbol{u}$, the selection of a new state $\boldsymbol{u}'$ consists of selecting one of the allowed literals and flipping its crisp membership value from 1 to 0 or vice versa. The new state is accepted with probability

$$P_{\mathbf{u},\mathbf{u}'} = \frac{1}{1 + \exp\left(\frac{1}{T}\Delta O_{\mathbf{u},\mathbf{u}'}\right)} \qquad (5)$$

where $\Delta O_{\boldsymbol{u},\boldsymbol{u}'}$ is the cost variation obtained when moving from $\boldsymbol{u}$ to $\boldsymbol{u}'$, and T is a positive real value that modulates the acceptance probabilities. High values of T favor the acceptance of any move. As T decreases, the probability distribution concentrates on states with lower cost, and, when T reaches zero, only states of minimum cost are accepted with non null probability. A proper cooling schedule for T is therefore needed to profitably exploit these benefits.

The stopping rule we adopted is to stop after a given number of moves. Our choice of the cooling schedule, of proper values for the free parameters and of the total number of exploring moves will be discussed in the next section in the light of the numerical experiments.

3.3 Numerical experiments

A series of reconstruction tests have been performed with the twofold aim of preliminarily assessing the free parameters of the procedure and appreciating the simplification capability of the introduced fuzzy sets.

The general scheme is to create random DNFs from a generator and then find the closest approximation to the formulas using a confined set of training examples. In particular, we generated formulas in the $\{0,1\}^{12}$ hypercube constituted of 2 to 4 monomials, having 2 to 4 literals each. For each DNF, the test set is constituted by all 4096 assignments to the 12 Boolean variables, each coupled with its membership value to the DNF. The training set is a subset of the test set with some optional features that will be detailed later.

The cooling schedule simply consists of a temperature decrease from T=0.05 to ε (ε<<1) in 6 steps, each of 30**i* iterations and T/*i* temperature, where *i* is the step number. In each experiment two complete cooling cycles are executed. From preliminary observations we set β_4 to 100 and β_3 to 40, and explore β_1 and β_2 as in the Figure 3. We distinguish a false positives percentage given by those negative examples that are annexed to the DNF by a monomial enlargement and a false negatives percentage due to exclusion of positive points after the suppression of a monomial. In the picture we report the mean between these two values computed on the whole search space (the mentioned 4096 points) mediated, in turn, over 30 experiments done for each pair of parameters. Similarly, we evaluated two different ratios of the length of the final formula: a) with respect to the length of the original one generating the example, and b) with respect to the length of the one constructed as in section 2 (which is the starting point of the search process). In Figure 3.*a* we refer to the former. While this ratio obviously decreases with the increase of the β_1/β_2 ratio, the error percentage finds an accentuated minimum in β_1=0.2 and β_2=1.2 that is assumed a good compromise point for jointly having a good compression and approximation.

With this parameter setting we concentrated our experiments, reconstructing 500 new DNFs. Figure 4 reports the mean course of the mentioned errors and compression rates over two cooling cycles with three different compositions of the traininmg set. We either use 50 positive and 50 negative uniformly extracted examples (same biased composition used in the previous

experiments) or simply extract a given amount of points in the search space with the label that the original DNF assignes them (unbiased composition).

Since the support of the original DNFs is generally quite small, the unbiased composition unfavors the learning of it. As a consequence we have more false negatives with this composition, a draw back that is overcome when we increase the training set size from 100 to 400. A similar behaviour is shown by the compression rate curves.

These performances are relatively not bad. It should be noted that in many cases we obtained a more concise formula than the original one. However, although in 14 cases over 500 we rediscovered exactly the original DNF in the biased composition (8 in the unbiased one with 100 examples and 56 with 400 examples in the training set), in some cases these good compression rates are payed by a higher error percentage. A procedure for learning at run time the optimal parameters with which to relax a formula is at moment under assessment.

4. REFERENCES

[1] M. J. Kearns and U. V. Vazirani, An Introduction to Computational Learning Theory, MIT Press, 1994.

[2] M. Li and P. Vitànyi, *An Introduction to Kolmogorov Complexity and its Applications*, Springer, Berlin, 1997.

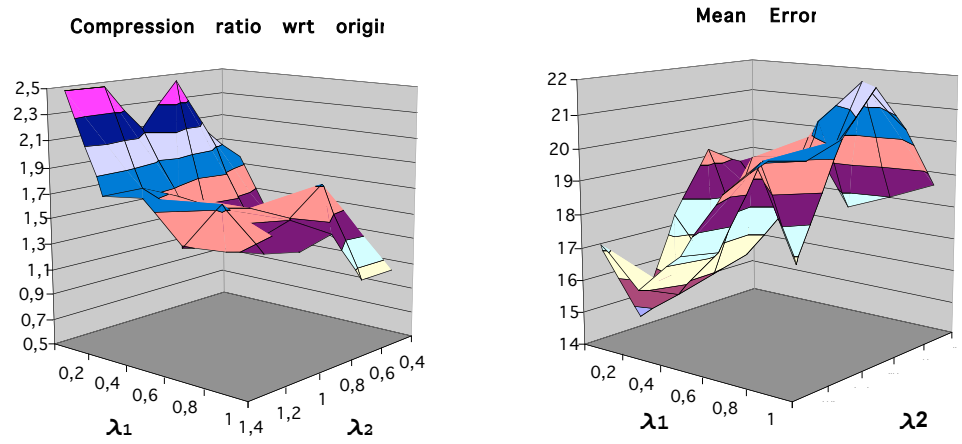
[3] E. Cox, The Fuzzy Systems Handbook, AP Professionals, San Diego, 1998.

[4] V. Vapnik, *The Nature of Statistical Learning Theory*, Springer, New York, 1995.

[5] E. Aarts and J. Korst, *Simulated Annealing and Boltzmann Machines: A Stochastic Approach to Combinatorial Optimization and Neural Computing*, Wiley, 1989.

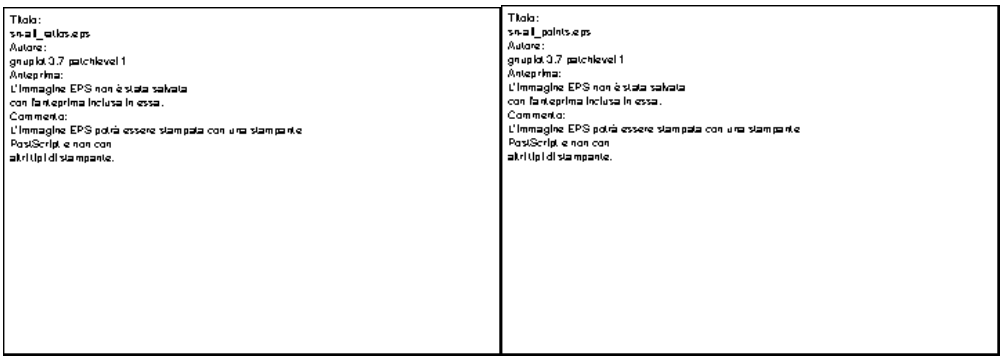
[6] B. Apolloni and C. Gentile, *P*-sufficient statistics for PAC learning *k*-term-DNF formulas through enumeration, Theoretical Computer Science 230 (2000), 1 – 37.

[7] V. K. Rohatgi, An Introduction to Probability Theory and Mathematical Statistics, *Wiley Series in Probability and Mathematical Statistics*, John Wiley & Sons, New York, 1976.



a **b**

Figure 3. Course of compression ratio (a) and mean error (b) with free parameters



a **b**

Figure 4 Compression ratios (a) and accuracy (b) with optimization iterations.