Published in:
ORSA Journal on Computing, Vol. 7, No. 3, pp. 283-285, Summer 1995.

A Relationship Between CNF and DNF Systems Derivable from Examples

Evangelos Triantaphyllou[1] and Allen L. Soyster[2]

1: Assistant Professor. Dept. of Industrial and Manufacturing Systems Engineering, Louisiana State University, 3134C CEBA Building, Baton Rouge, LA 70803-6409, U.S.A.,
E-mail: IETRIAN@LSUVM.SNCC.LSU.EDU

2: Professor and Head. Dept. of Industrial and Management Systems Engineering, Penn State University, 207 Hammond Building, Univ. Park, PA 16802, U.S.A., E-mail: ALSIE@ENGR.PSU.EDU

ABSTRACT. In learning from examples, the main goal is to use a collection of positive examples and a collection of negative examples to derive a Boolean expression which satisfies the requirements imposed by the examples. In order to represent such a Boolean expression, the conjunctive normal form (CNF) and the disjunctive normal form (DNF) have been proposed. This paper makes two contributions. First it shows how to use **any** DNF algorithm to derive a CNF formula (or vice-versa). Furthermore, it demonstrates how to make efficient use of DNF algorithms which cannot handle a large number of positive (or negative) examples by using them as negative (or positive) examples and deriving CNF (or vice-versa). Therefore, the findings of this paper can be used to solve efficiently large scale learning problems. Two learning algorithms are used to illustrate the above issues.

KEY WORDS: Artificial intelligence, learning from examples, satisfiability problem, Boolean expressions, CNF/DNF form.

Learning from examples has attracted the interest of many researchers in recent years. This is mainly due to the belief that any intelligent system should be able to improve its performance over time. In the typical learning problem of this type, both positive and negative examples are available and the main goal is to determine a Boolean expression that accepts all the positive examples, while it rejects all the negative examples. This kind of learning has been examined intensively in the last years (see, for instance, [2], [6], [8], [9], [10], [11], [12], and [13]). Typically the Boolean expression found by the system is either in the conjunctive normal form (CNF) or in the disjunctive normal form (DNF) (see, for instance, [1], [3], [4], [5], [6], [11], and [12]).

The findings of this paper reveal a useful relationship between the CNF and DNF systems derivable from the same data. This relationship can benefit approaches which attempt to solve large problems and use either the CNF or the DNF form in representing a Boolean expression. Suppose that a learning algorithm depends heavily on the number of positive (or negative) examples. If the number of positive (or negative) examples is larger than the number of negative (or positive) examples, then it is more efficient to solve a slightly different problem. That is, apply the same algorithm as before, but instead of using the original examples now use their **complements**. However, now the complements of the original positive (or negative) examples should be treated as the new negative (or positive) examples.

If the original algorithm derives a CNF (or DNF) expression, then in this way it will derive a DNF (or CNF) expression which will satisfy the constraints of the original data. The previous situation is **similar** to the strategy of solving the **dual** of an LP problem. Recall that if the primal LP problem has many constraints and few variables, then the Simplex approach is faster for the dual problem (which will have fewer constraints and many variables).

Let v be an example (either positive or negative). Then, $\bar{v}$ is defined as the **complement** of example v. For instance, if $v = (1,0,0)$, then $\bar{v} = (0,1,1)$. The following definition introduces the concept of the complement of a set of examples. Let E be a collection of examples (either positive or negative). Then, $\bar{E}$ is defined as the **complement of the collection** E.

1. Generating Systems in CNF and DNF Form.

Define the general form of a CNF and DNF system as (I) and (II), respectively.

$$\bigwedge_{j=1}^{k} \left(\bigvee_{i \in \rho_j} a_i \right) \qquad \text{(I)}$$

and

$$\bigvee_{j=1}^{t} \left(\bigwedge_{i \in \rho_j} a_i \right) \qquad \text{(II)}$$

where a_i is either A_i or $\bar{A}_i$. Then the following theorem states an important property which exists when CNF and DNF systems are inferred from collections of positive and negative examples. The proof of this theorem is based on the observation that an example v is accepted (rejected) by a conjunction $\bigwedge_{i \in \rho} a_i$ if and only if the example $\bar{v}$ is rejected (accepted) by the disjunction $\bigvee_{i \in \rho} a_i$.

THEOREM 1: *Let E^+ and E^- be the sets of positive and negative examples, respectively. A CNF system given as (I) satisfies the constraints of the E^+ and E^- sets if and only if the DNF system given as (II) satisfies the constraints of $\bar{E}^-$ (considered as the positive examples) and $\bar{E}^+$ (considered as the negative examples).*

This theorem will be applied to two algorithms from learning by examples. In [11] an algorithm which infers CNF systems from positive and negative examples is developed. In that approach, CNF clauses are generated in a way which attempts to **minimize** the number of CNF clauses that constitute the recommended CNF system. The strategy followed there is called the **One Clause At a Time (OCAT)** approach. A new and enhanced version of the OCAT approach with some extensions is given in [12]. The OCAT approach is sequential. In the first iteration it determines a single clause in CNF form which accepts all the positive examples in the E^+ set while it rejects as many negative examples in E^- as possible. In the second iteration it performs the same task using the original E^+ set but the revised E^- set has only those negative examples which have not been rejected by any clause so far. The iterations continue until a set of clauses is constructed which reject all the negative examples in the original E^- set.

The second algorithm formulates the clause inference problem as a **clause satisfiability (SAT)** problem [6]. In turn, this satisfiability problem is solved by using an interior point method proposed by Karmakar, et al., [7]. Let M_1 and M_2 be the numbers of examples in the E^+ and E^- sets, respectively, and N the number of atoms. In [6] it is shown that given two collections of positive and negative examples, then a DNF system can be inferred to satisfy the requirements of these examples. This approach **pre-assumes** the value of k; the number of conjunctions in the DNF system. In general, this SAT problem has $k(N(M_1 + 1) + M_2) + M_1$ clauses, and $k(2N(1 + M_1) + N M_2 + M_1)$ Boolean variables. Besides the fact that the first algorithm infers CNF systems, while the second infers DNF systems, the two approaches have another major difference. The first approach attempts to minimize the number of disjunctions in the CNF system, while the second approach assumes a given number, say k, of conjunctions in the DNF system and solves a SAT problem. If this SAT problem is infeasible, the conclusion is that there is no DNF system which has k or less conjunctions and satisfies the requirements imposed by the examples.

2. An Example of Deriving CNF and DNF Systems.

Suppose that the following are two collections of positive and negative examples:

$$E^+ = \begin{vmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \end{vmatrix} \quad and \; E^- = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

When the SAT approach is used on the previous data, the resulting satisfiability problem has 31 clauses and 66 Boolean variables (it is assumed that k=2). Since there are more positive examples than negative ones, the **complemented** problem is smaller . It has 58 variables and 26 clauses. The complemented sets are as follows:

$$\bar{E}^- = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad and \; \bar{E}^+ = \begin{vmatrix} 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{vmatrix}$$

When the SAT approach is applied on the sets $\bar{E}^-$ and $\bar{E}^+$ (treating the first set as the positive examples and the second set as the negative examples), then the following DNF system is derived: $(A_1 \Lambda A_2)$ V $(\bar{A}_1 \Lambda$

$\bar{A}_2$). Therefore, according to Theorem 1, the following CNF system satisfies the requirements of the original E^+ and E^- data: $(A_1 \vee A_2) \wedge (\bar{A}_1 \vee \bar{A}_2)$.

Similarly, when the OCAT approach is applied on the previous $\bar{E}^-$ and $\bar{E}^+$ data, then the following CNF system is derived: $(A_1 \vee \bar{A}_2) \wedge (\bar{A}_1 \vee A_2)$. According to Theorem 1, the following DNF system satisfies the requirements of the original E^+ and E^- data: $(A_1 \wedge \bar{A}_2) \vee (\bar{A}_1 \wedge A_2)$.

Table I.

Some Computational Results When N = 30 and the OCAT Approach is Used

$\|E^o\|$	$\|E^+\|$	time	$\|E^o\|$	$\|E^+\|$	time	$\|E^o\|$	$\|E^+\|$	time	$\|E^o\|$	$\|E^+\|$	time
100	1	1	200	10	6	400	5	3	500	35	194
100	3	1	200	11	38	400	7	10	600	8	16
100	5	2	200	18	18	400	10	10	600	11	15
100	5	3	200	19	4	400	15	7	600	23	184
100	7	2	200	51	212	400	16	8	600	44	41
100	7	6	300	2	1	400	17	23	600	49	315
100	7	3	300	3	2	400	36	282	600	83	300
100	8	2	300	14	29	400	47	97	700	13	26
100	9	2	300	14	25	400	49	400	700	18	30
100	15	7	300	17	107	500	6	13	700	19	60
200	1	1	300	22	102	500	7	15	700	19	15
200	2	1	300	22	70	500	13	8	700	56	467
200	4	2	300	24	12	500	16	38	800	64	739
200	5	2	300	36	243	500	20	19	900	72	836
200	6	2	300	71	524	500	34	73	1000	47	80

NOTE: Time is in CPU seconds on an IBM ES/3090-600S machine.

Table I presents some computational results when the OCAT approach is used on random test problems with N = 30 atoms. In this table, $|E^o|$ indicates the total number of examples and $|E^+|$ represents the number of positive examples. To understand the table, consider the third line of the first column. This represents a problem with 5 positive examples and 95 negative examples which required 2 CPU seconds to derive a set of CNF clauses (which accepts all 5 positive examples and rejects the 95 negative examples). For a given problem size, e.g. $|E^o| = 100$, note how the CPU time increases as the number of positive examples, $|E^+|$, increases.

The savings in applying theorem 1 for the SAT approach can be easily determined. Compute the

difference in the number of clauses and variables when M_1 and M_2 are interchanged. This difference is

$$(M_1 - M_2)\,[k\,(N - 1) + 1\,] \qquad \text{(clauses)}$$

and the difference in the number of variables is

$$(M_1 - M_2)\,k\,(N + 1). \qquad \text{(variables)}$$

Hence, when $M_1 > M_2$ (positive examples exceed negative examples) the number of clauses and the number of variables can be reduced by a factor proportional to $(M_1 - M_2)$ and a term of the order of kxN when the positive and negative examples are interchanged. For instance, if $M_1 - M_2 = 50$, $k = 10$ and $N = 20$, the reduction in the number of clauses and variables needed for the SAT approach would be 9,550 and 10,500, respectively.

3. Concluding Remarks.

In this paper we examined an interesting relationship that exists between CNF and DNF systems derivable from collections of positive and negative examples. The findings of this paper can benefit any algorithm which derives CNF or DNF clauses from positive and negative examples. This relationship was demonstrated on two learning algorithms. Furthermore, the findings of this research can lead to increased efficiency for solving large learning problems of the type described in this paper.

REFERENCES

1. T.M. CAVALIER, P.M. PARDALOS, and A.L. SOYSTER, 1990. Modeling and Integer Programming Techniques Applied to Propositional Calculus, *Computers & Operations Research*, J.P. Ignizio (ed.), 17:6, 561-570.
2. T.C. DIETTERICH and R.S. MICHALSKI, 1983. A Comparative Review of Selected Methods for Learning from Examples, R.S. Michalski, J.G. Carbonell, and T.M. Mitchell (eds.). *Machine Learning: An Artificial Intelligence Approach*. Tioga Publishing Company, Palo Alto, CA, 41-81.
3. J.N. HOOKER, 1988. A Quantitative Approach to Logical Inference, *Decision Support Systems*, 45-69.
4. R.G. JEROSLOW, 1989. *Logic-Based Decision Support*, North-Holland, Amsterdam, The Netherlands.
5. A.P. KAMATH, N.K. KARMAKAR, K.G. RAMAKRISHNAN, and M.G.C. RESENDE, 1990.

Computational Experience with an Interior Point Algorithm on the Satisfiability Problem, *Annals of Operations Research*. P.M. Pardalos and J.B. Rosen (eds.). Special issue on: Computational Methods in Global Optimization, 25, 43-58.

6. A.P. KAMATH, N.K. KARMAKAR, K.G. RAMAKRISHNAN, and M.G.C. RESENDE, 1992. A Continuous Approach to Inductive Inference, *Mathematical Programming*, 57:2, 215-238.

7. N. K. KARMAKAR, M.G.C. RESENDE, and K.G. RAMAKRISHNAN, 1991. An Interior Point Algorithm to Solve Computationally Difficult Set Covering Problems, *Mathematical Programming,* 52:3, 597-618.

8. M. KEARNS, MING LI, L. PITT, and L.G. VALIANT, 1987. On the Learnability of Boolean Formulae, *Journal of the ACM*, 9, 285-295.

9. L. PITT, and L.G. VALIANT, 1988. Computational Limitations on Learning from Examples, *Journal of the ACM,* 35:4 965-984.

10. J. R. QUINLAN, 1986. Induction of Decision Trees, *Machine Learning*, 1:1, 81-106.

11. E. TRIANTAPHYLLOU, A.L. SOYSTER, and S.R.T. KUMARA, 1993. Generating Logical Expressions from Positive and Negative Examples via a Branch-And-Bound Approach, *Computers & Operations Research*, 21:2, 185-197.

12. E. TRIANTAPHYLLOU, 1994. Inference of A Minimum Size Boolean Function From Examples by Using A New Efficient Branch-and-Bound Approach, *Journal of Global Optimization*, to appear.

13. L. G. VALIANT, 1984. A Theory of the Learnable, *Communication of the ACM*, 27:11, 1134-1142.