Constructive Induction: A Version Space based Approach

Michèle Sebag

Constructive Induction: A Version Space-based Approach

Michele Sebag
LMS, CNRS UMR 7649, Ecole Polytechnique, 91128 Palaiseau, France
& LRI, bat 490, Universite d'Orsay, 91405 Orsay, France
Michele.SebagCpolytechnique.fr

Abstract

By automatically reformulating the problem domain, constructive induction ideally overcomes the defects of the initial description. The reformulation presented here uses the Version Space primitives $D(E, F)$, defined for any pair of examples E and F, as the set of hypotheses covering E and discriminating F,
From these primitives we derive a polynomial number of M-of-N concept. Experimentally, many of these concepts turn out to be significant and consistent. A simple learning strategy thus consists of exhaustively exploring these concepts, and retaining those with sufficient quality. Tunable complexity is achieved in the *MONKEI* algorithm, by considering a user-supplied number of primitives $D(E_i, F_i)$, where E_i and F_i are stochastically sampled in the training set. *MONKEI* demonstrates good performances on some benchmark problems, and obtains outstanding results on the Predictive Toxicology Evaluation challenge.

1 Introduction

The goal of Machine Learning (ML) is to find a set of hypotheses accurately describing the target concept at hand, and to do so with an acceptable complexity This is made possible only if the learner, the description of the problem domain and the distribution of the training examples fit well together.

When learning small disjuncts [Holte, 1993] for instance, the difficulty might come from the distribution of the examples, and the existence of rare cases [Weiss and Hirsh, 1998]. It might also be due to the lack of relevant primitives in the problem description [Perez and Rendell, 1995] — and indeed new primitives might allow to generalize/cluster rare cases in such a way that they are not "rare" any more. Last, the existence of rare cases might be caused by the learning strategy, e.g. based on set-covering [Michalski, 1983].

Constructive induction traditionally focuses on refining (rewriting) the problem description [Michalski, 1983]. The quality of a reformulation is measured by the improvement of some base learner accuracy. Indeed, expert-driven reformulations of the problem domain can significantly improve the learning performances [Craven and Shavlik, 1993].

Constructive induction is the process of automatically finding a good quality reformulation. A first possibility is to derive the candidate reformulations from rules learned in a previous learning step. Wnek and Michalski [1994] look for new attributes allowing one to compact the previous rules. Gama [1998] uses the prediction of previously learned classifiers as new attributes. Another possibility is to syntactically define the space of candidate reformulations. For instance, MRP explores a set of *relational patterns,* defined as boolean functions of the initial attributes of the problem domain [Perez and Rendell, 1995]. In first order logic, SP searches a set of boolean functions, used to rewrite first-order examples in propositional form [Kramer *et* al., 1998].

These approaches strongly depend on the quality of the knowledge provided to the system (through rules, classifiers or syntactic definitions of the candidate reformulations), which must be relevant and make the search tractable.

To alleviate this limitation, we present a three-step approach interleaving induction and constructive induction: first, some initial hypotheses are constructed from the examples; second, these hypotheses incur a simple reformulation; last, some simple concepts of the reformulated problem are considered, and those satisfying the validation criteria (minimal number of covered examples, maximal number of allowed exceptions) are retained.
This way, the learning workload might be balanced between constructive induction (reformulation) and induction, making it possible to relax their respective requirements: the quality of the initial hypotheses might be low, as these will be reformulated; the complexity of the reformulation might be low, as it is based on hypotheses instead of examples; the last induction step might be rough, as simple worthy concepts are emerged by reformulation. Only attribute-value languages will be considered in the paper.

This approach is rooted in the Version Space (VS) framework, which canonically characterizes the hypotheses solutions of a learning problem [Mitchell, 1982]. In order to give a polynomial characterization of the VS, we have introduced the primitives *D(E, F)* of the VS, defined as the set of hypotheses covering any given example *E* and discriminating any example *F* [Sebag, 1996] (Section 2). Indeed, *D(E,F)* can be viewed as the logical analog of the set of hyper-planes separating *E* from *F*.

The point here is that *D(E,F)* naturally gives rise to an integer attribute noted $h_{E,F}$ (Section 3). Simple concepts built on this attribute (e.g. $[h_{E,F} = M]$) correspond to M-of-N concepts, of the initial domain language. Reformulating the problem domain according to these attributes thus gives access to a polynomial-sized subset of the exponential set of all M-of-N concepts. Experimentally, it turns out that many of these concepts are worthy, i.e. they cover a significant number of examples, and are (almost) consistent. It is then sufficient to evaluate all candidate concepts, and retain those with acceptable significance and consistency.

However, if all primitives *D(E,F)* were considered, the complexity of the approach would be cubic in the number of examples, making it unrealistic to handle medium-to-large datasets. This paper thus presents an algorithm called *MONKEI* (for *M-of-N-based KonstructivE Induction),* using stochastic heuristics to achieve resource bounded induction, along the lines of bounded resource reasoning [Zilberstein, 1996]: the number of considered primitives $D(E_i,F_i)$ is set by the user, and examples E_i and F_i are randomly selected in the training set (Section 4).

Our approach is situated with respect to related work (Section 5), and *MONKEI* is experimentally validated (Section 6). The advantage of this approach is successfully demonstrated on some problems in the Irvine repository [C. Blake and Merz, 1998], and a real-world problem proposed as an IJCAI challenge [Srinivasan, 1997], known as *Predictive Toxicology Evaluation II.*
The main limitation of *MONKEI* is that it provides a DNF theory, less intelligible than standard CNF theories. How to address this limitation, and other perspectives of research, are discussed in the last section.

2 The primitives of Version Space

We assume the reader's familiarity with the Version Space framework [Mitchell, 1982] and its limitations due to an exponential complexity [Haussler, 1988]. As a general remark, the complexity of a concept is commanded by its representation: a concept in DNF form (expressed as a conjunction of disjunctions) corresponds to an exponential concept in CNF form (expressed as a disjunction of conjunctions). One way of having an affordable level of complexity (for both inductive and deductive reasoning) might thus be the use of a DNF formalism instead of a CNF one [Khardon and Roth, 1997].

Along these lines, the Disjunctive Version Space proposes a polynomial DNF characterization of the Version Space [Sebag, 1996]. This characterization is built from elementary hypotheses *D(E,F),* called *Version Space primitives: D(E, F)* is defined as the set of all hypotheses covering *E* and rejecting F, where *E* and *F* are two distinct training examples.

We restrict ourselves to attribute-value logic, where hypotheses are conjunctions of selectors *[att* $\in$ *Interval)* and *[att = value]* respectively built on numerical-and nominal attributes *att.* In this language, the upper-bound of *D(E,F)* is the disjunction (the set) of all maximally general selectors covering *E* and rejecting *F,* termed *maximally discriminant* selectors [Michalski, 1983]. For instance in Table 1, $[Att_3 < 7.28]$ is the maximally discriminant selector built on attribute *Attz.*
By abuse of notations, *D(E, F)* is equated to its upper bound, hence characterized with linear complexity in the number of attributes.

	Att_1	Att_2	Att_3	Att_4	Att_5	Att_6
E	Yes	0	3.45	25	red	
F	No	0	7.28	18	?	blue

$$D(E,F) = [Att_1 = Yes] \vee [Att_3 < 7.28] \vee [Att_4 > 18]$$

Table 1: **Construction of** $D(E,F)$

Let *H* be a conjunction of selectors, and assume that *H* belongs to the version space; by definition *H* is complete (covers all positive training examples) and consistent (rejects all negative examples). It follows that *H* belongs to (is subsumed by at least one selector in) *D(E, F)* for *E* ranging over the set $\mathcal{E}^+$ of positive examples and *F* ranging over the set $\mathcal{E}^-$ of negative examples:

$$H \in VS \quad \Rightarrow H \in \bigwedge_{E\in\mathcal{E}^+,\ F\in\mathcal{E}^-} D(E,F)$$

Inversely, let G denote the above conjunction of *D(E, F).* One can show with no difficulty that all maximally general (conjunctive) hypotheses in G are complete and consistent. The version space and G thus have same upper bound, and G is expressed in DNF form with quadratic complexity in the number of examples and linear complexity in the number of attributes.

Other limitations of Version Spaces due to noisy and sparse data, or disjunctive target concepts, are dealt with by using parameterized combinations of the *D(E, F)* [Sebag, 1996].

3 Learning M-of-N Concepts

This section describes new attributes derived from the Version Space primitives and uses them to reformulate the problem domain. An overview of the *MONKEI* algorithm is then presented.

3.1 Separating concepts

Let $\mathcal{L}$ denote the attribute-value logic description of the problem domain (hypothesis and example language), defined by numerical and/or nominal attributes.

Let *E* and *F* be two training examples, and let *D(E, F)* be constructed as in Section 2, as the set (disjunction) of *N* maximally discriminant selectors Sel_i-

From $D(E, F)$ we derive a mapping $h_{E,F}$ from the problem domain C onto the set of integers: for each example or hypothesis U, $h_{E,F}(U)$ simply counts the number of selectors Sel_i that are satisfied by (covers) U. $h_{E,F}$ (or ft when no ambiguity can arise) maps C onto $[0, N]$: it defines a new computable attribute of the domain.

	Att_1	Att_2	Att_3	Att_4	Att_5	Att_6	$h_{E,F}$
E	Yes	0	3.45	25	red	?	3
F	No	0	7.28	18	?	blue	0
U	?	1	5.28	20	?	blue	2

Table 2: *Attribute* $h_{E,F}$ *maps* $\mathcal{C}$ *onto* [0,3]

Consider the selectors built on attribute ft, (e.g. [ft = Af]), termed *concepts.* By definition, concept [ft = *M]* corresponds to the *M-of-N* concept of selectors Sel_i: an example U satisfies [ft = *M]* (equivalently, $h(U) = M$) iff U satisfies exactly M selectors among the Sel_i.

By construction, concepts [h = 0],..., [ft = *N]* are disjoint and define a partition of the examples.

Attribute $h_{E,F}$ can conveniently be viewed as a new discrete "dimension" of the problem domain. This dimension separates E from F, as these examples belong to opposite regions along this dimension: E belongs to $[h_{E,F} = N]$ since E satisfies all Sel_i; and F belongs to $[h_{E,F} = 0]$ since F satisfies none of the Sel_i.

3.2 Properties

Concepts [ft = *M]* constitute a very flexible hypothesis language, ranging from conjunctive hypotheses (e.g. [ft = *N]* is conjunctive), to XOR patterns (e.g. [ft = 1] corresponds to the *XOR* of selectors Sel_i).

Note that concepts [h = *M]* can cover examples that are syntactically very different, inducing thereby unusual clusters of examples; this might hopefully decrease the number of "rare" cases.

Further, the distribution of the examples along dimension ft (i.e. the number of examples covered by [ft = M], for $M \in [0, N]$) shows an interesting characteristic. If the initial attributes were independent, with probability p_i for any example to satisfy a given selector Sel_i, the Centra! Limit Theorem shows that the distribution of examples along ft tends toward a Gaussian law of mean $\sum p_i$ when N goes to infinity (the approximation being considered accurate for $N > 30$ [Pitman, 1993]). The concepts [ft = *M]* in the tails of the distribution *(M* close to 0 or *N)* would then cover few or no examples.

As could have been expected, experiments show that the initial attributes are *not* independent: concepts in the tails of the distribution happen to cover a significant number of examples. Further, these concepts happen to be consistent, i.e. all or most covered examples belong to the same class. As concepts $[h_{E,F} = M]$ are, sufficiently often, significant and consistent, a simple learning strategy is to exhaustively explore these concepts, and retain all those that are sufficiently good. Further instances U are then classified by a majority vote of the concepts covering U.

The number of such new attributes $h_{E,F}$ is quadratic in the number P of examples; evaluating each $h_{E,F}$ on the training set is linear in the number of attributes and in the number of examples. Hence, the complexity of the exhaustive strategy is cubic in the number of examples, making it unrealistic to handle medium-to-large datasets.

3.3 Overview of *MONKEI*

Tunable complexity is achieved in *MONKEI* by considering a user-supplied number d of primitives $D(E_i, F_i)$, where E_i and F_i are iteratively selected in the training set. The pairs of examples (E_i,F_i), called seed examples, are sampled by a stochastic boosting mechanism, based on the notion of *margin.* Within a majority vote-based classifier, the margin of a training example E is the number of votes for the right class, minus the number of votes for the other class (or the second best class, in case of a multi-class discrimination problem) [Freund and Shapire, 1996]: E is misclassified iff its margin is negative.

In *MONKEI,* one of the seed examples, say E_i, is selected with uniform probability among low-margin *examples* (i.e., whose margin according to the current theory *Th* of the system is less than 1 through the following). The other seed example is selected with uniform probability among the training examples that do not belong to the same class as the first seed example.

After a seed pair (E_i, F_i) has been selected, the corresponding attribute ft is computed for all training examples, and its domain $[0, N]$ is discretized in K intervals $I_1, \ldots, I_K$ (concepts [h = *M]* and [ft = M+1] are merged if they cover examples in the same class). Each concept $[h \in I_k]$ is evaluated; it is added to the current theory if it covers more than a prescribed number A of examples and admits less than a prescribed rate e of exceptions[1]; ft is then termed *dimension* of the domain.

```
Algorithm MONKEI
dimension = 0; Th = {}; IdleSteps = 0
While (dimension < d)
    Draw E among the low margin examples
    If no such E is found, return Th
    Draw F s.t.  class(F) ≠ dass(E)
    Construct attribute h_E,F
    Discretize its domain into intervals I_1,..I_K
    For each I_k
        If [h_E,F ∈ I_k] is selected,
            Add [h_E,F ∈ I_k] to Th
    If at least one [h_E,F ∈ I_k] is selected,
        IdleSteps = 0
        Increment dimension
    Else increment IdleSteps
    If IdleSteps > T return Th
End while
return Th
```

It might happen that many pairs of seed examples (E_i, F_i) are considered and no concept is retained; these

[1] We further require the concept to cover at least one so far misclassified or unclassified example.

steps are named "idle steps". The number of idle steps increases as learning proceeds, as there are less and less low-margin examples, and less chances to cover previously misclassified examples.

MONKEI repeatedly selects pairs of examples until the desired number of dimension *d* is reached, or the number of consecutive idle steps reaches a threshold *T* (set to 100 through the following).

Complexity. The worst case learning complexity is *0{d* x T x *N* x P), where d stands for the user-supplied number of dimensions, *T* is the maximal number of consecutive idle steps allowed, *P* is the number of training examples, and *N* is the number of initial attributes.
At most *d* x *N* concepts are learned. As classifying *U* requires to compute $hE_i,F_i(U)$ only once (with complexity O(*N*)), the classification complexity finally is O(*d* x *N*).

4 Discussion

MONKEI explores a set of concepts that express some relations of the initial attributes, e.g. by counting the number of particular features that are simultaneously satisfied. Compared to the *relational patterns* used in MRP [Perez and Rendell, 1995], the difference is that the concepts explored here are automatically and polynomiaUy derived from the examples, and no preliminary discretization of numerical domains is required.
The constructed M-of-iV concepts are simply evaluated; as opposed to MRP [Perez and Rendell, 1995] or ID2-of-3 [Murphy and Pazzani, 1991], *MONKEI does* not consider their combination (conjunction).

MONKEI must also be compared to Support Vector Machines (SVMs) [Scholkopf *et* oi., 1998], which reformulate the problem domain using kernel functions derived from the examples. In the new description space, SVMs look for a separating surface optimizing the minimal margin over the training set; the optimization proceeds by pruning all examples but those with a minimal margin, called *support vectors.*
One major difference between SVM and *MONKEI is* that every kernel function used in SVMs depends on a single example; every "kernel function" $h_{E,F}$ used in *MONKEI* depends on a pair of examples. Another difference is that SVMs start with a large set of kernel functions which is gradually pruned along the optimization of the minimal margin. In opposition, *MONKEY* gradually grows the set of kernel functions until a satisfactory margin has been found for all examples - or the computational resources have been exhausted.

Like boosting algorithms [Preund and Shapire, 1996], *MONKEI pays* more attention to misclassified examples than to others; if naively done, this strategy might lead to rewarding noisy examples. This drawback is limited in *MONKEI as* all constructed concepts are independently validated: only sufficiently good concepts can be added to the current theory.

One main limitation of *MONKEI* is that it is unlikely to deal with irrelevant attributes. For any example U, let *h(U)* be decomposed as $h_{rel}(U) + h_{noise}(U)$, where $h_{rel}(U)$ and $h_{noise}(U)$ respectively denote the number of discriminant selectors based on relevant and irrelevant attributes that are satisfied by U; h_{noise} can be viewed as some kind of noise which blurs the information contained in h_{rel}, making it unlikely to discover the worthwhile concepts $[h_{rel} = M]$. Additional heuristics need be designed to overcome this problem.

Another limitation, discussed in Section 5.3, regards the intelligibility of the theory produced by *MONKEI.*

5 Experimental validation

Since a universal learner does not exist [Wolpert and Macready, 1995], experimental validation should make clear when a new learner is worth using.

5.1 Benchmark problems

We first consider six problems (artificial waveform [Breiman *et* al., 1984], glass, balance, tic-tac-toe, monks-2, vehicle) from the Irvine repository [C. Blake and Merz, 1998], illustrating various types of learning difficulties (noisy data, ill-distributed classes, many classes, many disjuncts).

	Train.	#class/att	previous best
Balance	625	3/4	93% ± 3 (LCG)
Glass	214	6/10	73% ± 7 (LCG)
Monks2	169	2/6	100 (HCI)
Tictactoe	956	2/9	99% ± .6 (G-NET)
Vehicle	846	4/18	79 % ± 3 (LCG)
Wave	10 * 300	3/21	84% ± 3 (LCG)

Table 3: *Datasets and Reference Results.*

Table 3 recalls the characteristics of the datasets and the previous best results, obtained (as far as we know) by *Local Cascade Generalization* [Gama, 1998], *G-Net* [Anglano *et* al., 1998] and *HCI* [Wnek and Michakki, 1994] using a 10-fold cross-validation.

The experiment goal here is to check whether competitive results can be obtained for a reasonable number of dimensions. The number *d* of considered dimensions is chosen from the interval [10,100]. Other parameters of *MONKEI* are frozen to their default value: the minimal number *A* of examples covered by a concept is 5, the maximal percentage of exceptions e is 10%.

On the balance, glass, tic-tac-toe and vehicle problems, *MONKEI* accuracy is evaluated by 2-fold cross-validation, averaged over 5 independent splits of the dataset as recommended by [Dietterich, 1998]. 15 independent runs are executed for each split of the dataset, as recommended when evaluating a stochastic algorithm.
On the Monks2 problem, the accuracy is evaluated on the 432-example test set (averaged on 15 independent runs). On the waveform problem, the predictive accuracy is evaluated on a 5000-examples test set, averaged over ten 300-examples training sets (15 independent runs are executed for each training set).
CPU times are given in seconds on a Pentium 11-166.
Table 4 shows the lowest number of dimensions d allowing *MONKEI* to match or outperform the state-of-the-art results on the balance, glass, vehicle and monks-2

problems. #C denotes the number of Af-of-iV learned concepts.

	Accuracy	CPU	d	# C
Balance	97.07 ± 1.6	< 1	20	22.4
Glass	89.67 ± 4.7	< 1	30	20.9
Monks2	98.3 ± 1.4	< 1	40	22.5
Vehicle	86.33 ± 2.1	6	80	90.3

for the lowest d matching reference results

Table 5 illustrates how the performance depends upon the number of dimensions, on the waveform and tic-tac-toe problems. *MONKEI* matches the optimal theoretical accuracy on the waveform problem (86%), but falls behind *G-Net* on the tic-tac-toe problem. Some care must be exercised when comparing the results, since the reference results were obtained according to a 10-fold CV against a 2-fold CV for ours (meaning that *MONKEI* is evaluated with a more pessimistic estimate).

	Waveform		Tic-tac-toe	
d	Accur.	CPU	Accur.	CPU
20	62.55 ± 6	8	91.76 ± 2.4	< 1
40	78.90 ± 4	14	96.8 ± 1.1	1
60	85.65 ± 2	18	97 ± 1	1

Table 5: *MONKEI Results (Dependence upon d)*

Generally, good results can already be achieved using comparatively small values of *d* (*d* > 60 for the waveform problem, *d* > 80 for the vehicle problem, *d* > 40 otherwise).

5.2 A real-world problem

The real-world Predictive Toxicology Evaluation (PTE2) problem is nicely motivated and detailed in [Srinivasan, 1997]. This problem both is very inspiring and appears difficult for experts, learners, and even learners cooperating with experts [Srinivasan, 1999].

This dataset includes 333 examples. The "test" set includes 30 examples which were unknown at the beginning of the PTE challenge; 20 have been since diagnosed and their class was available by November 1998. FOIL [Quinlan, 1990] and PROGOL [Muggleton, 1995] consider a first-order description of the problem, other learners consider a 417 attributes description. A comprehensive presentation of the descriptions and the reference results is found in [Srinivasan, 1999].

Representation	Algorithm	Accuracy
relational	FOIL	25.15%
	PROGOL	63 %
propositional	C4.5 prune	58.79%
	C4.5 rules	60.76%
	C4.5 rules 4- hand	78%

Table 6: *Reference Results on PTE2*

Given the practical importance of the problem, *MONKEI* was run with unbounded resources, meaning that all primitives *D(E,* F), for *E* and *F* respectively ranging over the positive and negative examples, were considered. The CPU time is 24 minutes on a Pentium II-300.

The selection of concepts thus only depends on the minimal number of covered examples *A*, chosen from the interval [30,50] and the maximal rate of exceptions *e* chosen from the interval [0,10%].

The results (Table 7) demonstrate the good performances of *MONKEI,* and its stability with respect to parameters *A* and ε. #C denotes the number of M-of-*N* learned concepts.

£	A	"Test"	Training	# C
0	30	100%	100%	188
0	50	100%	100%	69
5	30	100%	100%	356
5	50	100%	100%	104
10	30	95%	94%	1099
10	50	95%	98%	192

Table 7: *Unbounded MONKEI on PTES*

5.3 Discussion

In the field of scientific discovery, a major drawback of *MONKEI is* that it fails to produce an intelligible theory: this failure is basically due to the fact that it handles DNF concepts, which are generally considered to be less intelligible than standard rulesets, i.e. CNF concepts.

Still, an intelligible interface can be constructed on a DNF-based system [Khardon and Roth, 1997].

In *MONKEI* (as in SVMs), the system can answer the user's questions about the typicality of examples. The typicality of an example, interpreted with regard to its margin, can be computed (instead of, explained) from the theory. The experiment design might take advantage of this information, to preferably run physical experiments corresponding to borderline (untypical) examples.

The logic-based formalism of *MONKEI* can facilitate the detection of attribute dependencies, as it allows one to focus on the subsets of examples that are covered by the concepts, and the selectors that are simultaneously satisfied by the examples.

One might also focus on the selectors that are never simultaneously satisfied in these conditions, and search for XOR subconcepts. The favorable case is when all M-of-iV concepts can be expressed as conjunctions of simple XOR subconcepts; the theory would then directly be intelligible.

6 Conclusion and Perspectives

The approach investigated in this paper considers *DNF* concepts as hypothesis language; the advantage of the language is its high expressiveness (both conjunctive and XOR expressions are DNF concepts), and a low computational complexity.

Within the version space framework, we define a particular set of DNF concepts, polynomially characterized and evaluated from the examples. This set is explored by stochastic sampling, in order to let the user control the learning cost (any-time algorithm). All explored concepts that are "sufficiently good" are retained. This strategy demonstrates its efficiency on several Irvine problems, as it matches or outpasses the previous best results for quite a limited amount of resources. On the real-world PTE2 problem, *MONKEI* achieves outstanding results with unlimited resources, still in reasonable time.

This work opens up several perspectives. One already mentioned is to develop an "intelligible" interface for *MONKEI.* Another perspective is to upgrade *MONKEI* to first order logic, using the polynomial approximations of the Version Space relational primitives developed in [Sebag and Rouveirol, 1997].

Last, we shall examine in more detail the relationship between *MONKEI and* Support Vector Machines. Using *d* pairs of seeds (E_i , F_i), *MONKEI maps* the initial training set onto N^d; this makes it possible to apply Support Vector Machines, and determine the support vectors examples *Vj.* An interesting question is how, if any, the support vectors *Vj* are related to the seeds E_i and F_i.

Acknowledgments

Many thanks to Yves Kodratoff and Fabien Torre, LRI, and Marc Schoenauer, Ecole Polytechnique, for their support and many discussions. Thanks also to Lise Fontaine and the anonymous referees, for their help in making the paper readable.

References

[Anglano *et al.,* 1998] C. Anglano, A. Giordana, G. Lo Bello, and L. Saitta. An experimental evaluation of coevolutive concept learning. In J. Shavlik, ed., *Proc. of the 15th ICML.,* pages 19-27. Morgan Kaufmann, 1998.

[Breiman *et al*., 1984] L. Breiman, J.H. Friedman, R.A. 01-shen, and C.J. Stone. *Classification and Regression by tree.* Wadsworth, Belmont California, 1984.

[C. Blake and Merz, 1998] E. Keogh C. Blake and C.J. Merz. *UCI Repository of machine learning databases. http://www.ics.uci.edu/-mlearn/MLRepository.html,* 1998.

[Craven and Shavlik, 1993] M.W. Craven and J.W. Shavlik. Learning to represent codons: A challenge problem for constructive induction. In *Proc. of IJCAI-93,* pages 1319-1324. Morgan Kaufmann, 1993.

[Dietterich, 1998] T.G. Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation,* to appear, 1998.

[Freund and Shapire, 1996] Y. Preund and R.E. Shapire. Experiments with a new boosting algorithm. In *Proc. of the 13th ICML.,* pages 148-156. Morgan Kaufmann, 1996.

[Gama, 1998] J. Gama. Local cascade generalization. In J. Shavlik, ed., *Proc. of the 15th ICML.,* pages 206-214. Morgan Kaufmann, 1998.

[Haussler, 1988] D. Haussler. Quantifying inductive bias : AI learning algorithms and Valiant's learning framework. *Artificial Intelligence,* 36:177-221, 1988.

[Hoite, 1993] R.C. Holte. Very simple classification rules perform well on most commonly used datasets. *Machine Learning,* 11:63-90, 1993.

[Khardon and Roth, 1997] R. Khardon and D. Roth. Learning to reason. *Jal of the ACM,* 44(5):697-725, 1997.

[Kramer *et al.,* 1998] S. Kramer, B. Pfahringer, and C.-Helma. Stochastic propositionalization of non-determinate background knowledge. In D. Page, ed., *Proc. of Inductive Logic Programming'98,* pages 29-61, 1998.

[Michalski, 1983] R.S. Michalski. A theory and methodology of inductive learning. In R.S Michalski, J.G. Carbonell, and T.M. Mitchell, eds, *Machine Learning: an artificial intelligence approach,* volume 1, pages 83-134. Morgan Kaufmann, 1983.

[Mitchell, 1982] T.M. Mitchell. Generalization as search. *Artificial Intelligence,* 18:203-226, 1982.

[Muggleton, 1995] S. Muggleton. Inverse entailment and PROGOL. *New Gen. Comput.,* 13:245-286, 1995.

[Murphy and Pazzani, 1991] P.M. Murphy and M.J. Pazzani. Id2-of-3: Constructive induction of M-of-N concepts for discriminators in decision trees. In L.A. Birnbaum and G.C. Collins, eds, *Proc. of the 8th IWML,* pages 183-187. Morgan Kaufmann, 1991.

[Perez and Rendell, 1995] E. Perez and L.A. Rendell. Using multidimensional projection to find relations. In *Proc. of the 12th ICML.,* pages 447-455. Morgan Kaufmann, 1995.

[Pitman, 1993] J. Pitman. *Probability.* Springer Verlag, 1993.

[Quinlan, 1990] J.R. Quinlan. Learning logical definition from relations. *Machine Learning,* 5:239-266, 1990.

[Quinlan, 1993] J.R. Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann, 1993.

[Scholkopf *et al,* 1998] B. Scholkopf, C. Burgess, and A. Smola. *Advances in Kernel Methods.* MIT Press, 1998.

[Sebag and Rouveirol, 1997] M. Sebag and C. Rouveirol. Tractable induction and classification in FOL via Stochastic Matching. In *Proc. of IJCAI-97,* pages 888-892. Morgan Kaufmann, 1997.

[Sebag, 1996] M. Sebag. Delaying the choice of bias: A disjunctive version space approach. In L. Saitta, ed., *Proc. of the 13th ICML.,* pages 444-452. Morgan Kaufmann, 1996.

[Srinivasan, 1997] A. Srinivasan. The predictive toxicology evaluation challenge. In *Proc. of IJCAI-97,* pages 4-8. Morgan Kaufmann, 1997.

[Srinivasan, 1999] A. Srinivasan. *PTE-Challenge,* 1999. http://wwwxomlab.ox.ac.uk/oucl/groups/machlearn/PTE,

[Weiss and Hirsh, 1998] G.M. Weiss and H. Hirsh. The problem with noise and small disjuncts. In J. Shavlik, ed., *Proc. of the 15th ICML.,* pages 574-578. Morgan Kaufmann, 1998.

[Wnek and Michalski, 1994] J. Wnek and R.S. Michalski. Discovering representation space transformations for learning concept descriptions combining DNF and M-of-N rules. In T. Fawcett, ed., *Workshop on Constructive Induction, ICML-94,* 1994.

[Wolpert and Macready, 1995] D.H. Wolpert and W.G. Macready. No free lunch theorems for search. Technical report, Santa Fe Institute, 1995.

[Zilberstein, 1996] S. Zilberstein. Resource-bounded reasoning in intelligent systems. *Computing Surveys,* 28(4), 1996.