Learning to Classify Incomplete Examples

Dale Schuurmans
Department of Computer Science
University of Toronto
Toronto, ON M5S 1A4
dale@cs.toronto.edu

Russell Greiner
Siemens Corporate Research
Princeton, NJ 08540
greiner@scr.siemens.com

Abstract

Most research on supervised learning assumes the attributes of training and test examples are *completely* specified. Real-world data, however, is often incomplete. This paper studies the task of learning to classify *incomplete* test examples, given incomplete (resp., complete) training data.

We first show that the performance task of classifying incomplete examples requires the use of *default* classification functions which demonstrate nonmonotonic classification behavior. We then extend the standard PAC-learning model to allow attribute values to be hidden from the classifier, investigate the robustness of various learning strategies, and study the sample complexity of learning classes of default classification functions from examples.

1 Introduction

The central task of most expert systems is classifying objects from some domain of application; *i.e.*, determining whether a particular object belongs to a specified class, given a description of that object (Clancey, 1985). For example, a clinician must decide whether a patient, with a specified set of symptoms, has a particular disease; a chess player must determine whether a particular move is appropriate given a board configuration; and a planner must determine whether to apply a particular action, given the perceived state. In practice, it is often difficult to specify an expert classifier directly (*e.g.*, by explicitly extracting the knowledge of domain experts), so it is often advantageous to automatically *learn* an appropriate classifier from existing "solved" cases (*i.e.*, supervised learning). In fact, most successful real world applications of machine learning follow this paradigm.

Virtually all research on supervised learning addresses the task of learning to classify *completely* described domain objects. However, in many real world situations we often have to classify objects given only *incomplete* descriptions of their attributes. For example, in medical diagnosis applications doctors seldom have access to every potentially relevant fact about a patient (Porter, Bareiss and Holte, 1990). Here the patient is usually better off if the doctor makes a credulous assessment and suggests some treatment based on what is known, rather than skeptically withholding judgement until more information is obtained.

Many expert systems must classify such partially-described domain objects, even when the available data is insufficient to unequivocally determine the appropriate classification.

Such *incomplete*-data classifiers differ from standard *complete* data classifiers in a fundamental way: Since complete data classifiers specify necessary and sufficient conditions for each class, they are unable to categorically classify "ambiguous" incomplete descriptions.[1] By contrast, incomplete data classifiers can propose "default classifications" in such cases. Since these default classifications may change in light of further information about a domain object, incomplete-data classifiers are able to exhibit "nonmonotonic classification behavior" (Reiter, 1987). This type of nonmonotonic classification behavior cannot be described in terms of necessary and sufficient conditions, and hence cannot be encoded by a complete data classifier (Schuurmans and Greiner, 1994).

The task of learning an accurate incomplete-data classifier from examples raises a number of new issues which have not been addressed by traditional supervised learning research. First, we have to consider the types of processes that can cause an object attribute to be unobserved — for example whether this omission is uninformative, partially informative, or even misleading. We will see that different assumptions here affect the types of training strategies an effective learner should employ. Second, we can consider training on the natural source of *incomplete* examples, versus training on the corresponding artificially-*completed* examples. Intuitively, complete descriptions give the learner more information about each object, and hence, should make learning easier. We will see however that this intuition is only sometimes correct.

Although the task of learning classifiers for incomplete data has been considered in a few empirical studies (Porter, Bareiss and Holte, 1990; Quinlan, 1989; Breiman et al., 1984), the results of these studies have invariably been mixed: a number of techniques for handling missing data have been investigated, but none has been found to be uniformly superior to the others. Part of the problem is that this learning task has yet to receive the same degree of theoretical treatment as learning complete data classifiers; so we have no explanation of this phenomenon, nor any indication of which technique is best under specific circumstances. We attempt to fill this void by studying the task of learning accurate default concepts in a precise mathematical framework. The intent is to uncover a theoretical explanation of the observed empirical phenomena, and provide effective guidance for practice.

Overview: We develop a theoretical framework along the lines of Valiant's PAC-learning framework for learning complete data classifiers (Valiant, 1984). After introducing the basic classification framework in Section 2, Section 3 then extends Valiant's random example model to incorporate an "attribute blocking process". Section 4 summarizes our results regarding the efficacy of various learning strategies under particular circumstances, and Section 5 investigates the sample complexity of "PAC-learning" classes of incomplete data classifiers. Finally, Section 6 discusses the main contributions and suggests directions for future research.

This work provides a general, unifying framework for studying the task of learning incomplete data classifiers. By making various modeling assumptions explicit we are able to analyze various learning strategies and determine which are best (and which fail) under specific conditions, and also determine which conditions are conducive to efficient learning.

[1] An "ambiguous" incomplete description is a partial description that can be completed (*e.g.*, by filling in values for the currently-unspecified attributes) in different ways, to produce different classifications.

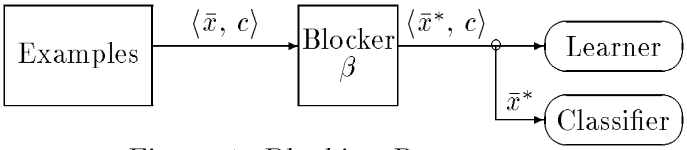

Figure 1: Blocking Process

These observations are important for synthesizing effective learning systems for real world applications where missing data is prevalent.

Related Work: Valiant's PAC-learning model (Valiant, 1984), and the subsequent research it inspired (Blumer et al., 1989; Haussler, 1992), has greatly advanced our understanding of learning complete data classifiers from examples. Many generalizations of Valiant's framework have been introduced in the learning theory literature, including: learning from noisy concept labels (Angluin and Laird, 1988; Kearns and Li, 1988), learning with attribute noise (Shackelford and Volper, 1988), learning probabilistic concepts (Kearns and Schapire, 1990), and general decision-theoretic learning (Haussler, 1992). However, our task differs from each of these, in that we address a fundamentally different form of classification task. For example, a system that learns with attribute noise will not know which attribute values have been corrupted; by contrast, we know explicitly which values are missing. Also, a system that learns a probabilistic concept will produce a mapping from the space of *complete* object descriptions to probability values; such a mapping does not directly handle missing attribute values.[2]

Classifying incomplete examples is closely related to the problem of reasoning from partial information, a problem which has been extensively studied in the default and nonmonotonic reasoning literature, *cf.*, (Reiter, 1987; Ginsberg, 1987). There are a number of connections between the problem of learning default concept definitions from examples and related work in AI and philosophy of statistics (Kyburg, 1991); these are explored in detail elsewhere (Schuurmans and Greiner, 1994).

2 Missing-data classifiers

We begin by formalizing the performance task of classifying incomplete examples, before going on to the task of learning such classifiers. For simplicity, we assume each domain object is described by a vector of boolean attributes $\bar{x} = \langle x_1, \ldots, x_n \rangle$, $x_i \in \{0, 1\}$; thus, the set of domain objects is given by $X_n = \{0, 1\}^n$. Also, we consider a simple two-category classification task, where domain objects are classified as members or non-members of some *concept* over the domain. A (complete) test example is specified by a pair $\langle \bar{x}, c \rangle$, consisting of a domain object $\bar{x} \in X_n$ and its actual class $c \in \{0, 1\}$. In standard classification models, this domain object $\bar{x}$ would be passed "as is" to the classifier; here, however, we assume the classifier only sees a degraded version of $\bar{x}$ in which certain attribute values have been

[2](Kearns and Schapire, 1990) consider the problem of learning the probabilistic concept induced by observing a fixed subset of the available attributes. Note that this does not address the issue at hand, as they are in effect learning a complete-data classifier over the fixed subset of visible attributes; clearly there is no potential for nonmonotonicity here.

replaced by the "unknown" value $*$. Thus, the set of possible object *descriptions* is given by $X_n^* = \{0, 1, *\}^n$. We model this degradation using a (possibly stochastic) *blocking process* β that may hide some of the attribute values: replacing certain values with $*$, but otherwise leaving $\bar{x}$ intact; see Figure 1. Thus, $x_i = 0$ can get mapped to $x_i^* \in \{0, *\}$, and $x_i = 1$ to $x_i^* \in \{1, *\}$. An *example* $\langle \bar{x}^*, c \rangle$ then consists of a (partial) description $\bar{x}^*$ of some domain object $\bar{x}$, along with $\bar{x}$'s true classification $c \in \{0, 1\}$; so the space of possible examples is given by $X_n^* \times \{0, 1\}$.

A *complete-data classifier* (CDC) is an indicator function $c\colon X_n \to \{0, 1\}$ for some concept over the domain, *i.e.*, $c(\bar{x}) = 1$ iff the *complete* description $\bar{x}$ belongs to the concept. A *missing-data classifier* (MDC) $d\colon X_n^* \to \{0, 1\}$, on the other hand, takes a *partial* description $\bar{x}^*$ as its input and returns $d(\bar{x}^*) = 1$ if the object described by $\bar{x}^*$ belongs to the concept *by default*, and returns $d(\bar{x}^*) = 0$ otherwise. Given a test example $\langle \bar{x}^*, c \rangle$, an MDC d makes a *correct* classification if $d(\bar{x}^*) = c$, otherwise it makes an *error*.

Structure and representation: Abstractly, an MDC is just a function $d\colon X_n^* \to \{0, 1\}$. Notice we are assuming d makes a classification given any possible object description (even $\bar{x}^* = \langle *, *, \ldots, * \rangle$), so there are 2^{3^n} distinct MDCs possible on n boolean attributes. Representing an MDC on X_n^* is far more complicated than just representing a CDC on X_n. MDCs have been implemented in a number of different ways in practice, based on different conceptualizations of the structure of a default classifier.

Perhaps the most common strategy for representing an MDC is based on first filling in the missing attributes with some default values, and then applying a standard CDC to the completed description (Quinlan, 1989; Little and Rubin, 1987; Breiman et al., 1984).

IM (Imputation) Represent a MDC d by a single CDC c and a list of default attribute values $v_1, v_2, \ldots, v_n$. To classify a description $\bar{x}^*$, first fill in each of $\bar{x}^*$'s missing attributes $x_i^* = *$ with its default value v_i, obtaining a complete description $\bar{x}$, and then determine $c(\bar{x})$.

Unfortunately, this technique is quite limited in the range of MDCs it can actually represent (see Proposition 1 below), which has lead many researchers to consider alternative, more general representation strategies. For example, one can implicitly represent a MDC over X_n by defining a total joint probability distribution P_{XC} over the space of complete examples $X_n \times \{0, 1\}$.

JT (Joint distribution) Represent a MDC d by a domain distribution P_{XC}. To classify a description $\bar{x}^*$, determine the most likely class given $\bar{x}^*$'s observed attributes.

Popular techniques for representing joint distributions over $\{0, 1\}^n$ include Bayes and Markov nets (Neal, 1992; Pearl, 1988), and Bernoulli mixture models (Ghahramani and Jordan, 1994). These representations provide compact and intuitive representations of MDCs, but they have the drawback that actually determining their classifications can often be computationally expensive (Roth, 1993; Pearl, 1988). Furthermore, this strategy is still not fully general in the range of MDCs it can represent; for example, JT can only represent MDCs satisfying a certain "inheritance" constraint in the hierarchy of default classifications (Schuurmans and Greiner, 1994).

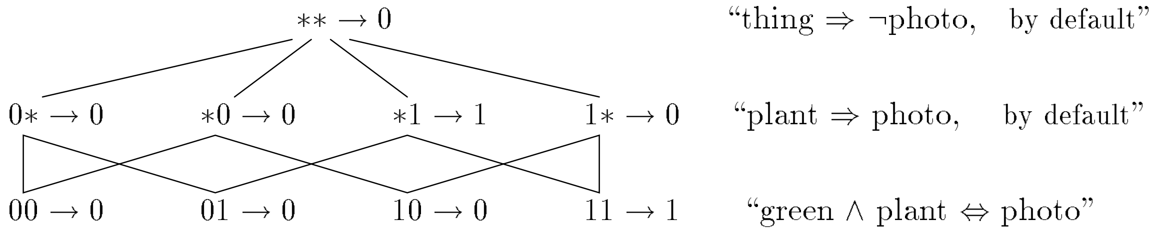

Figure 2: DR representation of an MDC

The simplest way to conceptualize an MDC is just as a direct mapping from partial descriptions $\bar{x}^*$ to classifications. In fact, this approach can be implemented simply by treating $*$ as a third attribute value, and specifying a classification for each description $\bar{x}^*$; a strategy which is often adopted in decision tree approaches to missing data classification (Quinlan, 1989; Breiman et al., 1984).

DR (Default rules) Represent a MDC d as a collection of 3^n default classification rules of the form $\bar{x}^* \to c$, where for each description $\bar{x}^* \in X_n^*$ either $\bar{x}^* \to 1 \in d$ or $\bar{x}^* \to 0 \in d$ but not both. A description $\bar{x}^*$ is then classified according to the matching rule $\bar{x}^* \to c \in d$.

To illustrate, consider the example of an MDC on two attributes shown in Figure 2, where the first attribute is "green", the second "plant", and the class is "photosynthetic". Each node in the graph represents a rule; *e.g.*, $*1 \to 1$ encodes the rule that a plant, of unspecified color, is classified as photosynthetic. Notice this collection of rules specifies *nonmonotonic* classification behavior, as its assessment of concept membership can change as more attributes are specified. For example, even though non-green–plants $\subset$ plants $\subset$ things, the predicted photosynthesis properties are 0, 1, 0, respectively, which means that such a classifier cannot be specified by a CDC. This default hierarchy perspective actually provides a revealing view of the structure of a MDC; *e.g.*, as in the JT example above.

An alternative view of MDCs, which will prove useful in Section 5, is to think of a MDC as a collection of CDCs, one defined on each subset of observed attributes.

CL (Classifier lattice) Represent an MDC d as a set of 2^n complete-data classifiers, one defined on each subset s of visible attributes. A description $\bar{x}^*$ is classified according to the CDC d_s defined on the observed set of attributes.

To illustrate, Figure 3 shows the CL representation for the same MDC on two attributes as shown in Figure 2. Of course explicitly representing 2^n CDCs is infeasible in general, but

Observed attributes	Local CDC	
$\{\quad\}$	$c = 0$	"thing ⇒ ¬photo, by default"
$\{x_1\quad\}$	$c = 0$	"green ⇒ ¬photo, by default"
$\{\quad x_2\}$	$c = x_2$	"plant ⇒ photo, by default"
$\{x_1,\ x_2\}$	$c = x_1 \wedge x_2$	"green ∧ plant ⇔ photo"

Figure 3: CL representation of an MDC

feasible representations can be based on choosing only a small subset of the attributes as relevant and only representing CDCs on those attributes.

There are clearly a wide number of ways to represent MDCs. In fact, other representations of MDCs are possible; for example based on finding best matches to prototype objects (Porter, Bareiss and Holte, 1990) (*i.e.*, case-based reasoning), but these will not be considered here. The choice among these many different representation paradigms is largely a matter of taste and/or convenience, and should be dictated by the application at hand. However, an important observation is that these strategies differ in their fundamental representation capacities.

Proposition 1 IM $\subset$ JT $\subset$ DR $\equiv$ CL *(inclusions are strict).*

There are many unexpected similarities between MDCs and existing nonmonotonic knowledge representation formalisms; see (Schuurmans and Greiner, 1994) for more details. We now turn to the problem of *learning* effective MDCs from training examples.

3 Learning missing-data classifiers

Our model of supervised learning in this missing-data context is analogous to the standard complete-data case: given a sequence of training examples $\langle \bar{x}_1, c_1 \rangle, \langle \bar{x}_2, c_2 \rangle, \ldots, \langle \bar{x}_m, c_m \rangle$, consisting of (possibly incomplete) object descriptions and their correct class labels, the learner L must produce a MDC $d : X_n^* \rightarrow \{0,1\}$. Ignoring computational details, a learner L is just a mapping from labeled training sequences to MDCs.

Training examples: Unlike standard CDC learning, however, it is natural to consider two different types of training examples in this setting: *incomplete* training examples (χ_I), where the teacher provides the correct class labels, but otherwise leaves the descriptions incomplete; and *complete* training examples (χ_C), where in addition to providing the correct class labels, the teacher also "fills in" the missing attribute values for each training object. Even though our goal is to learn MDCs that classify incomplete examples, it may make sense to consider learning from complete examples in many real-world domains. For example, a medical student may be trained to diagnose the presence of a particular disease given fairly complete descriptions of all relevant patient data, and yet as a doctor, be expected to produce diagnoses without the benefit (and cost) of performing every available diagnostic test. Intuitively we expect an advantage in training on complete examples as they appear to provide more information than incomplete examples, however this intuition turns out to be only sometimes correct. Alternatively, in many practical settings it may be impossible to obtain complete training examples, meaning the learner will have to learn from incomplete examples. One benefit of training on such partial examples is that the learner is exposed to the natural blocking process operating in the domain.

Biases and learning strategies: To achieve reasonable learning performance using only feasible amounts of training data, we eventually have to introduce some form of prior knowledge to constrain our learning systems. This points to the necessity of *bias*: in any successful application, the learning system must be constrained to search a restricted class of appropriate classifiers (Mitchell, 1980).

Domain distribution	$\boxed{\mathrm{P}_{XC}}$	over domain objects
	$\Downarrow$	
Observation process	$\boxed{\beta}$	*loses* information
	$\Downarrow$	
Observed distribution	$\boxed{\mathrm{P}_{X*C}}$	over object *descriptions*

Figure 4: Observation Process

Most empirical learning techniques for MDCs include two components: a technique for representing a *bias* (here a class of MDCs $\mathcal{D} = \{d\}$), and a *learning rule* for choosing some $d \in \mathcal{D}$ given the training examples. In fact, learning systems have been developed that adopt each of the MDC representation techniques discussed in the previous section.

IM (Imputation) *Bias:* A class of CDCs $\mathcal{C}$ and the set of possible default-value vectors $\mathcal{V}$ (which implicitly defines a class of MDCs $\mathcal{D}$).
Learning rule: Use the training set to estimate the most likely default value for each individual attribute, fill in the missing attribute values for each instance in the training set, and then choose a minimum error $c \in \mathcal{C}$ (Quinlan, 1989; Little and Rubin, 1987; Breiman et al., 1984).

JT (Joint distribution) *Bias:* A class of domain distributions $\{\mathrm{P}_{XC}\}$, usually defined by some Bayes/Markov net architecture, or mixture model (which implicitly defines a class of MDCs $\mathcal{D}$).
Learning rule: Choose the maximum likelihood domain distribution that accounts for the training examples. In practice this optimization problem can be quite difficult and most researchers resort to heuristic optimization techniques such as simulated annealing, the EM algorithm, or gradient descent procedures (Ghahramani and Jordan, 1994; Neal, 1992; Pearl, 1988; Little and Rubin, 1987).

DR (Default rules) *Bias:* A class of MDCs $\mathcal{D}$.
Learning rule: Choose a minimum error $d \in \mathcal{D}$ (Quinlan, 1989; Breiman et al., 1984).

CL (Classifier lattice) *Bias:* A class of CDCs $\mathcal{C}_s$ for each subset s of observed attributes (which implicitly defines a class of MDCs $\mathcal{D}$).
Learning rule: For each subset s of visible attributes, choose a minimum error $c_s \in \mathcal{C}_s$.

In order to analyze the efficacy of these learning strategies, and assess the difficulty of various learning problems, we need a mathematical model of the learning situation.

3.1 Formal model

Following Valiant, we assume there is a "natural" source of random test examples against which we can evaluate the accuracy of any MDC. In particular, we assume there is a distribution P_{XC} over the space of domain objects and concept labels $X_n \times \{0, 1\}$, called the *domain distribution*, from which random labeled objects are drawn independently. Before presentation to the MDC these labeled objects $\langle \bar{x}, c \rangle$ are first passed through the blocking process β

to yield test examples $\langle \bar{x}^*, c \rangle$. This induces a natural distribution P_{X*C} over the space of possible examples, called the *example distribution*; see Figure 4. The *accuracy* of a MDC d, written $\mathrm{P}_{X*C}(d)$, is just the probability that this d correctly classifies a random test example. Note that in general a classifier's accuracy depends both on the domain distribution *and* the blocking process.

To formalize the task of *learning* an accurate MDC from random training examples, as in (Valiant, 1984), we assume training examples are randomly drawn from the example distribution P_{X*C} along with their correct classifications, from which the learner L must produce a MDC that will be tested on examples drawn from the same example distribution. The learner's goal is then to produce an MDC with maximal accuracy. It turns out that the robustness of the various learning strategies, and the difficulty of specific learning problems, depend not only on the form of available training examples (χ), but also on our assumptions about the type of blocking process (β) operating on the domain.

Blocking processes: There are a number of reasonable assumptions one could make about the blocking process β, but we restrict our attention to just two: *independent blocking* (β_I), where each object attribute x_i is hidden with some fixed probability p_i, independently of x_i's value and the values of other attributes x_j, $j \neq i$; and *arbitrary blocking* (β_A), where object attributes x_i are hidden according to an arbitrary probability distribution that can condition on the complete object description $\bar{x}$ and its classification c. Independent blocking is a simple and convenient model that captures the intuitive notion that a missing value provides no information about an attribute's true value, nor any other attribute's value, nor the object's class. However, this is not an adequate model of every real world situation; for example, it cannot deal with circumstances where our knowledge of an attribute is *correlated* with its value. For example ex-inmates are less likely to answer questions of the form "have you ever been in prison?". Also, in medical databases missing values may actually provide clues about the subsequent classification (Porter, Bareiss and Holte, 1990; Rao, Greiner and Hancock, 1994). Here, this arbitrary blocking model is able to capture correlations between hidden attributes and their values, other attributes, or even concept membership.

We define a *learning context* (β, χ) by the type of blocking process β, and type of training examples χ, and consider various contexts below.

4 Robust learning

We first investigate the robustness of the various learning techniques introduced in Section 3 under the various learning contexts.

Definition 1 (Consistency) *A learning technique is* consistent *if, for any allowable bias $\mathcal{D}$, its learning rule is guaranteed to converge to the optimal $d \in \mathcal{D}$ for large training samples (a.s. in the limit).*

In particular, we consider the robustness of the previous learning strategies with respect to the following rather strong form of failure.

Definition 2 (Failure) *We say that a learning technique* fails in context (β, χ) *if there is an allowable bias $\mathcal{D}$ and example distribution P_{X*C} for which the technique converges (a.s.)*

to a $d \in \mathcal{D}$ with error rate arbitrarily close to $1/2$, when there exists a $d_{opt} \in \mathcal{D}$ with error rate 0.

Notice that for learning CDCs, consistency can be verified for any learning technique that chooses minimum error hypotheses, provided the bias $\mathcal{C}$ is not too expressive.[3] However, the situation is not so straightforward for learning MDCs; here, consistency of the various learning strategies is highly dependent upon the specific learning *context* (β, χ).

If we assume independent blocking (β_I) and train on the natural source of *incomplete* examples (χ_I), then all four techniques are guaranteed to converge to the optimal classifier (with caveats for IM and JT).

Proposition 2 *In context (β_I, χ_I):* CL *and* DR *are consistent;* IM *and* JT *are consistent if the domain distribution P_{XC} satisfies their implicit assumptions, but fail otherwise.*[4]

Of course, we might wish to learn from *complete* training examples in hopes of improving our overall learning efficiency. It turns out that in the case of independent blocking this intuition proves true for all but the DR learning strategy.

Proposition 3 *In context (β_I, χ_C):* CL *is consistent; (deterministic)* DR *fails;* IM *and* JT *are consistent if the domain distribution P_{XC} satisfies their implicit assumptions, but fail otherwise.*

However, even though complete training examples make learning easier under independent blocking for most strategies, they make consistent learning *impossible* under arbitrary blocking for *any* learning strategy.

Proposition 4 *In context (β_A, χ_C): all learning rules fail for any non-trivial bias $\mathcal{D}$.*[5]

This result makes intuitive sense, since complete examples supply *no* information about the blocking process that will be applied to future test examples. While this is not a problem under β_I where the optimal classifications are determined strictly by the instance distribution P_{XC}, this issue is fatal under β_A. Finally, however, if we consider training on the natural source of incomplete examples it is possible to obtain consistent learning once again, but now only via the DR strategy.

Proposition 5 *In context (β_A, χ_I):* DR *is consistent;* CL, IM, *and* JT *fail.*

Figure 5 summarizes the results.

---

[3] It is sufficient that $\mathcal{C}$ have finite Vapnik-Chervonenkis dimension (Vapnik and Chervonenkis, 1971) (and satisfy certain (benign) measure theoretic properties that we will not concern ourselves with here); see the next section. We place the same restriction on allowable MDC biases $\mathcal{D}$.

[4] That is, the IM and JT learning strategies each incorporate assumptions about the underlying domain distribution P_{XC}, and so each strategy may fail under (β_I, χ_I) if the domain distribution does not belong to the presumed class.

[5] A bias $\mathcal{D}$ is non-trivial iff it contains at least 2 distinct elements that are not complements.

	χ_C	χ_I
β_I	IM*,JT*,CL	IM*,JT*,CL,DR
β_A	{}	DR

Figure 5: Guaranteed convergence to optimal MDC

5 Efficient learning

We now consider the efficiency of learning MDCs. Following the methodology pioneered by Valiant, we consider how achievable learning performance is determined by the prior bias $\mathcal{D}$. Here we *quantify* the strength of bias by its measurable effects on the quality of learning that can be guaranteed: the difficulty of learning a set of MDCs $\mathcal{D}$ is measured by the number of training examples needed to reliably guarantee a near optimal hypothesis, in the worst case over all allowable example distributions permitted in the learning context.

Definition 3 (PACO-learning) *(Probably Approximately Class Optimal) For $\epsilon, \delta > 0$, a learner L* PACO-learns *a class of* MDC*s $\mathcal{D}$ given $m_L(\epsilon, \delta, \mathcal{D})$ training examples, if, for all example distributions P_{X*C} L outputs a* MDC *$d \in \mathcal{D}$ whose accuracy is within ϵ of the best $d_{opt} \in \mathcal{D}$, with probability at least $1 - \delta$.*

Notice our goal differs slightly from standard PAC-learning, in that we are forced to seek near-*optimal* rather than near-*perfect* classifiers, since with blocking *no* classifier can attain perfect accuracy in general. Notice also that we are only addressing the *sample* complexity of learning here, *not* computational complexity.

Definition 4 (Sample complexity) *For fixed ϵ and δ, let $m(\mathcal{D})$ denote the minimum sample size needed by any learner to* PACO*-learn $\mathcal{D}$; referred to as the* sample complexity *of learning $\mathcal{D}$.*

Intuitively, we expect the difficulty of learning a set of MDCs $\mathcal{D}$ to depend on the "complexity" of $\mathcal{D}$, *i.e.*, the more complex $\mathcal{D}$ is, the harder it is to learn. But what precise complexity measure actually determines the sample complexity of learning a class of MDCs $\mathcal{D}$? In the standard PAC-learning model, for fixed ϵ and δ the sample complexity of of learning a class of CDCs $\mathcal{C}$ is a linear function of $\mathrm{VCdim}(\mathcal{C})$; *i.e.*, $m(\mathcal{C}) = \Theta(\mathrm{VCdim}(\mathcal{C}))$ (Blumer et al., 1989; Ehrenfeucht et al., 1988). For learning MDCs however, we find that the appropriate complexity measure actually depends on the specific learning *context* (β_i, χ_j). Here we let $m_{i,j}(\mathcal{D})$ denote the sample complexity of learning $\mathcal{D}$ in context (β_i, χ_j).

First consider learning a class $\mathcal{D}$ under the *arbitrary blocking* model. Proposition 6 states that it is possible to consistently learn from *incomplete* training examples using the DR learning technique. Moreover, no strategy can do significantly better:

Proposition 6 *Under (β_A, χ_I): $m_{AI}(\mathcal{D}) = \Theta(\mathrm{VCdim}(\mathcal{D}))$. Furthermore,* DR *learning achieves this upper bound.*

	χ_C	χ_I
β_I	$m_{IC}(\mathcal{D}) = \Theta(\max_s \mathrm{VCdim}(\mathcal{D}_s))$	$m_{II}(\mathcal{D}) = \Theta(?)$
β_A	(impossible)	$m_{AI}(\mathcal{D}) = \Theta(\mathrm{VCdim}(\mathcal{D}))$

Figure 6: Sample complexities of PACO-learning $\mathcal{D}$

Proposition 7 shows that consistent learning is impossible under β_A if we train on artificially completed examples; this is sufficient to prevent PACO-learning in general.

Proposition 7 *Under* (β_A, χ_C)*:* no *non-trivial class* $\mathcal{D}$ *of* MDC*s is* PACO*-learnable.*

Learning under the *independent blocking* model (β_I) is more interesting. First, if we consider learning from *complete* training examples we immediately run into the problem that the classification accuracy of any MDC d cannot be estimated directly. This would be possible if we happened to know the attribute blocking rates a priori, but it is impossible to estimate these quantities from complete training examples. So at first blush, the prospect of learning a near-(class)optimal MDC under these circumstances appears quite bleak. However, the CL learning strategy is effective in this case.

Proposition 8 *Under* (β_I, χ_C)*:* $m_{IC}(\mathcal{D}) = \Theta(\max_s \mathrm{VCdim}(\mathcal{D}_s))$ *where* $\mathcal{D}_s$ *is the set of* CDC*s induced by* $\mathcal{D}$ *on attribute subset* s*. Furthermore,* CL *learning achieves this upper bound.*

Note CL does not estimate the accuracy of any candidate MDC, but instead builds a MDC by combining empirically optimal CDCs on each attribute subset. Surprisingly, the measure of class complexity is different in this case; so the inherent complexity of a given class of MDCs $\mathcal{D}$ depends on the learning context! In fact, there are parameterized class of MDCs $\mathcal{D}_n$ for which $m_{IC}(\mathcal{D}_n) = \mathrm{poly}(n)$ and yet $m_{AI}(\mathcal{D}_n) = \exp(n)$. This means that learning under (β_I,χ_I) is in some sense fundamentally easier than learning under (β_A,χ_I), as it can require exponentially fewer training examples in some cases.

Considering the problem of learning from *incomplete* training examples under independent blocking, we find that it is actually quite difficult to determine the precise measure of class complexity that determines the sample complexity of learning a class $\mathcal{D}$. First, comparing the relative difficulty of learning under the various conditions shows $m_{IC} \leq m_{II} \leq m_{AI}$, giving the intuitive result that learning from complete training examples is easier than learning from incomplete examples under β_I blocking. However, it is an open question as to whether complete training examples can ever provide an *exponential* advantage over incomplete examples in learning efficiency under β_I.

Open Question 1 *Under* (β_I, χ_I)*:* $m_{II} = \Theta(?)$.

Figure 6 summarizes the results.

6 Conclusion

Research directions: Of course, much work remains to be done. Aside from pursuing open technical questions, we are beginning to examine many extensions to better cope with real-world learning problems. For example, missing attribute values in medical databases typically provide *useful* information — namely that the missing attributes are *irrelevant* to the classification given the known attributes (Porter, Bareiss and Holte, 1990) — which could be exploited by a learning system (Rao, Greiner and Hancock, 1994). Notice that β_I is overly restrictive and β_A is too underconstrained to adequately model this situation. We are currently investigating alternative blocking models that (we hope) lead to better empirical learning performance in such domains.

Other interesting research directions involve alternative generalizations of standard classification learning: This work considers MDCs that classify every description, no matter how incomplete. Alternatively, we could consider *partial* classifiers that sometimes return "I don't know" (Rivest and Sloan, 1988). This would prove useful in domains where the consequences of an incorrect classification sometimes outweigh those of remaining silent. Another interesting generalization is to consider *active* classifiers which selectively observe only a few attributes before producing classifications — here, we would need to learn *active diagnosis strategies*. This raises the issue of how best to trade off the number of tests required against the accuracy of the classifier.

Contributions: This work constitutes a start on the general task of acquiring default knowledge from empirical observations. We specified the task of classifying incomplete examples, and noted that missing-data classifiers are more complex than complete data classifiers. We next formally defined the task of learning accurate missing-data classifiers from random examples and explicated the various assumptions making up a learning context in this case. Finally, we investigated the robustness of existing learning procedures, and studied the sample complexity of learning classes of missing-data classifiers; observing in each case that the results depend strongly on the specific context of learning.

This work provides a simple, unifying framework in which to study the problem of learning missing-data classifiers. By making various modeling assumptions explicit, we provide theoretical insights that can help explain empirical learning phenomena and provide guidance for applying learning techniques. For example, the preceding theoretical results clearly demonstrate that the relative effectiveness of particular learning strategies strongly depends on the nature of the *blocking* process involved. This observation has practical import: If blocking is known to be independent, then complete training examples will provide an advantage over incomplete examples, and the CL and JT learning strategies should be favored over DR. On the other hand, if blocking independence cannot be assumed, then it is extremely dangerous to use complete training examples and the DR learning strategy should be favored over CL and JT.

References

Angluin, D. and Laird, P. (1988). Learning from noisy examples. *Machine Learning*, 2(4):343–370.

Blumer, A., Ehrenfeucht, A., Haussler, D., and Warmuth, M. K. (1989). Learnability and the Vapnik-Chervonenkis dimension. *JACM*, 36(4):929–965.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984). *Classification and Regression Trees*. Wadsworth, Belmont, CA.

Clancey, W. (1985). Heuristic classification. *Artificial Intelligence*, 27:289–350.

Ehrenfeucht, A., Haussler, D., Kearns, M., and Valiant, L. (1988). A general lower bound on the number of examples needed for learning. In *Proceedings COLT-88*.

Ghahramani, Z. and Jordan, M. I. (1994). Supervised learning from real and discrete incomplete data. In *Proceedings CLNL-93*. (This volume).

Ginsberg, M., editor (1987). *Readings in Nonmonotonic Reasoning*. Morgan Kaufmann, Los Altos, CA.

Haussler, D. (1992). Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Information and Computation*, 100:78–150.

Kearns, M. J. and Li, M. (1988). Learning in the presence of malicious errors. In *Proceedings STOC-88*.

Kearns, M. J. and Schapire, R. E. (1990). Efficient distribution-free learning of probabilistic concepts. In *Proceedings FOCS-90*.

Kyburg, H. (1991). Evidential probability. In *Proceedings IJCAI-91*.

Little, J. A. and Rubin, D. B. (1987). *Statistical Analysis with Missing Data*. Wiley, New York.

Mitchell, T. M. (1980). The need for biases in learning generalizations. Technical Report CBM-TR-117, Rutgers University.

Neal, R. M. (1992). Connectionist learning of belief networks. *Artificial Intelligence*, 56:71–113.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, San Mateo, CA.

Porter, B. W., Bareiss, R., and Holte, R. C. (1990). Concept learning and heuristic classification in weak-theory domains. *Artificial Intelligence*, 45(1-2):229–263.

Quinlan, J. R. (1989). Unknown attribute values in induction. In *Proceedings ML-89*.

Rao, R. B., Greiner, R., and Hancock, T. (1994). Exploiting the absence of irrelevant information. In *AAAI Fall Symposium on 'Relevance'*, New Orleans.

Reiter, R. (1987). Nonmonotonic reasoning. *Annual Review of Computer Science*, 2:147–186.

Rivest, R. and Sloan, R. (1988). Learning complicated concepts reliably and usefully. In *Proceedings AAAI-88.*

Roth, D. (1993). On the hardness of approximate reasoning. In *Proceedings IJCAI-93.*

Schuurmans, D. and Greiner, R. (1994). Learning default concepts. In *Proceedings CSCSI-94.*

Shackelford, G. and Volper, D. (1988). Learning k-DNF with noise in the attributes. In *Proceedings COLT-88.*

Valiant, L. G. (1984). A theory of the learnable. *CACM*, 27(11):1134–1142.

Vapnik, V. N. and Chervonenkis, A. (1971). On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–280.