Computational Learning Theory: A Bayesian Perspective

Tom Chávez

Department of Engineering-Economic Systems, Stanford University / Rockwell International Science Lab, 444 High St., Palo Alto, CA 94301

Abstract: Computational learning theory (COLT), the field of research stemming from Valiant' seminal 1986 paper [Valiant], differs from previous attempts at formalizing inference in that it allows uncertainty and requires efficiency: an algorithm counts as a learning algorithm only if it errs with probability parameters we stipulate in advance, and only if it can be arrived at in polynomial time. Yet there are many aspects of learning that the model does not accommodate. It does not allow a learning agent to bring and assimilate prior beliefs to a problem. While it allows uncertainty in the form of its error (ε) and confidence (δ) parameters, it does not allow more specific "hedging" about the elements of a hypothesized concept. It does not permit us to support positively a hypothesis on the basis of sampled data, and it does not allow us to learn facts such as whether or not a coin is fair. Most distressingly, it relies on a classical statistical approach that is incompatible with the expressive requirements of real-world inference and the more convincing, well-justified notions of subjective probability.

This paper introduces a reformulation of the model which solves the preceding difficulties. We have been investigating a model of learning similar to COLT in its demands for efficiency and distribution-independence, yet one that also possesses the attractive features of Bayesianism, such as prior beliefs, continuous, incremental updating, and subjective probability. Our approach covers the classes of learnable concepts covered by COLT, as well as other categories of facts and concepts that it does not currently support. The new methodology possesses other distinct advantages: mistaken prior beliefs wash out quickly; the exactitude of inference over individual attributes can be specified in advance; and the efficiency of learning does not depend on the size of the concept space.

1.0 Introduction

Learning is such an intriguing topic because it draws on and concerns itself with some of the most interesting elements of psychology, computer science and philosophy. It has to do with psychology because we would like to know how people actually learn things; we would like to understand better how we form, support, extend and refute concepts and ideas. It touches on many ideas from philosophy, but most directly on topics from the philosophy of science -- how scientific theory emerges from large, complex and apparently self-contradictory bodies of data. Finally, it has much to do with computer science and artificial intelligence because we would like to build agents and programs that can go out in the world and infer new knowledge in some robust, efficient way.

As good as Valiant's model has proven at capturing and formalizing the most distinctive and significant aspects of the learning process, we believe that it possesses certain

shortcomings. We first present an overview of COLT, and we highlight its useful features and insights. We proceed to identify certain shortcomings with it and describe why they should concern anybody who studies inference. Many of those perceived difficulties have to do with the age-old debate between Bayesians and non-Bayesians in philosophy and mathematical statistics. We will try to focus that debate as much as possible into the current context, though some of what we argue requires -- or at least references -- arguments presented elsewhere and in other contexts. We go on to present an alternate model of inference -- or perhaps simply a more refined, Bayesian version of the COLT model -- based on the method of updating Bernoulli parameters using second-order conjugate distributions. Along the way, we observe how the approach presented here solves the deficiencies observed in the COLT model, and we proceed to identify its other strengths as well.

2.0 COLT: An Overview

2.1 An Example

We will first motivate things with a simple example, following [Kearns]. Suppose we are interested in learning about members of the animal kingdom, and to this end we create a collection of Boolean variables describing various features of animals. We assume that our variables, such as **has_mane, walks_on_four_legs, does_fly, hatches_eggs**, completely cover all the aspects we are interested in, and, of course, that the number of variables is finite. Now we might like to use this encoding, along with some sort of inference scheme, to learn what a particular animal is. More exactly, we want some framework for seeing examples of lions, say, and then constructing a concept that allows us to distinguish lions from non-lions on the basis of those observations.

2.2 Representations

We let X denote a set called a *domain*, also sometimes referred to as an instance space. We can think of X as the set of all encodings of objects, events, or phenomena of interest to us. For example, an element of X might be an object in a room possessing particular values for different properties such as color, height and shape. Or, in the case of our preceding example, an element of X is a particular animal, such as a cheetah. A learning algorithm seeks to build a concept for some element, or most typically a subset of elements, in X. In COLT, because concepts are usually considered clusterings of properties, we typically use concept representations that are expressible as simple rules over domain instances. Boolean formulae are obvious candidates for representing COLT concepts because they can capture such rule-based descriptions easily and elegantly.

We define a *representation* over X to be a pair (σ, C), where $C \subseteq \{1,0\}^*$ and σ is a mapping $\sigma: (C \rightarrow 2^X)$. For some fixed $c \in C$, we call $\sigma(c)$ the *concept* over X, and $\sigma(C)$ the *concept class*. We will also use $pos(c) = \sigma(c)$, the image of σ, or the set of *positive examples* of the concept. We assume that domain points $x \in X$ and concept representations $c \in C$ are efficiently encoded using any of the standard techniques (see, for example, [Garey]). Since X and σ are usually clear from context, we will often simply refer to the representation class C. [1]

A *labeled example* from a domain X is a pair $<x,b>$, where $x \in X$ and $b \in \{0,1\}$. In this paper, we will be considering only positive examples, i.e., elements of X for which $b = 1$. We shall refer to a *sample* as some finite set of positive examples over X.

In most of COLT, we find results relative to specific, parameterized classes of representations. Typically we have a stratified domain $X = \bigcup_{n \geq 1} X_n$ and $C = \bigcup_{n \geq 1} C_n$. The parameter n can be regarded as an appropriate measure of complexity. Usually n is the number of attributes, or variables, in the domain X. As an example, X_n might be the set $\{0,1\}^n$ and C_n the set M_n of all monomials of length $\leq n$, i.e., the set of all conjunctions of literals over the Boolean variables $x_1,\ldots,x_n$.

Returning to our example, we see that, given our variable definitions corresponding to animal attributes, we can represent a lion as some fixed combination of values over those variables. If all of our variables are Boolean, then each variable corresponds to the possession or non-possession of some attribute. Suppose that we have defined n variables such that each possible combination (out of the

1. The reader should not be disturbed by the sudden σ formalism. We are simply making explicit the notion that, in order to represent a concept in terms of a clustering of elements in the domain X, we must define a representation class C which identifies those elements for a fixed concept c.

2^n possible combinations) effectively describes some member of the animal kingdom. For example, **is_bird** $\wedge$ $\overline{\textbf{does\_fly}}$ $\wedge$ **weighs_over_200_lbs** $\wedge$ **runs_fast** might be a concept representation for an ostrich, where the concept class is taken to be M_n as defined above. Provided that we have a clearly defined ordering on the variables, it is a simple matter to construct a labeled sample consisting of lions and non-lions.

2.3 Distributions

Now that we have a way of representing concepts and phenomena, as well as definitions for examples and samples, we need some precise method for presenting those examples to a learning algorithm. Suppose we have a fixed representation for the concept of a lion given by **is_mammal** $\wedge$ **is_large** $\wedge$ **has_four_legs** $\wedge$ **has_claws.** In computational learning theory, a learning algorithm is given examples of a single concept $c \in C$ according to some fixed but unknown distribution D.

The idea of learning with respect to a fixed but unknown, arbitrary distribution on examples (often referred to as *distribution-free learning*) is one of the great strengths of the COLT model. We can think of the target distribution as a means of capturing the structure of the real world -- not necessarily capturing that structure *well*, but at least as representing some slice, however inaccurate or biased, of the phenomena we are trying to learn about. Because we seek a framework for describing learning in unanticipated scenarios, we will rarely have access to the distributions on data or examples prior to constructing the learning algorithms. Therefore we seek algorithms that perform well under *any* target distribution.

Of course, if the algorithm sees examples according to a skewed distribution, it will infer concepts reflecting only the data it has seen, and not necessarily the objective structure of the learning domain. Thus we are forced to characterize the error of the learning algorithm in terms of the target distribution. Suppose that a learning algorithm outputs a hypothesis h from a representation class H after seeing examples from the instance space drawn according to D. (Recent work in computational learning theory distinguishes between H and C because it is possible for some learning algorithms to create hypotheses in one representation class that serve as good approximations to target concepts in another.) COLT defines the error ε as

$$\varepsilon = \sum_{x \in X \mid h(x) \neq c(x)} D(x)$$

That is, ε is the probability weight according to D of all the mistakes h can possibly make. Measuring ε in this manner implies that "fluke" examples occurring with very low probability in the sampling process contribute only a negligible amount of error to the hypothesis. COLT requires that a learning algorithm output a hypothesis with error no larger than ε regardless of the distribution D, where ε is specified in advance. Returning to our example, if we specify a value of $\varepsilon = 0.1$, then with probability at least 90% the output hypothesis will correctly classify a lion as a lion and a non-lion as a non-lion. Conversely, the probability that the hypothesis mistakes a cheetah for a lion, say, is less than 10%.

2.4 Learnability

In [Kearns], Kearns makes a point not usually addressed in much of the COLT literature: because we are interested in computational efficiency, we must insure that the hypotheses output by a learning algorithm be evaluated efficiently, not just that they be arrived at efficiently. It is of little use to us if a learning algorithm outputs a system of differential equations that can only be evaluated in an exponential number of time steps. Thus we require a preliminary definition: if C is a representation class over X, then we say that C is *polynomially evaluatable* if there is a polynomial algorithm A that on input a representation $c \in C$ and a domain point $x \in X$ outputs $c(x)$.

We are now ready to present the central definition of learnability according to COLT. Let C and H be representation classes over X. Then C is *learnable from examples by H* if there exists a probabilistic algorithm A such that for some $c \in C$, if A has access to examples from $pos(c)$ drawn according to any fixed but unknown distribution D, then for any inputs ε and δ such that $0<\varepsilon,\delta<1$, A outputs a hypothesis $h \in H$ that with probability $1-\delta$ has error less than ε. Typically we include a requirement of efficiency as well: C is *polynomially learnable by examples from H* if C and H are polynomially evaluatable and A runs in time

polynomial in $1/\varepsilon$, $1/\delta$, and $|c|$. Often we will drop the phrase "by examples from H" and simply say that C is learnable, or polynomially learnable.

Suppose we are attempting to learn a monomial representation **is_mammal** $\wedge$ **is_large** $\wedge$ **has_four_legs** $\wedge$ **has_claws** for the concept of a lion. A COLT learning algorithm is given a distribution on lions, where each lion is represented as a truth assignment over the variables we have defined. The algorithm is then fed positive examples of lions as it attempts to construct a monomial sufficiently close to the target monomial so as to meet error specifications given by ε and δ. We can employ the learning algorithm given by Valiant in his ground-breaking paper [Valiant]. Let $x_1,\ldots,x_n$ denote literals (what we have been giving names such as **has_four_legs**) and let *POS* denote an oracle which returns a positive example of a lion drawn according to a distribution D.

$h=x_1\bar{x}_1x_2\bar{x}_2\ldots x_n\bar{x}_n$;

for i:= 1 **to** m **do** {m satisfies ε and δ constraints}

begin

$\hat{v}$=POS

for j:= 1 **to** n **do**

if v_j=0 **then** delete x_j from h

else delete $\bar{x}_j$ from h

end

output h

As the learning algorithm sees positive examples of lions, variables such as **can_speak** and **breathes_underwater** will be quickly deleted from h, since it is safe to assume that no lions can speak or breathe underwater. A variable such as **has_mane**, meanwhile, will be eliminated from h with probability roughly 1/2, presuming that the sampling distribution D accurately reflects the fact that about half of all lions are males, and only male lions have manes. Variables such as **has_claws** and **is_mammal** are true for all positive examples of lions, and so should withstand the pruning process of the learning algorithm and be included in the output hypothesis.

The learning algorithm can err in two ways corresponding to the two error parameters ε and δ. It can make an ε-type error in the following manner: suppose that a rare breed of Tibetan lion has no claws; yet because those lions are so rare, not one of them appeared in the training set of examples encountered by the algorithm. Thus the literal **has_claws** should have been deleted from the monomial. The problem is not serious, since the probability of misclassifying a Tibetan lion as a non-lion is bounded above by ε, and ε is at least as great as the probability weight of all Tibetan lions with respect to the distribution D. A δ-type error can occur if a randomly drawn set of lions is highly unrepresentative of lions in general -- for example, if every lion in the training set just happens to live in the circus, then the learning algorithm might include **lives_in_the_circus** in its concept for lion.

Ideally, we would like to have learning algorithms that never make mistakes. Such a goal, however, is clearly impossible, given the richness and unpredictability of most learning domains. The next best thing, it would seem, is to have some grasp of the types of error learning algorithms make as well as the magnitude of those errors in probabilistic terms. COLT provides such a capability, and this has distinguished it favorably from previous attempts at formalizing inductive inference.

3.0 Where COLT falls short

3.1 Prior beliefs, prior knowledge

Essential to the process of learning is building upon what is already known. Neither we nor agents approach new phenomena as *tabula rasa* -- blank boards upon which we write the solution to a problem or engrave the representation of a concept. If the car is broken, we do not infer what is wrong with it just by looking under the hood. Rather we employ all of our background knowledge about how cars work and especially what causes them to fail. Similarly, in the realm of intelligent agency, we would like robots to use knowledge we have given them at design time, as well as everything they learn in the meanwhile, as they proceed to take action in the world.

COLT provides a good model for inductive inference for a fixed problem, but it does not provide machinery to describe how a particular problem might be approached; in particular, it does not give us a means for incorporating

possibly uncertain prior beliefs or knowledge about the focus of inference. Arguably, a concept can be stored, and then at some other time the learning algorithm can see more examples and thereby reduce its error and increase its confidence for the concept. In this sense we might say that the learner has incorporated "prior" knowledge of a concept C in order to infer "more" about C, but clearly we are employing poorly-defined notions of "prior knowledge" and "more."

What we want, rather, is the ability to draw on knowledge in a knowledge base in order to focus and perhaps even accelerate the process of inference. Rather than throwing static concepts into a knowledge base and drawing on them only to complete contrived chains of reasoning or to answer straightforward queries, we wish to use static concepts as dynamic contributors to what we are learning. Rather than using prior knowledge as solely a constraining mechanism -- a tool for checking the consistency of what we already know to what we are seeing -- we should also use it as a means of enriching and clarifying inference.

Computational learning theory makes what we might call a "freeze frame" assumption: what we infer is based on what we see for a particular learning instance. COLT has nothing to say about what came before or after, and while this has proven useful for stripping away complexity in an attempt to get at an essential model for learning, it is too drastic; what has come before (and perhaps what comes after) would seem to matter a great deal to what we are inferring. Returning to our previous example, suppose we are trying to learn a concept for a lion based on positive examples. A COLT algorithm begins learning as if it knows nothing about lions: **has_mane**, **walks_on_four_legs**, and **flies** are all unknown variables that may assume the values 1 or 0 (**True** or **False**). Yet it seems quite reasonable that the learner might already know that lions do not fly and that they are mammals, and it would be good to incorporate this knowledge somehow into the learning process.

Now one might argue that we can simply set the values of those variables in advance. The first problem with this response is that those variables, since they are included in the representation scheme for the concept space, already determine the complexity of inference. We must redo our proofs of the running times of the learning algorithm or recompute lower bounds to match the representation we are using. In our example, if we are using a M_n representation with n=20 and we already know that lions do not fly and they are mammals, then we are really learning M_n for n=18. Some work has been done studying some versions of this problem (e.g., [Littlestone]), but still we might like to account for it in the basic model.

The second, much more serious problem is that we might not really be able to commit with certainty to the fact that lions do not fly, or, more to the point, that all lions have claws (remember the Tibetan breed). All the lions that the learner might have seen before had claws, but it would clearly be erroneous to assume the quantified proposition that all lions have claws. In fact, the learner's previous experience about lions with claws points to the fact that it has only seen clawed lions. Yet rather than discard this bit of information, we would like to bring it to the learning instance somehow in a way that allows revision yet adds information. Thus we would like to have a means for incorporating prior beliefs of the form "Most lions have claws" or "Most lions are some shade of brown." The way to do this is with probability, and we will have more to say about this soon.

3.2 Hedging within a concept

One of COLT's strengths is that it gives explicit probability parameters describing the uncertainty of the inference procedure. Thus it is often referred to as "PAC learning," for probably approximately correct learning.

Yet suppose that as we are learning, we want to "hedge" on particular aspects of the concept we are building. Going back to our example, suppose that we see a lion with a mane, and then another with no mane. In COLT, since the literal **has_mane** is not satisfiable in light of this apparent indeterminacy, it and its negation are eliminated from the output hypothesis. Rather than throw out the baby with the bath water, however, we might like to express and localize our uncertainty about the variable **has_mane**. In fact, the variable is not indeterminate at all. The truth of the matter is that about half of all lions have manes and the other half do not; this seems like information worth including in our concept for a lion. There are many conceivable cases where outside knowledge can affect the scope and accuracy of inference, particularly with respect to individual variables. COLT does not allow hedging on particulars, and this is something a more robust model ought to allow.

3.3 Degrees of support

Suppose as a learning algorithm goes about observing lions, it sees that every lion in the sample so far walks on four legs, and not a single one flies. Most weigh more than 100 pounds, and most chase antelope, while about half of them have manes. A COLT algorithm will infer only what is consistent in the Boolean logical sense with the entire sample. Thus, even though most of them weigh over 100 pounds, it might have seen some baby lions that weigh less than 100 pounds, and thus the value of the **weighs_more_than_100_pounds** variable in the algorithm's output concept is **False**. After seeing female lions, it will make the more egregious error and conclude that **has_mane** =**False**. It will infer only something like **has_four_legs** $\wedge$ $\overline{\textbf{flies}}$ -- clearly much valuable information has been hidden or effectively lost.

What we would like to have are degrees of support over all the attributes of the learning domain, some way of capturing and expressing all the richness and uncertainty of the learning process. If we observe that all lions have four legs, we are not merely learning that this is consistent with the concept **has_four_legs** $\wedge$ $\overline{\textbf{flies}}$, we are also learning something about the extent to which belief in the variable **has_four_legs** is supported. Plainly, this sort of uncertain knowledge goes to the very heart of what it is to learn something. Uncertainty is not a bad thing, so long as we have a means of capturing it in some expressive, axiomatic way.

3.4 Learning other facts

There are many other facts in the world worthy of learning that COLT does not accommodate. Although it is based on statistics, it does not give us a framework for learning various sorts of statistical facts -- for example, whether or not a coin is fair after observing a certain number of tosses. It does not contextualize what it learns to the realm of action, yet action would seem to constitute the backdrop against which anything worth learning is actually learned (for more discussion of this point, see [Good]). More significantly, it is not at all obvious how one would go about extending COLT into this decision-theoretic domain, e.g., accounting for learning in the inevitable presence of constraints on learning resources, or learning in the presence of variable costs of information (when it costs valuable time or money to see another example, say) Theoretical strides on research on decision making under bounded resources cannot be smoothly integrated into traditional COLT techniques.

3.5 The right kind of probability

The most serious problem with COLT is that it relies on a faulty interpretation of probability that in turn fosters a false sense of comfort with the probability parameters ε and δ. We can highlight this difficulty by asking simply, "What is meant by the term 'probably approximately correct'?" Nowhere in the literature (so far as we know) has there been any systematic discussion of COLT's probabilistic approach, its consequences and assumptions.

There has been ongoing debate in mathematical statistics, engineering and philosophy surrounding proper interpretations of probability. We can nevertheless isolate a few points relevant to our current context. Because the probability parameters in COLT are specified in advance, and sample sizes are computed relative to those parameters following standard methods from classical (frequentist) statistics, it seems that COLT's intended, or at least default, probabilistic interpretation is roughly as follows: given a learning domain (i.e., concept space, a suitable representation and some distribution D on examples fixed in advance), in N identical, repeatable learning instances where a number of examples given by the bounds of COLT are observed according to D, a learning algorithm will output representations such that as N goes to ∞, the proportion of representations which have error greater than ε is less than δ. Now this is a very rough sketch, for we have left much unspecified. Do we fix the learning algorithm, or do we allow different learning algorithms for varying values of N, as long as they all perform learning for the same concept space? Do we fix the concept being learned as well? Is it allowable to fix the concept and vary the learning algorithms? We have also ignored the question of error, though it is safe to take it as the weight relative to D of examples misclassified by the learning algorithm. But then we might ask if it is sensible to vary the distributions from learning instance to learning instance.

All of these permutations call into question the very meaning of repeatability required by a frequentist interpretation. In what sense is each learning instance a repeatable trial identical to any other learning instance? Notice how far we are from the simplistic realm of coin flipping, where,

all things being equal, it appears relatively safe to assume that distinct flips of a coin are repeatable and identical.

The approach to assigning values for the probability parameters in COLT is indistinguishable from the way in which people assign thresholds for confidence intervals. We say that we want δ=.1 -- but what does this mean, really, and what makes it different from δ=.05? What constitutes "extreme" error or confidence? (See [Good], p. 49.) The confidence interval is, as most honest statisticians admit, a confidence trick, a way of making confidence pronouncements that are true 95% or 99% of cases in the long run. Yet we have already seen how difficult it is to fix any reasonable notion of a "long run" for learning.

Even if we could fix some sensible notion of the "long run" for learning, confidence techniques from classical statistics still do not give us what we want. As Keynes succinctly put it, "In the long run we will all be dead." As anybody familiar with these kinds of statistical sampling techniques knows, the samples required to accommodate the pre-specified probability parameters are often absurdly large. Moreover, they flagrantly ignore information acquired during sampling. The criticism can be brought into focus by giving an example due to Good: if you observe 300 spins of a roulette wheel and there is no occurrence of a 7, should you regard the probability of a seven on the next spin as 0 or 1/37, its official value? A frequentist's most likely response is to tell you to spin it another 200 times.

Some might argue that this is irrelevant since COLT does not support that kind of inference, and though this reply is misguided, we can nevertheless accommodate the objection by means of another example. Suppose we are trying to build a concept for a lion, and our learning algorithm has seen r examples so far, where $r<<m$, the number of examples prescribed by COLT in order to achieve confidence $1-\delta$ and error ε. Yet the learning algorithm might have already observed that all lions are brown, walk on four legs, and have lots of teeth -- enough information to classify lions relative to the given distribution with error less than ε. Regardless of how clever this particular learning algorithm is, the number of samples required is fixed in advance, and, in many cases, much more than is needed or even possible. We take an example from [Kearns], *p.* 16, where we are interested in learning a monomial over n literals using positive examples only. Kearns gives the following exact bound for the number of examples required:

$$\frac{2n}{\varepsilon}(\log 2n + \log \frac{1}{\delta})$$

Suppose we are using 4 literals, and we set δ=1/16 and ε=1/10. Then the preceding equation implies that a COLT learning algorithm requires 560 examples; yet there are only 2^4, or 16, possible monomials over 4 literals, so clearly the number of examples needed by the algorithm is overstated (with just 16 examples, we have $\delta = \varepsilon = 0$). In short, the statistical techniques upon which COLT relies do not allow inference to proceed dynamically based on relevant knowledge acquired during the sampling process.

3.6 Bayesian learning as a solution

The solution is a more realistic, more honest acceptance of what we are really using all along: subjective, or personal, probability. Those who study COLT presumably have no difficulty using probability as a means of expressing uncertainty. If that is the case, then let us make ourselves clear about what we mean when we use it. We follow the conventional Bayesian view in defining a probability as a degree of belief in a proposition E given the background state of information I in the believer's mind. We shall not dwell on the distinct, yet consistent, perspectives on subjective probability (e.g., logical subjective probability versus personal subjective probability). Like probabilities interpreted as frequencies, we employ the axioms of probability to regulate and make consistent all our probability assessments, now considered as subjective measures of belief.

The approach is honest because, as has been argued forcefully elsewhere, it is most likely what frequentists and other empiricists are doing anyway; and, most attractively, it "covers" the other models and interpretations [de Finetti], so that one can continue adhering stubbornly to ones frequentist ways while still accepting everything we do here.

The immediate consequence of our approach is that rather than employing Boolean truth values (0 or 1, **True** or **False**) for the concepts and facts we will be inferring, we

employ a closed unit interval for each variable to describe belief in its truth or falsity. A high value means that we ascribe a high level of belief to the variable's being true, and a low value implies a low level of belief in its truth, or, conversely, a higher belief in its falsity. Such an attack will allow us to do all the things we would like to be able to do with COLT: hedging on particulars, degrees of support (where support, using our new terminology, is indistinguishable from belief), and prior beliefs.

4.0 Bayesian Learnability

The goal of a learning algorithm in the COLT literature is to infer some unknown subset of X, called a concept, after seeing a certain number of its instances. We accommodate such tasks, but we also allow some flexibility for learning propositions which need not be directly stated in strict logical terms relative to the domain. For example, learning whether or not a coin is fair is a proposition not directly expressible as a combination of "Tails" or "Heads," yet it is something our model allows as a valid subject of inference. Thus our approach subsumes COLT in that it allows learning beyond simple concepts expressible as logical rules over domain instances. For the sake of brevity, we shall not consistently make explicit these different types of learning, but it is useful to keep them in mind as we proceed.

We now present a definition central to this paper: a n-Bernoulli representation class is a representation class over n Bernoulli variables, i.e., variables assuming some value in the closed interval [0,1]. Moreover, we shall say that a p-Boolean representation class and a q-Bernoulli representation class are equivalent if there exists some isomorphism between the literals of the Boolean class and the variables of the Bernoulli class. Since such an isomorphism exists if the two classes use the same number of variables, we can see that two such classes are equivalent in general if $p=q$. (We will sometimes drop the parameter n and simply refer to Boolean and Bernoulli representation classes; we will always assume the existence of a fixed domain X.)

It is perhaps appropriate at this time to state the motivation for the preceding definition of equivalence. In Section 3, we observed some of COLT's shortcomings: in particular, its rigidity with respect to particular attributes of a concept being learned, and its inability to accommodate addition or subtraction of degrees of belief for elements of a concept based on observed data. The solution we posited was a Bayesian, subjective use of probability. The point of defining equivalence between representations is that the concept being learned or the fact being inferred can still retain all of its exactitude in the Boolean representation, only now we are being honest: no matter how many times we flip the coin, no matter how many lions we see with four legs, we will always allow room for some uncertainty.

Thus we will perform inference over a Bernoulli representation relative to the exact facts described in the Boolean one. The Boolean facts are still true or false -- we just never presume (or pretend) to know them with certainty. The most significant consequence of this approach is that we are always learning. Attributes of a concept are always undergoing positive or negative updating consistent with the laws of probability. The Bernoulli representation is a more epistemologically accurate frame within which to perform inference because it captures and expresses knowledge such as the fact that most birds fly, but penguins do not. Most lions have four legs, but we do not want to abandon that fact altogether if we encounter one that lost a leg in a bloody fight with a rhinoceros. One important advantage of this approach, then, is that it circumvents the traditional problems of monotonicity AI researchers have struggled with for so long, problems COLT never fully escapes.

Still, we have no formalism for performing the type of inference we seek. What we want is a framework that possesses all of the strengths of COLT -- in particular, demands for efficiency and allowances for uncertainty -- yet also accommodates desiderata such as hedging, ranges of uncertainty for particular attributes, and prior beliefs. The solution, and the main result of this paper, is what we call a Bayesian learning scheme, which we will also abbreviate by BLS.

Definition: Suppose that a Boolean representation class B is equivalent to a Bernoulli representation class C, and let b_i and c_i denote the ith variables of B and C, respectively. We shall denote the domain of inference by X. We shall call Φ a BLS for B relative to C iff the following all obtain:

1) Φ maintains a set of distributions β_i, for $i=1..n$, where n is the number of variables in B and C. Each β_i encapsu-

lates current belief about the value of c_i, and therefore is a second-order density function on the Bernoulli variable b_i.

2) Let c^o denote some fixed concept or fact in C. For some $x \in pos(c^o)$, let $x_i \in \{0,1\}$ denote the observation value of the ith attribute or variable of x. We denote the conjunction of all background information and knowledge by ζ, and brackets denote probabilities. After Φ observes x, it updates B according to Bayes' Theorem:

$$\{B_{after} | x,\zeta\} = \frac{\{x | B_{before},\zeta\}\ \{B_{before} | \zeta\}}{\{x | \zeta\}}$$

More exactly, we have $\forall i$

$$\{b_{i_{after}} | x_i,\zeta\} = \frac{\{x_i | b_{i_{before}},\zeta\}\ \{b_{i_{before}} | \zeta\}}{\{x_i | \zeta\}}$$

4) Given n-place vectors $\vec{\varepsilon}$ and $\vec{\delta}$, and a distribution D on examples in $pos(c^o)$, where $\forall i$ we have $\varepsilon_i<1$ and $0<\delta_i<1$, Φ will output an updated version of B, denoted B^*, that is probably approximately correct in nearly the same sense as Valiant: that is, Φ halts and outputs B^* such that $\forall i$, with probability greater than $1-\delta_i$ we have

$$b_i^* (1+\varepsilon_i)^{-1} \le c_i \le b_i^* (1+\varepsilon_i)$$

5) Let ε_{min} and δ_{min} be the smallest elements of $\vec{\varepsilon}$ and $\vec{\delta}$, respectively. Let μ_i denote the mean of the corresponding distribution β_i. Then Φ computes B^* in time polynomial in $1/\varepsilon_{min}$, $ln\ 1/\delta_{min}$, and $1/\mu_i$.

One of the shortcomings of this definition of a BLS is that its running time is determined solely by the smallest values of the probability parameters for a particular attribute. The time cost of using a BLS, then, is determined exclusively by the parameters of the most precise variable(s) of interest. In practice, sampling in order to satisfy the most precise variables' probability parameters will necessarily bring all the variables to matching levels. Future work will focus on accounting more completely for observation costs, not just in terms of sample size but also particular values of a given example's variables.

Finally, we shall say that a class $\mathcal{F}$ is Bayesian (polynomially) learnable iff

1) there exists some Bernoulli representation class B such that $\mathcal{F}$ and B are equivalent; and

2) there exists a BLS Φ for B relative to $\mathcal{F}$.

5.0 Preliminary Results and Future Work

We have just begin to study learning in this framework, though there a number of interesting results we have already derived for the model. Proofs and further development of the Bayesian learning framework are forthcoming in [Chavez].

Theorem 1: Let C be some Boolean representation class, and let B be some Bernoulli class equivalent to it. Then B converges to C in the following sense: if Φ is a BLS for B relative to C, then as N, the number of examples that Φ observes, goes to ∞, for all i we have $b_i \to c_i$.

A more significant result is Theorem 2, which essentially tells us that anything learnable in COLT is learnable in a BLS as well.

Theorem 2: Any Boolean representation class C is Bayesian polynomially learnable.

We prove theorem 2 by using Monte-Carlo sampling and recent approaches to Bayesian analyses of simulation algorithms [Dagum]. Such analysis also shows that the details of prior distributions often "wash out" quickly in the sampling process, thus addressing the standard complaint from frequentists worried about the adverse effects of assigning possibly arbitrary or mistaken prior distributions. Because we use Monte-Carlo sampling techniques, complexity of inference depends only on the sizes of the samples used, and not on the number of variables instantiated during learning. Thus, we have the following important result.

Corollary: The computational complexity of a BLS is independent of the complexity of the concept representation.

The preceding corollary appears to be a significant advancement over COLT methods, where the complexity of the representation used figures prominently in the running times of the algorithms. As work proceeds, we will pay special attention to the sources of cost and complexity in a BLS, and how they distinguish it, either positively or negatively, from COLT approaches.

Acknowledgments: I would like to thank Paul Dagum and Eric Horvitz for insightful comments and discussions on this work. Eric Horvitz read and helped clarify a draft of this paper. This research was supported by IR&D Funds of the Decision-Theoretic Systems Group of the Palo Alto Laboratory, Rockwell International Science Center.

References

[Chavez] Tom Chavez. "Bayesian Learning Schemes" Technical Report, Rockwell International Science Lab, Palo Alto, CA.

[Dagum] Paul Dagum and Eric Horvitz. "A Bayesian Analysis of Randomized Approximation Procedures," September, 1991. Knowledge Systems Laboratory, Stanford University, Stanford, CA. KSL-91-54.

[de Finetti] Bruno de Finetti. "Foresight: Its logical laws, in subjective sources," reprinted in H.E. Kyburg and H.E. Smokler (eds.), *Studies in Subjective Probability*. London: John Wiley, 1964, p. 93-158.

[Garey] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. 1979, W.H. Freeman and Company, New York.

[Good] I. J. Good. "The Bayesian Influence, or how to sweep subjectivism under the carpet," in Harper and Hooker (eds.), *Foundations of Probability Theory, Statistical Inference, and Statistical Theory of Science*. Dordrecht: Reidel.

[Kearns] Michael J. Kearns. *The Computational Complexity of machine Learning*. 1990, The MIT Press.

[Littlestone] N. Littlestone. "Learning quickly when irrelevant attributes abound: a new linear threshold algorithm," *Machine Learning*, 2(4), 1988, p. 245-318.

[Valiant] L.G. Valiant. "A Theory of the Learnable," *Communications of the ACM*, 27(11), 1984, p. 1134-1142.