ELSEVIER

Artificial Intelligence 85 (1996) 301–319

Artificial Intelligence

Exploratory analysis of speedup learning data using expectation maximization

Alberto Maria Segre[a,*], Geoffrey J. Gordon[b,1], Charles P. Elkan[c,2]

[a]*Department of Management Sciences, University of Iowa, Iowa City, IA 52242, USA*
[b]*School of Computer Science, Carnegie-Mellon University, Pittsburgh, PA 15213, USA*
[c]*Department of Computer Science and Engineering, University of California at San Diego, La Jolla, CA 15213, USA*

Received January 1995; revised September 1995

Abstract

Experimental evaluations of speedup learning methods have in the past used non-parametric hypothesis testing to determine whether or not learning is beneficial. We show here how to obtain deeper insight into the comparative performance of learning methods through a complementary parametric approach to data analysis. In this approach experimental data is used to estimate values for the parameters of a statistical model of the performance of a problem solver. To model problem solvers that use speedup learning methods, we propose a two-component linear model that captures how learned knowledge may accelerate the solution of some problems while leaving the solution of others relatively unchanged. We show how to apply expectation maximization (EM), a statistical technique, to fit this kind of multi-component model. EM allows us to fit the model in the presence of censored data, a methodological difficulty common to experiments involving speedup learning.

1. Introduction

Speedup learning methods, such as subgoal caching [17] or explanation-based learning [14], are generally intended to improve the performance of a resource-bounded problem-solving system. Performance improvement is usually defined to mean operating more quickly at a fixed level of competence. Unfortunately, determining the extent of any performance improvement—or, indeed, detecting whether there is any improvement at all—is difficult.

* Corresponding author. E-mail: segre@cs.uiowa.edu.
[1] E-mail: ggordon@cs.cmu.edu.
[2] E-mail: elkan@cs.ucsd.edu.

SSDI 0004-3702(95)00115-8

Since conclusive formal arguments about the performance improvement due to a speedup learning method are difficult to construct, experimental studies provide the only realistic means of detecting or quantifying performance improvement [7]. Data collected in these studies are typically analyzed using some form of hypothesis testing, where the null hypothesis is that there is no difference in performance with or without learning. In [15] we show how several common methodological choices can compromise the reliability of conclusions drawn from experimental studies of speedup learning. One of these methodological difficulties, the presence of *censored data*,[3] is subsequently addressed in [6], where non-parametric methods are used to test the hypothesis that learning improves system performance.

Hypothesis testing provides little insight into the *qualitative behavior* of a learning system. It simply provides an answer, with some degree of statistical certainty, to the question of whether or not learning improves performance on a sampled problem population. There are times where statistically significant differences can be uninteresting or even misleading from a practical standpoint:

> . . . even when a statistical result is obtained, it does not substitute for a careful intuitive examination of the data, checking that the test is not "hiding" important characteristics of the data [6].

This paper presents a rigorous approach to modeling system performance intended to expose this kind of "hiding".

The contributions of this paper are three-fold. First, we show how to augment traditional hypothesis testing with a complementary, more exploratory, approach. This approach to exploratory data analysis is parametric in nature: we show how, by positing a model and using a statistical technique to estimate parameter values for the model, a deeper understanding of system operation can be achieved. Exploratory analysis of the type advocated here is a quantitative, reproducible method of performing the kind of "intuitive examination" mentioned above.

The second contribution is a new mathematical model of speedup learning to support the type of exploratory analysis and parameter fitting just described. We have in previous work used a simple linear regression model of problem-solving performance to quantify the benefits of certain types of speedup learning [13,16, 17]. Here, we propose a more sophisticated two-component linear model that better captures the effects of speedup learning, in particular, how learned knowledge may affect some problems more than others.

The third contribution is a statistical technique to estimate the parameters of the two-component model in the presence of censored data. This technique is

[3] A censored measurement is one where we observe a bound on the measured value rather than the value itself. For example, if we wait three hours for the problem solver to solve a problem, then give up, we have a censored measurement: we know that the actual time to solution is more than three hours, but we do not know how much more. Resource bounds cannot be avoided when solving nontrivial search problems. Thus the censored data problem is fundamental and must be addressed in any credible empirical test of a problem-solving system.

based on *expectation maximization* (EM), a general method for maximum likelihood estimation in the presence of missing data, of which censored data are a special case. We show how to use EM to perform multiple-component linear regression in the presence of censored data.[4]

In the next section, we introduce a sample dataset reconstructed from the machine learning literature. This dataset serves as an example throughout the remainder of the paper. We review a non-parametric method for hypothesis testing in the presence of censored data, and we discuss how information still available in the dataset is not revealed by this test. In Section 3, we introduce EM and show how it can be applied to investigate the performance of speedup learning systems in the presence of censored data. In Section 4, we show how EM can fit a simple linear model from censored data, using non-learning system performance from the dataset of Section 2 as an example. We then introduce a two-component model of speedup learning and show how EM can be used to fit this new model (again in the presence of censored data), and illustrate the process using learning system performance from the sample dataset.

2. An illustrative example

The example used throughout this paper revisits the classic *logic theorist* (LT) experiment [11]. Reports of experiments in this domain have appeared several times in the speedup learning literature [10,12,14]. The primary question we explore here is whether or not an explanation-based learning component, when combined with a standard backward-chaining problem-solving system, provides a performance improvement in this domain.

The set of LT problems is taken from Chapter 2 of *Principia Mathematica* [19]. The 87 problems in the set correspond to the original 92 problems of Chapter 2, reformulated for use with definite-clause theorem provers [10] (a full printed version of the domain theory and problem set used in this paper can be found in [14]).

The backward-chaining problem solver used is a definite-clause theorem prover implemented in Common Lisp. This is the same theorem prover used in our previous work on subgoal caching [17] and explanation-based learning [14]. The theorem prover is configured to perform resource-bounded unit-increment depth-first iterative deepening. The data described here were collected on a 32MB 90MHz Pentium system running Gnu Common Lisp and the Linux operating system.

A resource limit of 5×10^4 node explorations was imposed on each problem attempted. In each trial CPU times and node exploration counts were recorded, along with an annotation indicating whether or not the problem was solved (i.e., whether the problem was "censored").

[4] After submitting the first version of this paper, we discovered that others have previously described a less efficient EM algorithm for this problem [8].

In the first trial, the theorem prover solved 34 of the 87 problems within the resource bound. In the second trial, 4 problems were selected randomly from among the 34 solved in the first trial and used to generate macro-operators with the EBL*DI algorithm [14]. The theorem prover was then tested on the remaining 83 problems. The learning system solved 36 of the remaining 83 problems tested within the resource bound.

2.1. Scatter plot inspection

The simplest method of analyzing the data collected in trials 1 and 2 is to make a scatter plot of elapsed CPU time for the learning system versus the non-learning system. Fig. 1 is such a plot, where a logarithmic transform has been applied to both axes for clarity. Each datapoint represents a single problem. CPU time to solution without learning is plotted on the horizontal axis, while CPU time to solution after learning is plotted on the vertical axis. Datapoints falling below (above) the $y = x$ line correspond to problems that are solved faster (slower) after learning.

An informal analysis of Fig. 1 seems to indicate that learning is indeed advantageous. The learning system solves 6 more problems than the non-learning system. In addition, of the 30 problems solved by both systems, 17 are solved faster after learning, while only 12 are solved more slowly (the time to solve one problem is unchanged). Unfortunately, the 47 doubly censored problems are difficult to factor into this kind of informal analysis. A comparison of summary

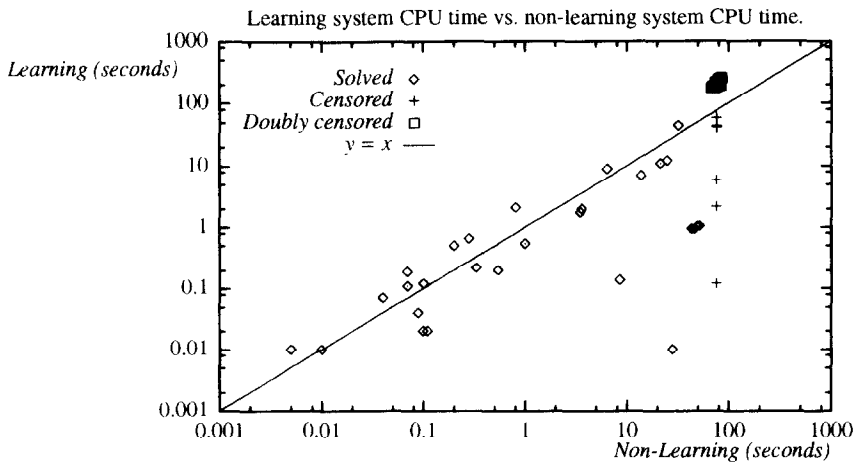

Fig. 1. Plot of learning system CPU time against non-learning system CPU time. The "diamond" datapoints shown correspond to the 30 problems solved both by the learning and non-learning systems, while the six "cross" datapoints correspond to those problems solved only by the learning system. The 47 "square" datapoints correspond to doubly censored problems, that is, problems where both the learning and non-learning system exhausted the 5×10^4 node exploration resource bound.

statistics (e.g., total CPU time used by each system on problems solved by both systems) is less subjective, but similarly confusing, and indeed potentially misleading [15].

2.2. Hypothesis testing

The analysis advocated in [6] relies on two non-parametric methods, the one-tailed paired sign test [2] and the one-tailed paired Wilcoxon signed-ranks test [20], suitably extended to account for censored data. These tests are non-parametric analogues to the more commonly used parametric Student t-test. They test for statistically significant differences between the solution times for the learning system and the solution times for the non-learning system.[5] Unfortunately, because of the maximally conservative way in which these tests handle censored data, they are not powerful when the dataset contains many censored observations: as Etzioni and Etzioni [6] point out, sufficiently many censored datapoints can cause these tests to accept the null hypothesis regardless of the strength of the evidence from the uncensored datapoints.

For example, when we apply these tests to the data of Fig. 1, in which more than half of the observations are censored, we can reach no useful conclusions.[6] If we test the null hypothesis that the learning system is faster, the censored data extension of the signed-ranks test strongly fails to reject it (if we reject for small values of p, then $(1-p) \ll 10^{-6}$). On the other hand, if we test the hypothesis that the non-learning system is faster, we fail even more strongly to reject it ($(1-p) \ll 10^{-11}$). These two results mean that, according to the extended signed-ranks test, at any reasonable significance level, the performance of the learning system is indistinguishable from that of the non-learning system. If the p values were less extreme, a few more datapoints might allow us to reach some conclusion, but with these p values, such prospects are dim. Thus, despite differences we can easily see, and despite differences revealed by the method described in Section 4, the extended signed-ranks test cannot detect a difference. The situation is similar, although less extreme, for the extended sign test.

---

[5] The t-test is inappropriate here since it requires the underlying distribution of the measured solution times for each problem solver to be normal. This assumption is unwarranted, since the censoring will cause datapoints to cluster around the resource limit.

[6] The form of censorship assumed in this paper is more general than the restrictive form used in [6] where every censored datapoint displays exactly the same resource consumption. The latter, more restrictive, variant arises naturally when a constant resource limit is imposed directly on the parameter of interest. In this case, all doubly censored points fall exactly on the $y = x$ line, and all singly censored points have both the true and the observed values of the censored coordinate larger than the value of the uncensored coordinate. Because of our more general setting, we extend the tests of [6] in the natural way: a censored observation is treated as if it lies either at its censoring point or at $+\infty$, whichever provides greater support for the null hypothesis. We also considered other ways of extending the tests, but each of these other extensions results in a test that can support a false conclusion.

3. Modeling problem-solving performance

Our approach to evaluating speedup learning performance combines the more informative nature of scatter plot inspection with a rigorous mathematical foundation. Briefly put, our approach is to first posit a model of system performance, and then to use a statistical technique called *expectation maximization* (EM) to estimate values for the parameters of this model. Using the EM technique allows us to estimate parameter values even though some performance data is censored. We can then examine the fitted model in order to identify trends that cannot be directed directly in the raw data (either because of its size or because of censoring), and that cannot be detected by hypothesis testing.

EM has been used to address problems studied in statistics and operations research under the names "life testing" and "reliability testing". The first name arises from medical studies in which the object is to estimate the average lifetime of a group of patients when some of the patients are still alive. The second name arises from quality control experiments in which the object is to estimate the mean time to failure for a sample of parts when not all the parts have failed yet. Recently, EM has also shown promise in unsupervised learning tasks such as the discovering patterns in DNA and protein sequences [3]. As might be expected given its heritage, EM, like the non-parametric tests of [6], can deal with censored data. But unlike these weaker methods, EM is a parametric technique. That is, it begins from a prespecified model with a prespecified finite number of parameters, and adjusts these parameters to fit the data.[7]

3.1. Using EM to model performance

Assume we are given a problem-solving system and wish to evaluate its performance with respect to some set standard. We gather data by presenting the system with n problems of calibrated "difficulty" (more on this later) and measuring its resource consumption (e.g., elapsed CPU time) on each problem. Since we generally cannot afford to let the system run to completion on every input (as it might take years or even centuries to finish), we sometimes cut the system off before it finishes, yielding censored data.

Our observations thus comprise three n-vectors, $\Delta = [\delta_1, \ldots, \delta_n]$, $X = [x_1, \ldots, x_n]$, and $Y = [y_1, \ldots, y_n]$, where δ_i measures the "difficulty" of the ith problem, x_i is the resource amount consumed on the ith problem, and y_i is 0 if the system actually solves the ith input and 1 if the system is cut off before solving the problem. Note that x_i is only a lower bound on resource consumption if $y = 1$.

[7] EM could also be used for parametric hypothesis testing, since the computations required to fit a model are similar to those required to test a hypothesis. However, violations of parametric assumptions (such as deviations from linearity or normality) can seriously affect the significance level of a hypothesis test, even while leaving the qualitative appearance of a fitted model unchanged. Thus the focus here is on the qualitative aspects of the analysis.

We assume that the observed vectors X and Y are obtained from a "true data vector" Z which we cannot directly observe. Each $z_i \in Z$ is the resource amount that the system would have consumed on the ith input if we had ignored the resource limit and let it run until it eventually halted. We also assume that the elements of (Δ, Z) are independent, identically distributed observations from some known density with parameters Θ (this is our parametric assumption). We are attempting to estimate the parameters of a known distribution, rather than trying to proceed without any information about (Δ, Z) whatsoever. In principle, we could assume an arbitrarily complicated distribution for (Δ, Z), but for this paper, we use the linear models described later. Our goal is to obtain a good estimate of Θ, since Θ characterizes the relationship between each δ_i and z_i.

Suppose that, instead of the censored observations X and the censoring flags Y, we could observe the true data Z. Then it would be relatively easy to estimate Θ using a technique such as maximum likelihood estimation [4]. In fact, we would not need the full true data Z; it would be enough to have *sufficient statistics* describing the data, where which statistics are sufficient depends on the distribution of the (δ_i, z_i). Let this vector of sufficient statistics be denoted $T_\Delta(Z)$.

If we knew $T_\Delta(Z)$, we could estimate Θ. On the other hand, if we knew Θ, we could approximate the sufficient statistics $T_\Delta(Z)$ with their expected values $E(T_\Delta(Z) \mid \Theta)$. This apparent dilemma is the basis of the EM algorithm.

We proceed as follows. We begin with an initial estimate $\hat{\Theta}_0$ of Θ. First, we use this estimate to compute $\hat{T}_0 = E(T_\Delta(Z) \mid \hat{\Theta}_0, X, Y)$, an initial estimate of the sufficient statistics $T_\Delta(Z)$ based on the observed data and the guess at the parameters. Next, we use $\hat{T}_0$ to update the estimate of Θ: we set $\hat{\Theta}_1$ to be the maximum likelihood estimate of Θ assuming that $\hat{T}_0$ are the true sufficient statistics. We repeat this process over and over, alternately improving the estimate of Θ or of $T_\Delta(Z)$. This process is called the *expectation maximization* algorithm, because it alternates between computing an expectation (the E step) and a maximum likelihood estimate (the M step).[8] The EM algorithm is described in detail in [5]. It is guaranteed, under certain general conditions, to converge to the maximum likelihood estimate of Θ based on the observables Δ, X and Y.

3.2. Measuring problem difficulty

Our experiment explores the relationship between two variables, Δ and the true resource consumption Z. Above, we informally explained δ_i as the "difficulty" of the ith problem. In this section, we describe in more detail what Δ is, why we need to know it, and how we measure it for the LT experiment.

Beneath the machinery of EM, our experiment is a regression analysis. The dependent variable is the true resource consumption Z and the independent variable is Δ. So, the basic requirement for Δ is that it be a good predictor of Z. In other words, a problem with low δ_i should consume fewer resources on average

[8] This is not the most general form of the EM algorithm, but it is sufficient for our purposes. This form works whenever the logarithm of the probability density function for the true data is linear in $T_\Delta(Z)$.

than a problem with a high δ_i. It is because of this requirement that we have informally called δ_i the "difficulty" of the ith problem.[9] A second requirement for Δ is that it should be easily and accurately measurable. In addition, since later we want to compare the performance of two different problem solvers, it is essential that measurements of Δ be independent of the problem solvers we are testing.

These requirements suggest several possibilities for Δ. For example, if the problems are drawn from a planning domain, the number of steps in the shortest solution to a problem is likely to predict its resource consumption. Another attractive alternative for defining Δ is to use a separate *control problem solver* to provide a benchmark of performance. Here, a measurable aspect of the resource usage of the control system (e.g., the CPU time required to solve the problem) constitutes Δ. To avoid censoring of these values, it is necessary to run the control system with a high resource limit. The high cost incurred can be amortized over multiple experiments that use the same problem set.

For this paper, we adopt the latter approach. The control system used is a single processor version of our WAM-based parallel first-order logic theorem prover described in [18] with subgoal caching and intelligent backtracking disabled. Data were collected using a dedicated 128MB Sun Sparc 670MP "Cypress" system with a resource limit of 5×10^6 node explorations per problem. Of the 87 problems in the LT problem set, 46 problems were solved within the control system resource limit. We exclude the 41 problems not solved by the control system from the analyses below. (Neither the learning nor the non-learning system solved any of these problems.) While this omission does introduce a slight bias into our results, we believe that this bias is negligible: since the control system's resource limit was two orders of magnitude larger than the resource limit for the experimental systems, all these problems correspond to censored datapoints far below the regression line, and ignoring such points has only a small effect on the regression coefficients.

4. Using EM to analyze the LT data

We are now ready to show how to analyze problem-solving performance data using EM. Specifically, we wish to compare the behavior of a backward-chaining problem-solving system on the LT domain with the same system augmented by an explanation-based learning component. We first posit a linear model for the performance of the non-learning system, and show how EM can be used to compute parameters of the model from censored data. Next, we posit a two-component linear model for the performance of the learning system and provide an algorithm to fit this more complex model, again in the presence of censored

[9] For a different experiment, another name for δ_i might be more appropriate. For example, if we were exploring the relationship between quantity administered of a drug and treatment effectiveness, δ_i might be the quantity given to the ith patient. In that case, a high δ_i might predict a strong patient response, while a low δ_i might predict a weaker response.

data. Finally, we compare the performance of the two systems in order to draw some qualitative conclusions based on these analyses.

4.1. A linear model

Consider the non-learning system data from trial 1. Let us assume there is a linear relationship between Δ and $\mathbf{Z}$ such that each $z_i - a\delta_i - b$ is normally distributed with mean 0.[10] This is the standard linear regression model, and, if it were not for the censored data, we would not need EM. In fact, we could still do regular regression if we threw out the censored datapoints: the benefit of EM is that if allows us to keep the censored points in our sample without biasing the regression line. If we threw out these points, we would be wasting potentially valuable information, thereby at best losing some statistical power, and at worst reaching incorrect conclusions.

Before explaining mathematically how to use EM to fit the linear model to censored data, it is useful to understand the effect of censored data intuitively. In an ordinary regression, a datapoint (δ, z) can be seen as "attracting" the regression line towards itself. A point above the line pulls the line upwards, and a point below the line pulls it downwards. A censored datapoint (δ, x)—where x is a lower bound for the true z value—appearing above the line behaves similarly: it pulls the line upwards at least as much as an uncensored datapoint in the same apparent position, since the true position of the censored datapoint is at least as high as its apparent position. In contrast, a censored datapoint below the line does not pull the line downwards, since the true datapoint may actually lie on the line or even above it. In fact, a censored datapoint below the line pushes the line upwards, although perhaps only slightly, since a higher line makes it more likely for this datapoint to be censored.

We do not have to worry about the effects of doubly censored datapoints: this is because we do not try to compare the performance of one system directly with the other (as we did, for example, in our scatter plot comparison of Fig. 1). Instead, we rely on an independent standard, δ_i, and assume that δ_i is available for each i. Thus problems solved by neither the learning nor the non-learning system, which appear as doubly censored points in a direct comparison, are transformed into two singly censored points (one in the learning plot and one in the non-learning plot) in the indirect comparison.[11]

---

[10] In order to achieve (approximately) this distribution, we may have to take the log of Δ, $\mathbf{Z}$, or both, as we do in the experiments below. Without this transform, the variances at one end of the regression line might be much smaller than the variances at the other end. Also, this model implicitly assumes that Δ is not subject to measurement error. Just as in standard linear regression, the lack of measurement error is a convenient fiction which does not seriously influence the results.

[11] Of course, we must also deal with the corresponding disadvantage. If we use a control problem solver to obtain Δ and the control system fails to solve a problem, then we cannot use information about how well the two test problem solvers perform on that problem. Fortunately, in the LT experiment, the control system solves every problem solved by either the learning or non-learning system within the specified control system resource limit.

4.1.1. Fitting a linear model to censored data with EM

When using EM to fit the linear model just described, the M step involves finding maximum likelihood estimates for the values of the model parameters a, b, and σ.

Formally, the model is a probability density function:

$$f(\mathbf{Z}|a, b, \sigma) = \prod_i \frac{1}{\sqrt{2\pi}\sigma} e^{-(z_i-(a\delta_i+b))^2/(2\sigma^2)}. \tag{1}$$

We can find the maximum likelihood estimates for a, b, and σ by differentiating $\ln f$ and setting the result to 0. This process gives the well-known estimates [4]:

$$\hat{a} = \frac{\sum_i \delta_i z_i - \frac{1}{n}\sum_i \delta_i \sum_i z_i}{\sum_i \delta_i^2 - \frac{1}{n}\left(\sum_i \delta_i\right)^2}, \tag{2}$$

$$\hat{b} = \frac{1}{n}\left(\sum_i z_i - \hat{a}\sum_i \delta_i\right), \tag{3}$$

$$\hat{\sigma} = \frac{1}{n}\sum_i (z_i - (\hat{a}\delta_i + \hat{b}))^2. \tag{4}$$

These estimates can be expressed in terms of the sufficient statistics

$$T_3(\mathbf{Z}) = \left[\sum_i z_i, \sum_i z_i^2, \sum_i \delta_i z_i\right] \tag{5}$$

after multiplying out the expression for $\hat{\sigma}$ to obtain elements of $T_3(\mathbf{Z})$.

For the E step, we must find the expected value of $T_3(\mathbf{Z})$ in terms of $\hat{a}$, $\hat{b}$, and $\hat{\sigma}$. Since

$$E(T_3(\mathbf{Z})) = \left[\sum_i E(z_i), \sum_i E(z_i^2), \sum_i \delta_i E(z_i)\right], \tag{6}$$

it is sufficient to find $E(z_i)$ and $E(z_i^2)$ for each i. The trick is to do this in the presence of censored data.

If $y_i = 0$, then $z_i = x_i$, so $E(z_i) = x_i$ and $E(z_i^2) = x_i^2$. However if $y_i = 1$, the situation is more difficult. Assuming for the moment that $\hat{a}\delta_i + \hat{b} = 0$ and $\hat{\sigma} = 1$, the density of z_i is

$$\phi(z) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}z^2}. \tag{7}$$

Let the probability that $z_i > z$ be

$$\Phi(z) = \int_z^\infty \phi(t)\,dt. \tag{8}$$

Conditioning on the fact that $z_i > x_i$ gives the density for z_i given $z_i > x_i$,

$$f(z_i \mid z_i > x_i) = \frac{\phi(z)}{\Phi(x_i)}. \tag{9}$$

The required statistics are then

$$E(z_i \mid z_i > x_i) = \int_{x_t}^{\infty} t \frac{\phi(t)}{\Phi(x_i)}\,\mathrm{d}t \tag{10}$$

$$= \frac{\phi(x_i)}{\Phi(x_i)}, \tag{11}$$

since $\int t\phi(t)\,\mathrm{d}t = -\phi(t) + C$, and

$$E(z_i^2 \mid z_i > x_i) = \int_{x_t}^{\infty} t^2 \frac{\phi(t)}{\Phi(x_t)}\,\mathrm{d}t \tag{12}$$

$$= \frac{1}{\Phi(x_t)} \left(-t\phi(t) + \int \phi(t)\,\mathrm{d}t \right) \Bigg|_{x_t}^{\infty} \tag{13}$$

$$= \frac{1}{\Phi(x_t)} (x_i \phi(x_i) + \Phi(x_i)) \tag{14}$$

$$\equiv x_i \frac{\phi(x_i)}{\Phi(x_i)} + 1. \tag{15}$$

Shifting and scaling to handle arbitrary $\hat{a}$, $\hat{b}$, and $\hat{\sigma}$ gives

$$E(z_t \mid z_i > x_i, \mu_i, \hat{\sigma}) = \mu_i + \hat{\sigma} \frac{\phi\left(\frac{x_i - \mu_i}{\hat{\sigma}}\right)}{\Phi\left(\frac{x_i - \mu_i}{\hat{\sigma}}\right)} \tag{16}$$

and

$$E(z_i^2 \mid z_i > x_i, \mu_i, \hat{\sigma}) = \mu_i^2 + 2\mu_i \hat{\sigma} \frac{\phi\left(\frac{x_i - \mu_i}{\hat{\sigma}}\right)}{\Phi\left(\frac{x_i - \mu_i}{\hat{\sigma}}\right)} + \hat{\sigma}^2 \left(1 + \frac{x_i - \mu_i}{\hat{\sigma}} \frac{\phi\left(\frac{x_i - \mu_i}{\hat{\sigma}}\right)}{\Phi\left(\frac{x_i - \mu_i}{\hat{\sigma}}\right)} \right) \tag{17}$$

where $\mu_i = \hat{a}\delta_i + \hat{b}$.

4.1.2. Application to the LT non-learning data

Fig. 2 plots non-learning CPU time versus δ_i for the 46 problems for which δ_i is available, with a logarithmic transform applied to both axes. The line shown in Fig. 2 is the censored linear regression fit found by EM using the 34 solved and 12 censored problems.

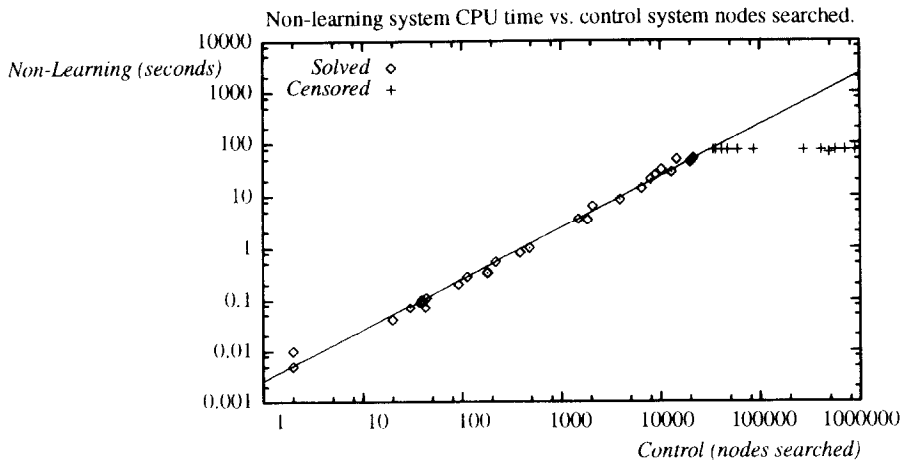

Fig. 2. Plot of non-learning system CPU time versus control system nodes searched. The "diamond" datapoints correspond to the 34 problems solved both by the control system and by the non-learning system, while the 12 "cross" datapoints correspond to censored problems solved only by the control system. The line is the result of using EM to perform censored linear regression.

While it is possible to obtain a substantially similar fit using a simple linear regression on only the 34 problems for which both δ_i and z_i are available, the fit obtained by EM exploits the additional information available from the 12 censored problems. A regular regression line on all 46 problems would have an incorrect, smaller, slope because the censored datapoints would pull it downwards.

4.2. A multi-component model

In order to complete the analysis of the LT experiment, we must now analyze the learning data. We could perform the same analysis again, but experience with learning systems suggests a different model is more appropriate. In this section, we introduce a model that is a mixture of two submodels [1] for learning system performance. The premise is that the behavior of some speedup learning systems is a combination of two behavioral modes. We show how EM can be used to model these behavioral modes separately in the presence of censored data.

4.2.1. Justifying a two-component model of speedup learning

Speedup learning algorithms generally operate by perturbing the search space explored by a problem-solving system. The exact nature of the perturbation depends on what has been learned from previous problem-solving experience, e.g., cache contents in the case of subgoal caching, and learned rules or search heuristics in the case of explanation-based learning. Typically certain problems are "helped" by the learned information, while other problems are mostly unaffected by what has been learned previously. Speedup learning performance is

thus a good candidate for a two-component model, where performance on a subset of problems is changed by learned knowledge, and performance on the remaining problems is largely unchanged.[12]

More precisely, assume that, when augmented with a speedup learning mechanism, the system displays two distinct linear relationships between difficulty Δ and resource consumption Z: with probability $1-\lambda$, a given problem is mostly unaffected by learning and the point (δ_i, z_i) lies approximately on one line, while with probability λ, learned knowledge contributes to the solution of the problem and the point (δ_i, z_i) lies along a different, lower, line. The number λ is called the *mixing parameter*. Given performance data like that collected in trial 2 of the LT experiment, expectation maximization can estimate λ and assign each datapoint to one of the two subpopulations, as well as simultaneously characterize both model relationships in the presence of censored data.

The next natural step after the above two-component censored linear model is a model with three or more components. The techniques described below can be extended to multiple-component models, but we need only two components to analyze the LT data.

4.2.2. *Fitting a two-component linear model to censored data with EM*

The two-component linear model just described is the natural extension of the simple linear model of Section 4.1, given the hypothesis that there are two subpopulations. Here we show how to estimate the parameters of such a model in the presence of censored data using EM. Others have described an algorithm based on two nested EM iterations to fit this model [8]. The algorithm proposed here is more efficient because it needs only one level of iteration.

In the simple censored regression case, the unobserved vector of true resource consumptions Z gives rise to observed resource consumptions X and censoring flags Y. In the more complicated two-component model, there are more unobserved variables: in addition to the true resource consumptions Z, we introduce an $n \times 2$ matrix of unobserved data V telling which population each problem belongs to. The element v_{ij} of V is defined to be 1 if observation i belongs to population j, and 0 otherwise.[13] As before, we estimate V and Z from the observed variables X and Y using EM. The estimates for V and Z allow us to infer values for the mixing parameter and the slope and intercept of each population.

---

[12] The addition of learned knowledge may adversely affect the performance of the problem solver on those problems not directly helped by learning. This *utility problem* is often associated with the use of EBL algorithms as well as other speedup learning techniques [9]. The method of analysis advanced here can clarify how strongly the utility problem affects a particular experiment, as explained in Section 4.3.

[13] In some situations we may have outside knowledge about v_{ij}. We can encode such knowledge in a prior distribution for the v_{ij}. For example, it may be possible to determine by inspection if any learned macro-operator is employed in a solution. However, such outside information is not always available (e.g., failure caching and intelligent backtracking generally leave no trace in the solution generated) or reliable (e.g., using learned knowledge is a necessary but not sufficient condition for speedup).

To derive the M step of the EM algorithm, we need the density function for the unobserved data. For convenience, we give the logarithm of this density:

$$-\ln(f(\mathbf{Z}, \mathbf{V} \mid \mu_{t0}, \mu_{t1}, \sigma_0, \sigma_1, \lambda))$$
$$= \sum_t \left(v_{t0}\left(\frac{1}{2\sigma_0^2}(z_t - \mu_{t0})^2\right) + v_{t1}\left(\frac{1}{2\sigma_1^2}(z_t - \mu_{t1})^2\right) - v_{t0}\ln(\lambda) - v_{t1}\ln(1-\lambda) \right.$$
$$\left. + v_{t0}\ln(\sigma_0) + v_{t1}\ln(\sigma_1)\right) + C \tag{18}$$

where $\mu_{t0} = a_0\delta_t + b_0$ and $\mu_{t1} = a_1\delta_t + b_1$ and C is a constant.

To reduce the number of parameters to estimate, we assume that we know a priori the variances σ_0^2 and σ_1^2 of the two populations about their regression lines.[14] There are five remaining parameters: the slope and intercept for each of the two lines and the mixing parameter λ. As before, we find their maximum likelihood estimates by differentiating $\ln f$ and setting the result equal to zero. The resulting estimate for the mixing parameter is exactly what one might expect, namely the fraction of datapoints that belong to the first line:

$$\hat{\lambda} = \frac{1}{n}\sum_t v_{t0}\,. \tag{19}$$

The estimates for the slope and intercept of the first line are

$$\hat{a}_0 = \frac{\sum_t v_{t0}\delta_t z_t - \dfrac{\sum_t v_{t0}\delta_t \sum_t v_{t0} z_t}{\sum_t v_{t0}}}{\sum_t v_{t0}\delta_t^2 - \dfrac{\left(\sum_t v_{t0}\delta_t\right)^2}{\sum_t v_{t0}}}\,, \tag{20}$$

$$\hat{b}_0 = \frac{\sum_t v_{t0} z_t - \hat{a}\sum_t v_{t0}\delta_t}{\sum_t v_{t0}}\,. \tag{21}$$

These estimates are similar to the estimates for the single-line case (Eqs. (2) and (3)), except that every term has an additional factor of v_{t0} so that the sums include only those points that belong to the first line. In particular, $n = \Sigma_t 1$ is replaced by $\Sigma_t v_{t0}$.

[14] Trying to estimate these variances adds many local maxima to the EM search space. These are the models where one line latches onto just a few points, fits them (almost) exactly, and thus has variance (almost) zero. If the true variances are unknown, as is usually the case, we recommend trying several variances for each line, both to find the best fit and to determine how sensitive the fit is to the choice of variances.

The estimates for the slope and intercept of the second line are analogous to the estimates for the first line.

We can compute these estimates from the sufficient statistics[15]

$$T_{\Delta}(\mathbf{Z},\mathbf{V}) = \Big[\sum_i v_{i0}z_i, \sum_i v_{i0}z_i\delta_i, \sum_i v_{i0}\delta_i, \sum_i v_{i0}\delta_i^2, \sum_i v_{i1}z_i, \sum_i v_{i1}z_i\delta_i, \sum_i v_{i1}\delta_i, \sum_i v_{i1}\delta_i^2, \sum_i v_{i0}\Big]. \tag{22}$$

These formulas constitute the M step for the EM algorithm.

For the E step, we must find the expected value of each of the sufficient statistics given the current estimates of a_0, a_1, b_0, b_1, and λ. Since v_{i0} and v_{i1} may only take the values 0 and 1, we can compute these expected values as for the single-line case. For example, $E(v_{i0}z_i) = E(v_{i0})E(z_i \mid v_{i0} = 1)$, and we can calculate $E(z_i \mid v_{i0} = 1)$ using Eq. (16) with $\mu_i = \mu_{i0}$.

The only remaining calculation is $E(v_{i0})$, which we can derive from Bayes' rule and the normal density function:

$$E(v_{i0}) = \frac{w_{i0}}{w_{i0} + w_{i1}}, \tag{23}$$

$$E(v_{i1}) = \frac{w_{i1}}{w_{i0} + w_{i1}}, \tag{24}$$

where

$$w_{i0} = \frac{\lambda}{\sigma_0}\phi\left(\frac{x_i - \mu_{i0}}{\sigma_0}\right), \tag{25}$$

$$w_{i1} = \frac{(1-\lambda)}{\sigma_1}\phi\left(\frac{x_i - \mu_{i1}}{\sigma_1}\right) \tag{26}$$

in the uncensored case, and

$$w_{i0} = \lambda\Phi\left(\frac{x_i - \mu_{i0}}{\sigma_0}\right), \tag{27}$$

$$w_{i1} = (1-\lambda)\Phi\left(\frac{x_i - \mu_{i1}}{\sigma_1}\right) \tag{28}$$

in the censored case.

In practice, rather than computing the slopes and intercepts directly, we perform two weighted regressions, one to find $\hat{a}_0$ and $\hat{b}_0$ and one to find $\hat{a}_1$ and $\hat{b}_1$. For the first regression, we use weights $E(v_{i0})$ and treat censored points as if they all came from the first line, while for the second we use weights $E(v_{i1})$ and treat censored points as if they all came from the second line.

[15] We have defined nine statistics in order to estimate five parameters. Many common distributions only need one statistic per parameter, but the mixture distribution is not so well behaved. We need all nine of these statistics to make the log likelihood linear in $T_{\Delta}(\mathbf{Z},\mathbf{V})$.

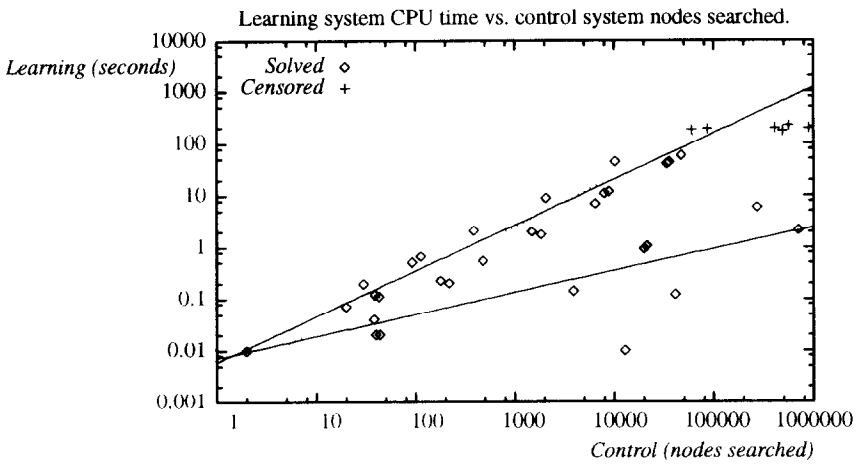

Fig. 3. Plot of learning system CPU time versus control system nodes searched. The 34 "diamond" datapoints correspond to the 34 problems solved by the control system and the learning system, while the eight "cross" datapoints correspond to censored problems solved only by the control system. The two solid lines are the result of using EM to fit the two-component linear model, and the dotted line is the censored linear regression from Fig. 2 (included for comparison).

4.2.3. *Application to the LT learning data*

Fig. 3 shows the 34 LT problems solved by the learning system within the 5×10^4 node exploration resource limit and the eight censored problems solved only by the control system. The remaining $46 - 34 - 8 = 4$ problems were used as input for the learning mechanism. As for the non-learning system, only the 46 problems for which a control system node exploration count is available are considered. The two solid lines shown in Fig. 3 are the result of fitting the two-component linear model using fixed variances of 1.0 and 0.5 for the two components. The dotted line is the censored linear regression from Fig. 2, included for comparison.

Recall the slope of a line represents the relationship discovered between "difficulty" and resource consumption for some population of datapoints, where a smaller (larger) slope corresponds to a faster (slower) system. Since the slopes of the upper solid line of Fig. 3 and the dotted line imported from Fig. 2 are comparable, we conclude that the performance of the learning system on the subpopulation of problems corresponding to the upper solid line is similar to the performance of the non-learning system on the entire set of problems. This is consistent with the original intuition underlying our two-component model (see Section 4.2.1), that is, that speedup learning is applied selectively, leaving performance on a portion of the problems largely unchanged. In a similar fashion, since the lower solid line has a smaller slope than the dotted line, we conclude that the subpopulation of problems corresponding to the lower solid line was noticeably helped by learning.

4.3. The benefits of EBL*DI

What does the overall analysis say about the performance of the EBL*DI algorithm in the LT domain? First, the macro-operators produced by EBL*DI are shown to be useful, since datapoints lying on the lower line correspond to problems that are solved noticeably faster after learning. The estimated mixing parameter $\hat{\lambda}$ indicates approximately how many problems were helped by EBL*DI: here it is 0.324, or about 3 of every 8 problems. To confirm that EBL*DI is the source of the difference between the two lines, i.e., that EM is not finding a spurious distinction, we can examine the 27 problems whose solutions contain a learned macro-operator. These problems contain all 13 of the points EM assigns to the lower line, showing that the use of a learned rule is a necessary (although not sufficient) condition for being helped by learning.[16]

Second, there is little evidence here of the utility problem. This problem would manifest itself as an increase in the slope of the upper solid line with respect to the dotted line imported from Fig. 2. This would indicate that, even though one subpopulation of problems might be helped by learning (the lower solid line), the other subpopulation was hindered by learning. Here, since the two lines have comparable slopes, we can conclude that the utility problem is not an issue. These observations are consistent with our own previously reported results [14], where we compared the performance of the EBL*DI algorithm and another EBL algorithm drawn from the machine learning literature.

5. Discussion

This paper has shown how model fitting is valuable in the analysis of speedup learning data. Model fitting goes beyond hypothesis testing to provide a deeper understanding of experimental results. More specifically, model fitting—when applied to the multi-component model of speedup learning proposed in this paper—can provide information like how often the learning system solves problems faster (the mixing parameter λ), the magnitude of the typical speedup (the ratio of the slope of the lower line in the learning analysis to the slope of the line in the non-learning analysis), and the effect of learning on "unhelped" problems (the ratio of the slope of the upper line in the learning analysis to the slope of the line in the non-learning analysis). This is much finer-grained information than the typical "it is not the case that the learning system is the same as the non-learning system" conclusion provided by hypothesis testing.

We have illustrated how to fit our multi-component model of speedup learning to real data obtained from experiments with a particular EBL algorithm on a

[16] These are the 13 points whose estimated probability of belonging to the lower line is greater than one half. Note that $13/42 \neq \hat{\lambda} = 0.324$. There is no contradiction here, since the estimate of λ is formed from the raw (not the thresholded) probabilities.

problem set taken from the machine-learning literature. In order to fit this model, we have applied the statistical technique of expectation maximization (EM), both to handle censored data and to provide a probabilistic partitioning of datapoints between the two submodels. Whenever one suspects that experimental data arise from distinct subpopulations, it is valuable to eliminate human bias in identifying the subpopulations by using EM to fit a multi-component model. In the case of problem-solving performance data, the prior evidence that multiple subpopulations exist may be relatively shallow (e.g., based on visual inspection of the data) or relatively deep (e.g., based on the selective use of macro-operators produced by learning).

Ultimately, any exploratory data analysis tool is justified only if it is useful in practice. Recall that in our own previous work [14], we claimed that macro-operators produced by EBL*DI were both more effective and less likely to cause the utility problem than the macro-operators produced by a given other EBL algorithm when operating in the LT domain (as well as several others). These conclusions were based on relatively coarse-grained comparisons of summary statistics collected from experimental trials much like the ones described here, and were only credible because censoring was never an issue (the EBL*DI learning system solved every problem solved by both the non-learning control system and the other EBL system within the allotted resource bound). Had this not been the case, we would not have been able to support our claim without access to an analysis technique like the one advocated here. More to the point, however, is that if one were to remove from Fig. 3 the lines found by EM, simple visual examination of the data would reveal little structure. Only through the use of EM and our two-component linear model is the bimodal structure of the data revealed.

The message of this paper is that parametric analysis can give valuable insight into experimental data, even when hypothesis testing is inconclusive. In situations where non-parametric methods (e.g., the Wilcoxon signed-ranks test) are preferable for hypothesis testing, one may still gain useful understanding of empirical data by positing a parametric model and using EM to estimate values for the parameters of the model. Modeling gives more information than hypothesis testing, and compared to traditional model-fitting methods, EM lets one use more complicated models.

Acknowledgements

The authors wish to thank Paul Cohen and a second, anonymous, reviewer for their constructive comments on a draft version of this paper. Support for this research was provided by the Office of Naval Research through grants N0014-88-K-0123, N00014-90-J-1542, and N0014-94-1178 (AMS), by the Advanced Research Project Agency through Rome Laboratory Contract Number F30602-93-C-0018 via Odyssey Research Associates, Incorporated (AMS), by a National Science Foundation graduate fellowship (GJG), by the National Science Founda-

tion through grant BES-9402439 (GJG), and by a Hellman faculty fellowship (CPE).

References

[1] M. Aitkin and D.B. Rubin, Estimation and hypothesis testing in finite mixture models, *J. Roy. Stat. Soc.* **47** (1985) 67–75.
[2] J. Arbuthnott, An argument for divine providence, taken from the constant regularity observed in the births of both sexes, *Philos. Trans.* **27** (1710) 186–190.
[3] T.L. Bailey and C.P. Elkan, Unsupervised learning of multiple motifs in biopolymers using expectation maximization, *Mach. Learn.* (to appear).
[4] G. Casella and R. Berger, *Statistical Inference* (Brooks/Cole Publishing Company, Pacific Grove, CA, 1990).
[5] A.P. Demptster, N.M. Laird and D.B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, *J. Roy. Stat. Soc. B* **39** (1977) 1–37.
[6] O. Etzioni and R. Etzioni, Technical note: statistical methods for analyzing speedup learning experiments, *Mach. Learn.* **14** (1994) 333–347.
[7] J.N. Hooker, Needed: an empirical science of algorithms, *Oper. Res.* **42** (1994) 201–212.
[8] K. Jedidi, V. Ramaswamy and W. Desarbo, A maximum likelihood method for latent class regression involving a censored dependent variable, *Psychometrika* **58** (1993) 365–394.
[9] S. Minton, Quantitative results concerning the utility of explanation-based learning, *Artif. Intell.* **42** (1990) 363–392.
[10] R. Mooney, The effect of rule use on the utility of explanation-based learning, in: *Proceedings IJCAI-89*, Detroit, MI (1989), 725–730.
[11] A. Newell, J.C. Shaw and H. Simon, Empirical explorations with the logic theory machine: a case study in heuristics, in: E. Feigenbaum and J. Feldman, eds., *Computers and Thought* (McGraw Hill, New York, 1963).
[12] P. O'Rorke, LT revisited: explanation-based learning and the logic of Principia Mathematica, *Mach. Learn.* **4** (1989) 117–160.
[13] A.M. Segre, On combining multiple speedup techniques, in: *Proceedings Ninth International Conference on Machine Learning*, Aberdeen (1992) 400–405.
[14] A.M. Segre and C. Elkan, A high-performance explanation-based learning algorithm, *Artif. Intell.* **69** (1994) 1–50.
[15] A.M. Segre, C. Elkan and A. Russell, Technical note: a critical look at experimental evaluations of EBL, *Mach. Learn.* **6** (1991) 183–196.
[16] A.M. Segre, C.P. Elkan, D. Scharstein, G.J. Gordon and A. Russell, Adaptive inference, in: A. Meyrowitz and S. Chapman, eds., *Foundations of Knowledge Acquisition Vol. 2* (Kluwer Academic Publishers, Boston, MA, 1993) 43–81.
[17] A.M. Segre and D. Scharstein, Bounded-overhead caching for definite-clause theorem proving, *J. Autom. Reasoning* **11** (1993) 83–113.
[18] A.M. Segre and D.B. Sturgill, Using hundreds of workstations to solve first-order logic problems, in: *Proceedings AAAI-94* Seattle, WA (1994) 187–192.
[19] A.N. Whitehead and B. Russell, *Principia Mathematica* (Cambridge University Press, Cambridge, 1913).
[20] F. Wilcoxon, Individual comparisons by ranking methods, *Biometrics* **1** (1945) 80–83.