Learning from Bad Data

Foster John Provost*
NYNEX Science & Technology, Inc.
400 Westchester Ave.
White Plains, NY 10604
foster@nynexst.com

Andrea Pohoreckyj Danyluk
Williams College
Department of Computer Science
Williamstown, MA 01267
andrea@cs.williams.edu

Abstract

The data describing resolutions to telephone network local loop "troubles," from which we wish to learn rules for dispatching technicians, are notoriously unreliable. Anecdotes abound detailing reasons why a resolution entered by a technician would not be valid, ranging from sympathy to fear to ignorance to negligence to management pressure. In this paper, we describe four different approaches to dealing with the problem of "bad" data in order first to determine whether machine learning has promise in this domain, and then to determine how well machine learning might perform. We then offer evidence that machine learning can help to build a dispatching method that will perform better than the system currently in place.

1 INTRODUCTION

The data describing resolutions to telephone network local loop "troubles," from which we wish to learn rules for dispatching technicians, are notoriously unreliable. Anecdotes abound detailing reasons why a resolution entered by a technician would not be valid, ranging from sympathy to fear to ignorance to negligence. Initial attempts at learning from these data were not promising. However, data are plentiful and the task is an enormous one. A small increase in accuracy can have a large impact on the company's bottom line. For example, if we are willing to ignore details for the moment, New York State alone has over 300,000 residential trouble reports per month; if an erroneous dispatch costs the company (on the average) \$100, then every 1% decrease in dispatch error rate can save the company over **\$3** million annually. Thus, however poor the quality of the data, it is worthwhile to investigate methods for increasing the accuracy of local loop trouble diagnosis.

In this paper, we describe four different approaches to dealing with the problem of "bad" data in order first to determine whether machine learning has promise in this domain, and then to determine how well machine learning might perform. First, we use an existing expert system as a model to generate clean data from which we can learn rules that dispatch almost perfectly (with respect to the expert system's model). Second, we gather expert analyses of a small set of data and show that it is likely machine learning can also model the behavior of experts. Third, we show that learning from the field (technicians') data may, in fact, be better than we had previously believed. Fourth, we use prior knowledge to "clean up" the field data and show that we can learn quite well from the cleaned-up data.

Finally, we offer evidence that machine learning can help to build a dispatching method that will perform better than the system currently in place.

2 NYNEX MAX AND THE LOCAL LOOP MAINTENANCE DOMAIN

2.1 OVERVIEW

MAX (Rabinowitz *et al* 1991) is an expert system developed by NYNEX Science and Technology[1] for the purpose of troubleshooting customer-reported telephone problems. MAX deals specifically with problems in the local loop, the part of the telephone network between the central office and the customer's premises.

When a customer of the phone company has difficulty with his telephone line he calls the phone company to report the problem (the *trouble*). A phone company representative creates a trouble report and also

*Presented at the workshop on *Applying Machine Learning in Practice* at IMLC-95.

[1] NYNEX is the parent company of NYNEX New York and NYNEX New England, formerly New York Telephone and New England Telephone.

initiates electrical tests on the customer's line, called the Mechanized Loop Test (MLT).[2] The MLT measures the electrical signature of the customer's line and gives such information as voltages and resistances. This information is then interpreted by a primitive rule-based system called the Screening Decision Unit (SDU), which gives a diagnosis based upon a summary of the MLT results. All this information is then sent to a Maintenance Administrator (MA) who determines a high-level diagnosis for the trouble.

MAX (Maintenance Administrator EXpert) plays the role of an MA. It gives a high-level diagnosis of a trouble based upon MLT results, the SDU recommendation, and other information about the customer. The high-level diagnosis determines roughly in which part of the customer's line the trouble lies. There are five possible diagnoses: (1) dispatch to the cable; (2) dispatch to the distribution wiring; (3) handle the trouble in the central office; (4) do further testing; or (5) send to a human MA for diagnosis.

2.2 APPLYING MACHINE LEARNING

MAX is a rule-based system of approximately 150 rules.[3] It has been used successfully throughout NYNEX's phone companies in New York and New England. To handle regional differences as well as differences over time, MAX's developers built into the system a set of parameters that may be tuned for different locations. The parameters essentially define thresholds for rule application. While the parameters are a valuable concept in theory, they are difficult to tune in practice, largely due to their subtle interactions in the rules.

The problem of tuning MAX for local conditions is a particularly promising application for machine learning for the following reasons: (1) Diagnosis in this domain is a static problem. That is, all data are gathered and a diagnosis (or classification) is then based on the values given. Difficult problems such as incorporating time into the solution are not an issue here. (2) Data are abundant. (3) A knowledge base already exists, providing a wealth of information about the domain.

The appeal of learning in this domain is its potential for generating dispatch knowledge that captures local differences. Learning is also appealing because of its potential for tracking changes in dispatch knowledge as the network equipment degrades or is replaced with new equipment.

Several approaches to the problem of tuning MAX have been investigated: (1) The application of inductive learning to generate completely new knowledge bases for specific locations (Danyluk & Provost 1993a). (2) The application of analytic and inductive learning to modify the existing knowledge base for specific locations (Pazzani & Brunk 1993; Goodman 1989). (3) The application of techniques to perform parameter tuning (Merz *et al* 1994). This paper discusses the first of these only.

2.3 MACHINE LEARNING RESULTS REPORTED IN THIS PAPER

All results reported in this paper were generated using C4.5 (Quinlan 1992) with default settings.[4] Results given are after pruning. C4.5 is trained and then tested on MAX data, where specific class labels have come from a variety of sources: MAX itself, technicians who have been dispatched to solve a problem, or experts in the domain of trouble dispatch. Success of C4.5 on the training set is measured in two ways: (i) we measure error rate on independent test sets; and (ii) we measure the percentage decrease in error rate (PDER) of the learned concept description over the error rate of the default class. This indicates the extent to which the learned decision tree decreases the error rate that would result from always selecting a default class. The default class is taken to be the most frequently occurring class in any particular training set.

Numbers of test examples are given with each set of runs. All results reported have been averaged over 10 runs with training and test sets chosen randomly. Unless indicated otherwise, all data used in the runs in this paper are taken from a single site during a period of approximately 8 months.

3 RESULTS

In this section we report results of running C4.5 on data for this domain. We discuss the performance of C4.5 when considering 3 different sources for the class labels given to training and test examples. In the runs reported, 22 features have been used to describe examples (though results with a subset of 14 features have been similar). The 22 features used are, with one exception, the features used by MAX for diagnosis. Throughout this section, we use "the default" as a shorthand for "classifying all cases identically using the most frequently occurring class."

3.1 MODELING MAX

As reported previously (Danyluk & Provost 1993a, b), in order to evaluate the potential of machine learning

[2] MLT is a product of AT&T.

[3] It is difficult to give an accurate estimate of the size as some rules are fairly short while others are quite complex and are essentially the equivalent of many smaller rules.

[4] Earlier results were obtained with other systems (see Section 2.2), but C4.5 consistently has yielded results that are at least as good as the other systems and has done so more efficiently.

in this domain, we used the existing MAX expert system to create a "clean" data set from which to learn. MAX is currently in regular use across the NYNEX corporation for the dispatch of technicians for local-loop troubles. We ran a series of experiments with the goal of showing that given good data we could learn to dispatch well. The major assumption of this approach is that MAX is dispatching correctly.

As the results in Table 1 show, given a large enough quantity of data, using machine learning we can duplicate MAX's performance almost perfectly. One study in which experts were asked to analyze the troubles on which C4.5 failed to model MAX indicated that C4.5's decision was better than MAX's in approximately 50% of the cases. Although these results show promise for machine learning as a method of creating the knowledge base for a dispatch system, they do not offer a solution to the problem of generating knowledge that will necessarily increase the performance of MAX.

Table 1: C4.5 results: Classes = MAX dispatches. Size of test set = 4874. ER = Error Rate

Training Set	ER on Test Set	StDev
100	29.18	2.20
1000	9.60	0.69
10000	2.54	0.25

3.2 MODELING EXPERTS

In order to evaluate the potential of machine learning as a tool to build a better MAX, we enlisted the help of several experts in local loop trouble-shooting. The experts were phone company veterans with many years of experience in the areas of maintenance and repair of the local loop.

We ran a set of experiments testing the ability to learn dispatch knowledge from expert-classified data. The rationale behind this set of experiments is that if machine learning can create knowledge that models the behavior of human experts well, then it may be possible, albeit resource consuming, to have local experts analyze large numbers of troubles and learn new dispatch knowledge from these data.

As the results in Table 2 show, for one expert who analyzed 500 troubles from a site with which the expert is very familiar, C4.5 can model the expert's behavior fairly well as compared to the default. A similar analysis of other experts' answers have yielded comparable results. Results are given for one expert only for several reasons: 1) No two experts, of the 5 experts surveyed, agreed upon diagnoses more than 65% of the time. This might be evidence for the differences that exist between sites, as the experts surveyed had gained their expertise at different locations. If not, however, it raises questions about the correctness of the expert data. 2) Troubles analyzed by the experts were taken from 2 different sites. Experts who had been given data from one of the sites indicated that the electrical readings appeared to be questionable, probably due to a problem with MLT. Results given in Table 2 are for the better site.

Unfortunately, the size of the data set in these experiments was limited. The results shown in Table 1 suggest that 400 examples may be too few for effective learning. Analysis of the concept description learned by MAX explains why many examples are needed: very small disjuncts comprise a large portion of the concept description (Danyluk & Provost 1993a). However, the results of these experiments are promising with respect to the potential for machine learning to model the behavior of human experts.

Table 2: C4.5 results: expert classes. Size of test set = 100. Error rate with default class = 57.8.

Training Set	ER on Test Set	StDev	PDER
100	38.6	4.22	33.22
200	35.9	5.20	37.89
300	34.4	3.17	40.48
400	35.3	3.68	38.93

3.3 MODELING THE FIELD DATA

Questions as to the ability of MAX and the experts to dispatch accurately (see Sections 3.2 and 5) led us to revisit learning from the field (technicians') data. We wanted to characterize how well machine learning would perform on these data, and whether we could do anything to increase the performance. As the results in Table 3 show, the performance of the learned decision trees is less than inspiring. However, the learned trees do perform slightly better than the default.[5]

Table 3: C4.5 results: classes = technicians' results. Size of test set = 863. Error rate of default = 61.6

Training Set	ER on Test Set	StDev	PDER
100	60.98	3.26	1.01
500	59.27	1.81	3.78
1000	58.80	1.12	4.54
5000	57.54	1.23	6.59

Quite surprisingly, we were able to significantly in-

[5] Although MAX may classify a trouble in one of 5 ways, there are only 4 dispatch classes that correspond to the field technicians' diagnoses. (MAX can send a trouble to a human MA for dispatch.)

crease our ability to dispatch accurately by reducing the feature set to a single feature: vercode. Vercode is essentially a summary of the electrical readings produced by MLT. MAX was originally designed with the goal of using additional information to increase the performance of vercode alone. As the results in Table 4 show, the decision stumps (Holte 1993) learned by C4.5 on the field data perform much better than those learned with larger feature sets.

Table 4: C4.5 results: classes = technicians' results; vercode only. Error rate of default = 61.6

Training Set	ER on Test Set	StDev	PDER
100	62.40	4.47	-1.30
500	54.42	1.45	11.65
1000	53.43	1.64	13.26
5000	51.74	0.87	16.01

4 CLEANING UP THE DATA

4.1 MODELING CLEANED-UP FIELD DATA

By analyzing the different trouble resolutions reported by the field technicians, it becomes clear that machine learning programs would have a difficult time modeling the data. For some borderline resolutions, it is not clear what the correct dispatch should have been, either because the resolution does not provide a diagnosis or because the diagnosis cannot be unambiguously mapped to a dispatch. Furthermore, there are many cases for which the resolution is a "Test OK." This resolution indicates that the technician retested the line in the process of attempting to locate the trouble, and found that there was not a problem. These cases are particularly troublesome for a machine learning program because it is impossible to know what the "correct" dispatch should have been. For example, if the trouble is a short due to water in a cable that has dried by the time the technician retests the line, the correct dispatch should be "dispatch to cable," because the trouble is very likely to reoccur during the next heavy rain. Unfortunately, it is impossible to tell the difference between cases where there was no problem to begin with and cases where the problem was transient. These cases are placed together with cases where there are not enough data (and thus a retest is needed) into a catch-all "retest" dispatch.

In order to evaluate how machine learning might perform on reliable field data, we used prior knowledge of trouble resolutions and dispatches to remove sources of confusion from the field data. We ran a set of experiments to test the hypothesis that cleaning up the data will elicit better learning performance (lower error rates).

As the learning results in Tables 5 and 6 show, the performance on the cleaned-up data is considerably better than the performance on the original field data. It is important to note that the cleaned-up data have only 3 classes instead of 4, and using the default yields a lower error rate than on the previous data. However, as the results in Table 5 and Table 6 show, the percentage decrease in error rate (PDER) for the learned concept descriptions (in particular, for the vercode decision stump as shown in Table 6) is larger on the cleaned-up data than on the original data. This indicates that, in both absolute terms and relative to the performance of the default, we can learn more accurate concept descriptions from the cleaned-up field data than from the original data.

Table 5: C4.5 results: Cleaned data. Size of test set = 686. Error rate of default = 47.13

Training Set	ER on Test Set	StDev	PDER
100	41.46	3.92	12.03
500	38.00	3.08	19.37
1000	37.35	1.90	20.75
2000	36.17	1.63	23.25

Table 6: C4.5 results: Cleaned data; vercode only

Training Set	ER on Test Set	StDev	PDER
100	38.11	4.31	19.14
500	34.62	1.58	26.54
1000	34.51	1.20	26.77
2000	34.16	1.52	27.52

4.2 MODELING A TWO-CLASS PROBLEM: IN VS. OUT

In the previous section we cleaned up the field data using prior knowledge, in order to reduce the confusion between the three dispatch classes: dispatch to the central office, dispatch to a cable technician, and dispatch to an outside repair technician. The latter two dispatches address problems in the "outside plant." There are *a priori* reasons why it might be desirable to combine these classes into a single "dispatch out" class. One such reason is that often a repair technician can fix a minor cable problem without operating on the cable (*e.g.*, by "swapping pairs"). A common practice is to dispatch to the repair technician first; if he cannot fix the problem he reroutes it to a cable technician. Experts have suggested that being able confidently to differentiate between dispatching "in" *versus* "out" is desirable.

In order to test the hypothesis that we could differentiate accurately between dispatching in and dispatching out, we combined the two outside plant dispatches in the cleaned-up dataset. This modification also allowed us to reinsert the cases that were borderline between repair and cable, so the example sets were larger than those in the previous section.

As the results in Tables 7 and 8 show, C4.5 was able to learn decision trees that dispatch very accurately. In particular, the vercode decision stumps have an average error rate of less than 7%. As in the previous section, it is important to note that by combining classes the default error rate decreased. In this case, the error rate for the default is just over 9%. The tables show the percentage decrease in error rate over the error rate of the default dispatch for these experiments. In sum, although the absolute decrease in error rate is smaller for the *In vs. Out* data than for the cleaned-up data of the previous section, the percentage decrease is comparable.

Table 7: C4.5 results: In vs Out. Size of test set = 738. Error rate of default = 9.4.

Training Set	ER on Test Set	StDev	PDER
100	9.37	0.82	0.36
500	9.46	0.97	- 0.60
1000	8.48	1.61	9.82
2000	7.06	0.65	24.92
3000	7.27	0.63	22.69

Table 8: C4.5 results: In vs Out; vercode only

Training Set	ER on Test Set	StDev	PDER
100	9.39	0.83	0.15
500	9.39	0.83	0.15
1000	9.39	0.83	0.15
2000	7.41	1.39	21.20
3000	6.71	0.78	28.65

5 COMPARISON WITH EXISTING METHODS

A comparison with existing methods will be a major component of NYNEX's final decision as to whether learned knowledge can help with the local-loop dispatch problem. Table 9 shows a comparison of the performance of the vercode decision stumps with the MAX expert system on the same field data. The additional possible dispatches for the MAX expert system complicate the comparison; MAX can opt to route a case to a human (PSH) or can request additional tests (PDT). Of the three data sets included in Table 9, only Field Data contains PDT as a possible correct dispatch, and none contains PSH. Fortunately for the comparison, on these data MAX chose PSH less than 1% of the time. However, MAX chose PDT often. To facilitate comparison, in Table 9 we report the error rate of MAX when compared against all data in each data set. An answer of PDT is considered to be an error if the field result is labeled with a dispatch (PDI, PDO, or PDF). Since no PDTs appear in Cleaned Field Data or In vs Out Only, MAX is considered to be in error if it chose not to make a dispatch. To temper the harshness of this comparison, we also compared the error rate for MAX when it chose to make a dispatch (PDI, PDO, or PDF). Column MAXD gives MAX's error rate on those cases where it chose one of the three dispatches. Column MAXD-C gives the percentage of examples in each data set on which MAX chose one of the three dispatches. Please note that these figures are for one preliminary study and are intended to suggest that there may be room for improvement in the existing system. They should not be interpreted as a comprehensive evaluation of the performance of the existing system.

Table 9: Comparison of error rates of Learned Decision Stumps (LDS), MAX, and MAX when it issues a dispatch (MAXD) on Field Data, plus the coverage of data by MAX's dispatches (Cov)

Data	LDS	MAX	MAXD	Cov
Field Data	51	67	68	53
Cleaned Data	34	67	41	56
In vs Out Only	7	46	4	56

These results show that the learned knowledge is significantly better than MAX at predicting the dispatch corresponding to the resolution reported by the field technician. The only situation in which MAX performs better than the learned knowledge is on In vs Out Only, only considering the cases where MAX chose to dispatch. In this scenario, MAX is incorrect only 4% of the time. The learned decision stump is incorrect 7% of the time, but it chooses a dispatch for every case. Note that these were all cases where a technician reported a resolution that mapped unambiguously to a dispatch. Preliminary results show that if we use the learned stump to dispatch only when the confidence is high, the error rate scales gracefully with the coverage. For example, the learned knowledge can achieve a 4% error rate and cover 86% of the data. When the coverage of the learned knowledge decreases to 56%, it is incorrect only 2% of the time. The increase in performance using the learned vercode mapping over the MAX system is one piece of evidence supporting the conclusion that by looking at the data we can extract dispatch knowledge that can improve MAX's performance.

A second piece of evidence comes from a similar comparison of the dispatch error rate of the experts as compared to that of the learned vercode decision stump. Table 10 shows that the learned vercode mapping outperforms the experts on the field data.

Table 10: Comparison of error rates of Learned Decision Stumps (LDS), Expert (Exp), and Expert when he chose a dispatch (ExpD) on Field Data, plus the coverage of data by Exp's dispatches (Cov)

Data	LDS	Exp	ExpD	Cov
Field Data	51	67	61	50
Cleaned Data	34	69	42	53
In vs Out Only	7	53	8	51

A potential criticism of the above argument is that the learning is fitting error in the data, and the experts and MAX are actually better at dispatching. However, when MAX and the five experts are compared with each other, there is very little agreement. No expert agrees with MAX's dispatch more than 65% of the time. The average agreement is less than 50%. The agreement between any pair of experts is in the same range. These results suggest that the problem is much more difficult that previously thought, and the data may not be as full of errors as conventional wisdom would have you believe.

Further support for the contention that the learned knowledge is not just modeling errors in the data comes from a comparison of the effectiveness of the learned knowledge for dispatch in other areas. We took the best decision stump learned from one location's data and used it for dispatch in four other areas. As shown in Table 11, in three of the four comparisons, the knowledge learned in one area transfers well to the other areas.

Table 11: Comparison of error rates of knowledge learned from location X when applied to other locations.

Location	Field	Cleaned	In vs Out
X	52	34	7
A	54	25	5
B	57	38	7
C	56	21	3
D	64	51	18

It is important also to consider that error rate is not the only basis for comparison of dispatching knowledge. Different errors have different costs (*e.g.*, a very costly error would be to dispatch a cable technician when a retest would have revealed that there was no problem). Recent and current work is addressing learning knowledge for cost-effective local loop dispatch (Pazzani, *et al.* 1994; Provost 1994; Turney 1995).

6 CONCLUSIONS

The results presented here suggest that one of the following two conclusions is true: (i) by modeling the data we can find a vercode-to-dispatch mapping that can improve the performance of MAX, or (ii) machine learning is modeling systematic error in the data *and* the error is systematic with respect to the vercode.

Conclusion (i) is based on the apparent decrease in dispatch error rate of the learned vercode mapping over MAX. The learned vercode mapping also has a smaller error rate than that of the experts, when the experts are compared to the resolutions reported from the field.

Conclusion (ii) is based on the fact that if the error in the data were random with respect to the vercode, it would be virtually impossible to learn to fit the error.

We assert that conclusion (i) is more likely than conclusion (ii), because conclusion (ii) rules out most of the forms of error generally believed to exist in the data.

7 MACHINE LEARNING IN PRACTICE: LESSONS LEARNED

The first lesson learned from our efforts to apply machine learning in practice is that real data are unreliable. They can be filled with errors that reach far beyond the noise that machine learning work traditionally assumes. The errors in real data do not only exceed researchers' expectations in terms of their volume, but also in their quality. Researchers typically model noise as random perturbations of correct data. Errors in real data, while including random noise of this sort, may also include systematic errors that may occur for a variety of reasons ranging from miscalibration of measuring devices to management pressures to report particular results. These "non-traditional" sources of error make the results of learning correspondingly unreliable. If the learning techniques cannot model the data well, it is not clear whether it is due to the quality of the data, or it is due to the inadequacy of the description language. On the other hand, if the learning can model the data better than existing techniques, it is not clear that the increase in performance is not due to modeling some systematicity in the error.

We have investigated several approaches to dealing with poor data:

- **Obtain data from multiple sources.** In the case of local loop troubleshooting, we were able to obtain classifications from the existing expert system, from experts in the field of dispatch, and from technicians. The fact that machine learning is able to model the existing expert system supports the viability of machine learning in this domain. Using expert knowledge provides some additional evidence for this. It also, however, indicates the complexity of this domain, evidenced by the lack of agreement among experts.
- **Use domain knowledge to clean data.** We have used domain knowledge to "clean up" the data available for learning. In our case, "cleaning up" occurs by removing examples where classifications might be ambiguous or otherwise difficult to interpret. While this yields better learning results, it might be the case that we have simply eliminated the difficult parts of the domain from the data set (and therefore from the model).
- **Get what you can from the data.** Though we cannot claim a machine learning success story, we have unearthed interesting properties of the domain, and more specifically about the data. For instance, given the data available, decision stumps appear to beat full-blown learning and the existing system.

Overall, in this domain, we have not been able to claim machine learning as a successful solution to the problem, but there is promise. Though we cannot definitively say that a learned concept description is more accurate than MAX, our results indicate that we are likely to be able to increase MAX's performance. This domain also brings the issue of learning in the presence of systematic error to the fore, something that has lacked attention in the research community.

Acknowledgements

Kim Tabtiang performed the runs reported in this paper. Tom Fawcett helped in developing new mappings from field-technicians' resolutions to dispatches (for the cleaned-up and In vs. Out data), performed many of the initial c4.5 runs using the new mappings, and contributed to many lively discussions on learning and local loop dispatch.

References

Danyluk, A. P. & Provost, F. J. (1993a). Small Disjuncts in Action: Learning to Diagnose Errors in the Local Loop of the Telephone Network. In *Proceedings of the Tenth International Conference on Machine Learning*, 81-88. San Mateo, CA: Morgan Kaufmann Publishers, Inc.

Danyluk, A. P. & Provost, F. J. (1993b). Adaptive Expert Systems: Applying Machine Learning to NYNEX MAX, in Working Notes of the AAAI-93 Workshop on AI in Service and Support: Bridging the Gap Between Research and Applications.

Goodman, R. M. & Smyth, P. (1989) The Induction of Probabilistic Rule Sets - the ITRULE Algorithm. In *Proceedings of the Sixth International Workshop on Machine Learning*, 129-132. San Mateo, CA: Morgan Kaufmann Publishers, Inc.

Holte, R. C. (1993) Very Simple Classification Rules Perform Well on Most Commonly Used Datasets, Machine Learning:11:1, 63-90.

Merz, C. J., Pazzani, M., & Danyluk, A. P. (1994). Tuning Numeric Parameters of a Knowledge-Based System for Troubleshooting the Local Loop of the Telephone Network. Submitted to the IEEE Expert Special Track on Intelligent Telecommunication Systems.

Pazzani, M. J. & Brunk, C. (1993). Finding Accurate Frontiers: A Knowledge-Intensive Approach to Relational Learning. In *Proceedings of AAAI-93*, 328-334. Menlo Park, CA: AAAI Press.

Pazzani, M., Merz, C., Murphy, P., Ali, K., Hume, T., Brunk C. (1994). Reducing Misclassification Costs. In *Machine Learning: Proceedings of the Eleventh International Conference*, 217-225. San Mateo, CA: Morgan Kaufmann.

Provost, F. (1994). Goal-Directed Inductive Learning: Trading off Accuracy for Reduced Error Cost. In Working Notes of the AAAI-94 Workshop on Goal-Driven Learning, 94-101.

Quinlan, J. R. (1992). *C4.5: Programs for Machine Learning.* San Mateo, CA: Morgan Kaufmann Publishers, Inc.

Rabinowitz, H., Flamholz, J., Wolin, E., & Euchner, J. (1991) NYNEX MAX: A Telephone Trouble Screening Expert. In R. Smith & C. Scott (ed.), *Innovative Applications of AI 3*, 213-230. Menlo Park, CA: AAAI Press.

Turney, P. (1995). Personal Communication.