Feature Subset Selection for Rule Induction Using RIPPER

Jihoon Yang
Computer Science Dept.
Iowa State University
Ames, IA 50011
yang@cs.iastate.edu

Asok Tiyyagura
Computer Science Dept.
Iowa State University
Ames, IA 50011
asokt@cs.iastate.edu

Fajun Chen
Computer Science Dept.
Iowa State University
Ames, IA 50011
fjchen@cs.iastate.edu

Vasant Honavar
Computer Science Dept.
Iowa State University
Ames, IA 50011
honavar@cs.iastate.edu

CATEGORY: REAL WORLD APPLICATIONS

Abstract

The choice of features or attributes used to represent patterns in the synthesis of pattern classifiers using machine learning algorithms has a strong impact on the accuracy of the classifier, the number of examples needed to attain a given classification accuracy on test data, the cost of classification, and the comprehensibility of the learned classifier. This presents us with a feature subset selection problem, namely, the selection of a subset of features from a much larger candidate set of features to represent patterns to be classified so as to optimize multiple criteria such as the accuracy and the cost of pattern classification. Evolutionary algorithms, because of their ability to find good solutions offer a promising approach to such a multi-criteria optimization problem. Results of experiments reported in this paper demonstrate that feature subset selection using a genetic algorithm results in substantial improvement in classification accuracy and comprehensibility, and substantial reduction in the cost of classification associated with pattern classifiers trained using RIPPER on a number of benchmark problems. RIPPER is a relatively fast algorithm for induction of pattern classification rules from labeled examples. Given the ability of RIPPER to induce rules from large, noisy datasets consisting of patterns that are encoded using binary, numeric, or nominal attributes, this makes GARIPPER, the proposed hybrid approach to rule induction, an attractive approach to data-driven knowledge discovery.

1 Introduction

Many practical pattern classification tasks (e.g., medical diagnosis, power system applications) require learning of an appropriate classification function that assigns a given input pattern (typically represented using a vector of attribute or feature values) to one of a finite set of classes. It has been observed in the past that the choice of features, attributes, or measurements used to represent patterns that are presented to a classifier affect (among other things):

- The accuracy of the classification function that can be learned using an inductive learning algorithm (e.g., a decision tree induction algorithm or a neural network learning algorithm): The features used to describe the patterns implicitly define a pattern language. If the language is not expressive enough, it would fail to capture the information that is necessary for classification and hence regardless of the learning algorithm used, the accuracy of the classification function learned would be limited by this lack of information.
- The time needed for learning a sufficiently accurate classification function: For a given representation of the classification function, the features used to describe the patterns implicitly determine the search space that needs to be explored by the learning algorithm. An abundance of irrelevant features can unnecessarily increase the size of the search space, and hence the time needed for learning a sufficiently accurate classification function.
- The number of examples needed for learning a sufficiently accurate classification function: All other things being equal, the larger the number of features used to describe the patterns in a domain of interest, the larger is the number of examples needed to learn a classification function to a desired accuracy [Langley, 1995, Mitchell, 1997].

- The cost of performing classification using the learned classification function: In many practical applications e.g., medical diagnosis, patterns are described using observable symptoms as well as results of diagnostic tests. Different diagnostic tests might have different costs as well as risks associated with them. For instance, an invasive exploratory surgery can be much more expensive and risky than say, a blood test.

- The comprehensibility of the knowledge acquired through learning: A primary task of an inductive learning algorithm is to extract *knowledge* (e.g., in the form of classification rules) from the training data. Presence of a large number of features, especially if they are irrelevant or misleading, can make the knowledge difficult to comprehend by humans. Conversely, if the learned rules are based on a small number of relevant features, they would much more concise and hence easier to understand, and use by humans.

This presents us with a *feature subset selection problem* in automated design of pattern classifiers. The feature subset selection problem refers to the task of identifying and selecting a useful subset of features to be used to represent patterns from a larger set of often mutually redundant, possibly irrelevant features with different associated measurement costs and/or risks. An example of such a scenario which is of significant practical interest is the task of selecting a subset of clinical tests (each with different financial cost, diagnostic value, and associated risk) to be performed as part of a medical diagnosis task. Other examples of feature subset selection problem include large scale data mining applications, power system control [Zhou et al., 1997], construction of user interest profiles for text classification [Yang et al., 1998] and sensor subset selection in the design of autonomous robots [Balakrishnan and Honavar, 1996b].

The rest of the paper is organized as follows: Section 2 describes our approach that uses a Genetic Algorithm for RIPPER pattern classifiers. Section 3 explains the implementation details in our experiments. Section 4 presents the results of various experiments designed to evaluate the performance of our approach on some text classification tasks and on certain benchmark classification problems. Section 5 concludes with summary and discussion of some directions for future research.

2 Feature Selection Using a Genetic Algorithm for Rule Learning With RIPPER

2.1 Genetic Algorithms

Evolutionary algorithms [Goldberg, 1989, Holland, 1992, Koza, 1992, Fogel, 1995, Michalewicz, 1996, Mitchell, 1996, Banzaf et al., 1997] are randomized, population-based heuristic search techniques which include genetic algorithms, genetic programming, evolutionary programming, evolutionary strategies, and related approaches. They are inspired by processes that are modeled after biological evolution. Central to such evolutionary systems is the idea of a population of potential solutions (individuals) that corresponds to members of a high-dimensional search space.

The individuals represent candidate solutions to the optimization problem being solved. A wide range of genetic representations (e.g., bit vectors, `LISP` programs, matrices, etc.) can be used to encode the individuals depending on the space of solutions that needs to be searched. In genetic algorithms [Goldberg, 1989, Michalewicz, 1996, Mitchell, 1996], the individuals are typically represented by n-bit binary vectors. The resulting search space corresponds to an n-dimensional boolean space. In the feature subset selection problem, each individual would represent a feature subset.

It is assumed that the quality of each candidate solution (or fitness of the individual in the population) can be evaluated using a fitness function. In the feature subset selection problem, the fitness function would evaluate the selected features with respect to the criteria of interest (e.g., cost of the resulting classifier, classification accuracy of the classifier, etc.).

Evolutionary algorithms use some form of fitness-dependent probabilistic selection of individuals from the current population to produce individuals for the next generation. A variety of selection techniques have been explored in the literature. Some of the most common ones are *fitness-proportionate* selection, *rank-based* selection, and *tournament-based* selection [Goldberg, 1989, Michalewicz, 1996, Mitchell, 1996]. The selected individuals are transformed using genetic operators to obtain new individuals that constitute the next generation. The genetic operators are usually designed to exploit the known properties of the genetic representation, the search space, and the optimization problem to be solved. Genetic operators enable the algorithm to *explore* the space of candidate solutions. See [Balakrishnan and Honavar, 1995] for a discussion of some desirable properties of genetic representations and operators.

Mutation and *crossover* are two of the most commonly used operators that are used with genetic algorithms that represent individuals as binary strings. Mutation operates on a single string and generally changes a bit at random. Thus, a string 11010 may, as a consequence of random mutation, get changed to 11110. Crossover, on the other hand, operates on two parent strings to produce two offspring. With a randomly chosen crossover position 4, the two strings 01101 and 11000 yield the offspring 01100 and 11001 as a result of crossover. Other genetic representations (e.g., matrices, `LISP` programs) require the use of appropriately designed genetic operators [Michalewicz, 1996, Mitchell, 1996, Banzaf et al., 1997].

The process of selection and application of genetic operators to generate successive generations of individuals is repeated until a satisfactory solution is found (or the search fails to do so within the time allocated). It can be shown that evolutionary algorithms of the sort outlined above simulate highly opportunistic and exploitative randomized search that explores high-dimensional search spaces rather effectively under certain conditions [Holland, 1992]. In practice, the performance of evolutionary algorithms depends on a number of factors including: the choice of genetic representation and operators, the fitness function, the details of the selection procedure, and the various user-determined parameters such as population size, probability of application of different genetic operators, etc. The specific choices made in the experiments reported in this paper are summarized in Section 3.

2.2 RIPPER: An Effective Fast Algorithm for Rule Induction from Examples

Rule induction algorithms offer an attractive approach to data-driven knowledge discovery from labeled examples. Pattern classifiers that are induced by rule learning algorithms are often simpler and easier to comprehend by humans than those induced using genetic programming or most neural network approaches. Yet, results of experiments [Mooney et al., 1989] indicate that the classification accuracies of the classifiers induced using different approaches are often comparable.

A variety of algorithms for rule induction from labeled examples have been proposed in the literature. Some of them first construct a decision tree e.g., using C4.5 [Quinlan, 1993], and then extract a set of classification rules from the decision tree. Other algorithms (e.g., RIPPER [W.Cohen, 1995]) directly induce rules from the training data using a separate-and-conquer approach. The learned ruleset is post-processed to discard (as in C4.5) or prune (as in RIPPER) some of the rules using various criteria to improve their classification accuracy on test data.Recently, [Frank and Witten, 1998] has proposed an approach to rule learning that combines aspects of decision tree learning and the separate-and-conquer approach to eliminate the need for post-processing of the learned ruleset.

The design of RIPPER is an extension of IREP (Incremental Reduced Error Pruning) algorithm [Furnkranz and Widmer, 1994]. IREP tightly integrates reduced error pruning with separate-and-conquer strategy. IREP is a greedy rule induction algorithm which learns a rule at a time. The rule so learned covers a maximal subset of examples in its current training set. The rule is pruned so as to maximize a desired performance measure. All of the examples that are correctly labeled by the resulting rule are elminated from the training set. This process is repeated until a predertemined stopping condition is satisfied or the training set becomes empty. The IREP algorithm is shown in Figure 1 [W.Cohen, 1995].

```
procedure IREP(Pos,Neg)
begin
Ruleset: = Φ
while Pos≠Φ do
/* grow and prune a new rule */
split(Pos,Neg) into (GrowPos,GrowNeg)
and (PrunePos,PruneNeg)
Rule:=GrowRule(GrowPos,GrowNeg)
Rule:=PruneRule(Rule,PrunePos,PruneNeg)
/* stopping condition */
if the error rate of Rule on
(PrunePos,PruneNeg) exceeds 50% then
return Ruleset
else
add Rule to Ruleset
remove examples covered by Rule
from (Pos,Neg)
endif
endwhile
return Ruleset
end
```

Figure 1: The IREP algorithm

RIPPER [W.Cohen, 1995] is an improvement over IREP. It includes a better pruning and stopping criteria as well as post-processing of the rule set:

Among candidate sequences of conditions from the rule, IREP deletes conditions that maximize v(Rule,PrunePos,PruneNeg)$\equiv\frac{p+(N-n)}{P+N}$ where P and N represent the total number of examples in PrunePos and PruneNeg respectively and and p and n denote the number of examples in PrunePos and PruneNeg respectively that are correctly classified by the rule. RIPPER replaces v(Rule,PrunePos,PruneNeg) by v*(Rule,PrunePos,PruneNeg)$\equiv\frac{p-n}{p+n}$.

The stopping criterion used in RIPPER is based on the total description length of the rule set and the examples which is motivated by the minimum description length (MDL) heuristic [Rissanen, 1978]. For each rule in rule set, say R_i, A MDL heuristic is used in RIPPER decide whether to keep R_i, or a pruned version of R_i.

Experiments reported show that the classification accuracy of the rule set induced by RIPPER is comparable to that induced by C4.5. However, the time complexity of RIPPER is O($mlog^2m$) where as C4.5 has a time complexity of O(m^3) where m is the number of examples in the training set. This makes RIPPER especially attractive for large datasets. It is also worth noting that RIPPER is able to induce rules from examples that are encoded using boolean, numeric, or nominal attributes making it a useful technique for induction of pattern classifiers in a wide range of practical applications.

3 IMPLEMENTATION OF GARIPPER

The implementation of GARIPPER is shown in Figure 2.

1. Choose an initial population by generating the strings randomly.
2. Test each string to determine its effectiveness. In our case we used classification accuracy of the RIPPER algorithm as the fitness of the string.
3. Select best strings (based on fitness values) and apply crossover and mutation to generate new population.
4. Repeat from step 2 for a fixed number of generations or until the desired classification accuracy is met. Get the best string.

Figure 2.

We briefly describe in what follows, the design choices that were made for the implementation of GARIPPER (genetic approach to feature subset selection for RIPPER). Each individual in the population represents a candidate solution to the feature subset selection problem. Let m be the total number of features available to choose from to represent the patterns to be classified. In a medical diagnosis task, these would be observable symptoms and a set of possible diagnostic tests that can be performed on the patient. Each candidate feature subset is represented by a binary vector of dimension m. If a bit is a 1, it means that the corresponding feature is selected. A value of 0 indicates that the corresponding feature is not selected. The fitness of a candidate feature subset can be based on variour criteria such as: accuracy of the rule set learned by RIPPER (using the selected features), the measurement cost associated with the features that were selected, the computational cost of classification using the learned rule set, etc.

The current implementation of GARIPPER uses the accuracy of the learned ruleset (measured by ten-fold crossvalidation) as the fitness measure.

Our experiments with GARIPPER were run using a genetic algorithm used standard mutation and crossover operators and the tournament selection strategy [Goldberg, 1989, Mitchell, 1996]. The parameter settings used in the experiments were as follows:

- Population size: 50
- Number of generation: 20
- Probability of crossover: 0.6
- Probability of mutation: 0.001
- Population Replacement value: 99% (Best String is always retained)

The parameter settings were based on results of several preliminary runs. They are comparable to the typical values mentioned in the literature [Mitchell, 1996]. No attempt was made to to choose the best parameter

4 EXPERIMENTS

4.1 Description of Datasets

The experiments reported here used a wide range of real-world datasets from the machine learning data repository at the University of California at Irvine [Murphy and Aha, 1994] as well as a carefully constructed artificial dataset (3-bit parity) to explore the feasibility of using genetic algorithms for feature subset selection for neural network classifiers. The feature

Table 1: Datasets used in the experiments. *Size* is the number of patterns in the dataset, *Features* is the number of input features, and *Class* is the number of output classes.

Dataset	*Size*	*Features*	*Feature Type*	*Class*
ionosphere structure (**Ionosphere**)	351	34	numeric	2
DNA sequences (**Promoters**)	106	57	nominal	2
sonar classifiction (**Sonar**)	208	60	numeric	2
vehicle silhouettes (**Vehicle**)	846	18	numeric	4
house votes (**Votes**)	435	16	nominal	2
wine recognition (**Wine**)	178	13	numeric	3
zoo database (**Zoo**)	101	16	numeric, nominal	7
paper abstracts 1 (**Abstract1**)	100	790	numeric	2
paper abstracts 2 (**Abstract2**)	100	790	numeric	2
news articles 1 (**Reuters1**)	939	1568	numeric	6
news articles 2 (**Reuters2**)	139	435	numeric	4
news articles 3 (**Reuters3**)	834	1440	numeric	8

subset selection using RIPPER is also applied to document classification problem for journal paper abstracts and news articles.

4.1.1 Datasets from UCI Repository

In our experiments with real world datasets, our objective was to compare the classification accuracy of RIPPER using feature subsets selected by the genetic algorithm with those that use the entire set of features available. Table 1 summarizes the characteristics of the datasets. Some medical datasets include measurement costs for the features, but most of the datasets lack this information. Therefore, our experiments with the datasets from UCI repository focused on identifying a subset of features that yield high accuracy RIPPER classifiers. If measurement costs were available, the performance considering the cost in addition to the accuracy can also be included.

4.1.2 Document Datasets

The paper abstracts were chosen from three different sources: IEEE Expert magazine, Journal of Artificial Intelligence Research and Neural Computation. The news articles were obtained from Reuters dataset. Each document is represented in the form of a vector of numeric weights for each of the words (terms) in the vocabulary. The weights correspond to the term frequency and inverse document frequency (TFIDF) [Salton and McGill, 1983, Yang et al., 1998] values for the corresponding words. The training sets for paper abstracts were generated based on the classification of the corresponding documents into two classes (interesting and not interesting) by two different individuals, resulting in two different data sets (**Abstract1** and **Abstract2**). The classifications for news articles were given based on their topics (6, 4 and 8 classes) following [Koller and Sahami, 1997], resulting in three different datasets (**Reuters1**, **Reuters2** and **Reuters3**), respectively. These datasets are also summarized in Table 1. Since these datasets do not have measurement costs for the features, our experiments with document datasets also focused on identifying a minimal subset of features that yield high accuracy neural network classifiers.

4.2 Experimental Results

The experiments were designed to explore the effect of feature subset selection on the performance of RIPPER on a given choice of training and test sets. Each dataset was randomly partitioned into a training and test set (with 90% of the data used for training and the remaining 10% for testing). The genetic algorithm was used to select the best feature subset on the basis of this choice of training and test sets. The results were averaged over 5 independent runs of the genetic algorithm, for a given choice of training and test set. This process was repeated 10 times with 10 different choices of training and test set. The results of these experiments (which represent $5 \times 10 = 50$ runs of the genetic algorithm) are shown in Table 2. The entries in the tables give the means (and standard deviations) in the form *mean* ($\pm$ *standard deviation*).

4.2.1 Improvement in Generalization using Feature Subset Selection

To study the effect of feature subset selection on generalization, experiments were run using classification accuracy as the fitness function. The results in Table 2 indicate that the results obtained using GA-selected subset of features compare quite favorably with RIP-

Table 2: Comparison of rule induction pattern classifiers constructed by RIPPER using the entire set of features with the best classifier constructed by GARIPPER using fitness estimates based on 10-fold cross-validation. GARIPPER (best) represents the accuracy of the best classifier produced by GARIPPER using 10-fold crossvalidation among the 5 independent runs of the genetic algorithm. For Reuters 1 and Reuters 3 Datasets we divided them into trainset(60%) and testset(40%) and did the GARIPPER test for 5 independent runs. We skipped 10 fold crossvalidation part as it was taking long time. So the total number of GA runs were just 5.GARIPPER (average) represents the mean and the standard deviation (computed over 5 independent runs of the genetic algorithm) of the accuracy of the best classifier produced by GARIPPER. See Section 4.2 for details.

	RIPPER		GARIPPER (average)		GARIPPER (best)	
Dataset	*Features*	*Accuracy*	*Features*	*Accuracy*	*Features*	*Accuracy*
Ionosphere	34	88.91 ± 1.41	14.6 ± 4.16	94.1 ± 0.71	10	94.61
Promoters	57	84.75 ± 5.01	28 ± 3.54	92.8 ± 1.43	28	94.75
Sonar	60	76.86 ± 2.49	30.4 ± 2.70	81.24 ± 2.37	27	84.3
Vehicle	18	67.25 ± 1.73	9.4 ± 1.34	73.24 ± 0.18	10	73.42
Votes	16	94.0 ± 1.02	4.8 ± 1.30	96.1 ± 0.3	3	96.67
Wine	13	91.95 ± 1.93	7.4 ± 0.55	96.95 ± 0.52	7	97.48
Zoo	16	89.00 ± 4.29	8.5 ± 1.97	95.2 ± 0.84	7	96.0
Abstract1	790	84.0 ± 2.81	385.8 ± 17.43	87.0 ± 0.0	371	87.0
Abstract2	790	86.00 ± 2.81	402.4 ± 6.88	88.0 ± 0.0	395	88.0
Reuters1	1568	92.77 ± 0.98	787 ± 23.54	97.4 ± 0.94	753	98.09
Reuters2	435	81.80 ± 1.8	214.6 ± 7.44	92.24 ± 0.55	203	92.49
Reuters3	1440	95.92 ± 0.76	719.2 ± 19.48	97.99 ± 0.17	712	98.12

PER that use all of the features in all randomly partitioned datasets. In particular, feature subset selection resulted in substantial improvement in generalization on almost all of the datasets. Also, the number of features selected is significantly smaller than the total number of features present in the original data representation in all of the datasets.

The results shown in Table 2 indicate that the results obtained using GA-selected subset of features are better than the RIPPER that use all of the features in almost all of the datasets with 10-fold cross-validation.The best individual generated by GARIPPER outperformed RIPPER in almost all datasets. The number of features selected is significantly smaller than the total number of features present in the original data representation in all of the datasets; Again the learned rule sets were observed to be very simple and easy to comprehend for a human user.

5 Summary and Discussion

An approach to feature subset selection using a genetic algorithm for RIPPER classifiers is proposed in this paper. A fast inductive rule learning algorithm, RIPPER, is employed to evaluate the fitness (in terms of the generalization accuracy) of candidate feature subsets in the genetic algorithm. The results presented in this paper indicate that genetic algorithms offer an attractive approach to solving the feature subset selection problem in inductive learning of pattern classifiers in general, and RIPPER pattern classifiers in particular as it is very fast in learning the rule sets.

The GA-based approach to feature subset selection does not rely on monotonicity assumptions that are used in traditional approaches to feature selection which often limits their applicability to real-world classification and knowledge acquisition tasks. It also offers a natural approach to feature subset selection by taking into account, the distribution of available data. This is due to the fact that feature selection is driven by estimated fitness values, which if based on multiple partitions of the dataset into training and test data, provide a robust measure of performance of the feature subset. This is not generally the case with many of the greedy stepwise algorithms that select features based on a single partition of the data into training and test sets. Consequently, the feature subsets selected by such algorithms are likely to perform rather poorly on other random partitions of the data into training and test sets.

The approach to feature subset selection can be extended to incorporate multiple criteria (e.g., accuracy, cost) into the feature selection process. This finds applications in cost-sensitive design of classifiers for tasks such as medical diagnosis, computer vision, among others. Another interesting application is automated data mining and knowledge discovery from datasets with an abundance of irrelevant or redundant features. In such cases, identifying a relevant subset that adequately captures the regularities in the data can be particularly useful, particularly in scientific knowledge discovery tasks. Techniques similar to the one discussed in this paper have been successfully used recently to select feature subsets for pattern classification tasks that arise in power system security assessment [Zhou et al., 1997], sensor subsets in the design of behavior and control structures for autonomous mobile robots [Balakrishnan and Honavar, 1996a, Balakrishnan and Honavar, 1996b, Balakrishnan and Honavar, 1996c], intrusion detection[Helmer Guy, 1999].

Additional experiments with GARIPPER in scientific knowledge discovery tasks in bioinformatics (e.g., discovery of protein structure–function relationships, carcinogenicity prediction, gene sequence identification) are currently in progress. Some directions for future research include: Extension of feature subset selection by incorporating *feature construction* and *genetic programming* [Koza, 1992]; Extensive experimental (and wherever feasible, theoretical) comparison of the performance of the proposed approach with that of conventional methods for feature subset selection; More principled design of multi-objective fitness functions for feature subset selection using domain knowledge as well as mathematically well-founded tools of multi-attribute utility theory; [Keeney and Raiffa, 1976];find novel genetic operators which would improve the performance of GARIPPER approach.

Acknowledgements

This Research is partially supported by NSF-IRI9409580,ISU Council on international programs and EPRI,ISU Graduate College.

References

[Balakrishnan and Honavar, 1995] Balakrishnan, K. and Honavar, V. (1995). Properties of genetic representations of neural architectures. In *Proceedings of WCNN'95 July 17-21 Washington D.C.*, volume 1, pages 807–813.

[Balakrishnan and Honavar, 1996a] Balakrishnan, K. and Honavar, V. (1996a). Analysis of neurocontrollers designed by simulated evolution. In *Proceedings of the International Conference on Neural Networks*, Washington, D.C.

[Balakrishnan and Honavar, 1996b] Balakrishnan, K. and Honavar, V. (1996b). On sensor evolution in robotics. In Koza, Goldberg, Fogel, and Riolo, editors, *Proceedings of the 1996 Genetic Programming Conference – GP-96*, pages 455–460. MIT Press, Cambridge, MA.

[Balakrishnan and Honavar, 1996c] Balakrishnan, K. and Honavar, V. (1996c). Some experiments in the evolutionary synthesis of robotic neurocontrollers. In *Proceedings of the World Congress on Neural Networks (WCNN'96)*, pages 1035–1040, San Diego, CA.

[Banzaf et al., 1997] Banzaf, W., Nordin, P., Keller, R., and Francone, F. (1997). *Genetic Programming - An Introduction.* Morgan Kaufmann, Palo Alto, CA.

[Fogel, 1995] Fogel, D. (1995). *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence.* IEEE Press, Piscataway, NJ.

[Frank and Witten, 1998] Frank, E. and Witten, I. H. (1998). Generating accurate rule sets without global optimization.

[Furnkranz and Widmer, 1994] Furnkranz and Widmer (1994). Incremental reduced error pruning. In *Proceedings of the Eleventh Annual Conference.* Morgan Kaufmann.

[Goldberg, 1989] Goldberg, D. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley, New York.

[Helmer Guy, 1999] Helmer Guy, Wong Johnny S, H. V. (1999). Automated discovery of concise predictive rules for intrusion detection. In *Proceedings of AAAI'99.*

[Holland, 1992] Holland, J. (1992). *Adaptation in Natural and Artificial Systems.* MIT Press, Cambridge, MA.

[Keeney and Raiffa, 1976] Keeney, R. and Raiffa, H. (1976). *Decisions with Multiple Objectives: Preferences and Value Tradeoffs.* Wiley, New York.

[Koller and Sahami, 1997] Koller, D. and Sahami, M. (1997). Hierarchically classifying documents using very few words. In *International Conference on Machine Learning*, pages 170–178.

[Koza, 1992] Koza, J. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* MIT Press, Cambridge, MA.

[Langley, 1995] Langley, P. (1995). *Elements of Machine Learning.* Morgan Kaufmann, Palo Alto, CA.

[Michalewicz, 1996] Michalewicz, Z. (1996). *Genetic Algorithms + Data Structures = Evolution Programs.* Springer-Verlag, New York, third edition.

[Mitchell, 1996] Mitchell, M. (1996). *An Introduction to Genetic algorithms.* MIT Press, Cambridge, MA.

[Mitchell, 1997] Mitchell, T. (1997). *Machine Learning.* McGraw Hill, New York.

[Mooney et al., 1989] Mooney, R., Shavlik, J., Towell, G., and Gove, A. (1989). An experimental comparison of symbolic and connectionist learning algorithms. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 775–780. Morgan Kauffman.

[Murphy and Aha, 1994] Murphy, P. and Aha, D. (1994). Repository of machine learning databases. Department of Information and Computer Science, University of California, Irvine, CA.

[Quinlan, 1993] Quinlan, R. (1993). *C4.5: Programs for Machine Learning.* Morgan Kaufmann, San Mateo, CA.

[Rissanen, 1978] Rissanen, J. (1978). Modelling by shortest data description. *Automatica*, 14:465–471.

[Salton and McGill, 1983] Salton, G. and McGill, M. (1983). *Introduction to Modern Information Retrieval.* McGraw Hill, New York.

[W.Cohen, 1995] W.Cohen, W. (1995). Fast effective rule induction. San Fransisco. Morgan Kaufmann.

[Yang et al., 1998] Yang, J., Pai, P., Honavar, V., and Miller, L. (1998). Mobile intelligent agents for document classification and retrieval: A machine learning approach. In *14th European Meeting on Cybernetics and Systems Research. Symposium on Agent Theory to Agent Implementation*, Vienna, Austria.

[Zhou et al., 1997] Zhou, G., McCalley, J., and Honavar, V. (1997). Power system security margin prediction using radial basis function networks. In *Proceedings of the 29th Annual North American Power Symposium*, Laramie, Wyoming.