Integrating Models of Discrimination and Characterization for Learning from Examples in Open Domains

Paul Davidsson
Department of Computer Science, University of Karlskrona/Ronneby, S-372 25 Ronneby, Sweden
paul.davidsson@ide.hk-r.se http://www.ide.hk-r.se/~pdv

Abstract

It is argued that in applications of concept learning from examples where not every possible category of the domain is present in the training set (i.e., most real world applications), classification performance can be improved by integrating suitable discriminative and characteristic classification schemes. The suggested approach is to first discriminate between the categories present in the training set and then characterize each of these categories against all possible categories. To show the viability of this approach, a number of different discriminators and characterizers are integrated and tested. In particular, a novel characterization method that makes use of the information about the statistical distribution of feature values that can be extracted from the training examples is used. The experimental results strongly supports the thesis of the paper.

1 Introduction

Most algorithms that learn from examples (e.g., decision tree algorithms such as ID3 [Quinlan, 1986] and C4.5 [Quinlan, 1993], nearest neighbor algorithms such as the IB family [Aha *et* al., 1991] and the backpropagation algorithm [Rumelhart *et* al., 1986]), learn *discriminative* category descriptions. That is, they learn to discriminate between the categories present in the training set. As a consequence, the classification schemes computed by these algorithms suffer from an inability of detecting instances of categories not present in the set of training examples. Instead, such instances are assigned to one of the categories actually represented in the training set, resulting in undesired misclassifications. Although this problem often is ignored in machine learning research, it arises in most real world applications as these typically can be characterized as *open domains* [Hutchinson, 1994]. We simply do not have complete information about the domain, e.g., the actual number of categories is not known.

For example, consider the decision mechanism in a coin-sorting machine of the kind often used in bank offices. Its task is to sort and count a limited number of different coins (e.g., a particular country's), and to reject all other coins. Supposing that this decision mechanism is to be learned, it is for practical reasons impossible to train the learning system on every possible kind of coin, genuine or faked. Rather, the system should be trained only on the kinds of coins it is supposed to accept. Another example are decision support systems, for instance in medical diagnosis, where the cost of a misclassification often is very high — it is better to remain silent than to give an incorrect diagnosis.

As pointed out by Smyth and Mellstrom [1992], it is necessary to learn *characteristic* descriptions to solve the problem. Such a description characterizes the category with regard to all possible categories (regardless of their occurrence in the training set), and are thus, at least potentially, able to reject instances of categories not present in the training set. We will here call algorithms that learn characteristic descriptions *characterizers* and algorithms that learn discriminative descriptions *discriminators.* Holte et al. [1989] have shown that the CN2 algorithm can be modified to learn characteristic descriptions in the form of rule-based *maximum specific descriptions.* Besides simple memorization of the training instances (without doing any generalization), constructing maximum specific description is the simplest way to learn characteristic descriptions; for each category (or disjunct) one just compute the minimum and maximum value of each feature from the training instances belonging to the category and then construct a description that only accepts instances within the multidimensional space delimited by these values. (For nominal features, allow only the values present in the training instances belonging to the category.)

It is possible make distinctions between different kinds of characterizers in terms of how much they generalize. On one end we have those that learn the most general descriptions that do not cover any description of other concepts (e.g., AQ11 [Michalski and Larson, 1978]) which in some cases degenerates into discriminators, and on the other end we have those that, just as the above mentioned modification of CN2, learns the most specific descriptions possible. However, in most applications both these extreme approaches are inadequate as they tend to over- and under-generalize respectively. Moreover, these approaches are very static in the sense that there is no way of controlling the degree of generalization. It is the author's belief that the ability to control the degree of generalization is essential in most real world applications. There is a trade-off between the number of rejected and misclassified instances that must be balanced in accordance with the constraints of the application. In some applications the cost of a misclassification is very high and rejections are desirable in uncertain cases, whereas in others the number of rejected instances belonging to categories present in the training set are to be kept low and a higher number of misclassifications are accepted. Of course, it is always desirable to reject as many as possible of the instances of categories not present in the training set.

Another disadvantage of current characterizes is that they are not as good as current discriminators to discriminate between the categories actually present in the training set. The reason is simply that often they do not take into account the information that can be extracted from the training data about those categories (or do so only to a certain extent, cf. AQ11).

2 Integrating discrimination and characterization

The weaknesses of algorithms learning discriminative descriptions correspond closely to the strengths of those that learn characteristic descriptions and vice versa. That is, discriminative descriptions are in general good at discriminating between categories present in the training set but are unable to identify instances of categories not present in the training set whereas characteristic descriptions are able to do this but are not as good to discriminate between the categories present in the training set. As a consequence, it would be useful to try to combine their strength and at the same time reduce their weaknesses. A general method for doing this is to first apply an algorithm that discriminates between the categories present in the training set, and then another one that characterizes these categories separately against all possible categories.[1] Thus, the learning is divided into two separate phases:

1. discrimination between all known categories
2. characterization of these categories against all possible categories.

[1] There are obvious reason for doing it in this order. For instance, we do not know which entities (i.e., disjuncts) to characterize before we have made the discrimination. Also the classification would be less computationally efficient if we first test all the characteristic description and then discriminate between those that accepted the instance.

One instantiation of this general method would be to apply a discriminator (e.g., ID3) in the first phase and then compute the maximum specific description for each category/disjunct (i.e., for each leaf, in the ID3 case). Classification, on the other hand, would consist of first applying the discriminative description to get a preliminary classification, and then apply the maximum specific description to decide whether to reject or accept the instance. In what follows we will refer to the multidimensional subspace of the instance space defined by a characteristic description, in this case the maximum specific description, as the *acceptance region.* Thus, each category/disjunct has its own acceptance region. If the instance to be classified is outside the acceptance region, it will be rejected whereas if it is inside the acceptance region, it will be classified according to the preliminary classification made by the discriminator.

Now, which discriminators and characterizers are suitable for integration? As indicated above, it is possible to use existing components, e.g., ID3 as discriminator and compute maximum specific descriptions for characterization. Since most research on learning from examples has focused on creating discriminative descriptions, many good discriminators besides ID3 have been invented. The problem is with the characterization phase. As pointed out earlier, the maximum specific description method is rather static in the sense that there is no way of controlling the degree of generalization, i.e., the size of the acceptance regions. Moreover, the method is very sensitive to noise. In the next section an example of a more noise-tolerant approach to characterization in which it is possible to control the degree of generalization will be described.

3 A novel approach to characterization

The central idea of this method is to make use of statistical information concerning the distribution of the feature values hidden in the training data. Two versions of this approach, below referred to as the SD approach, have been developed: one *univariate* that computes separate and explicit limits for each feature (just as methods based on maximum specific descriptions do) which is suitable for application to, e.g., decision tree algorithms,[2] and one *multivariate* method, able to capture covariation among two or more features, suitable for integration with discriminators that do not use rule- or tree-based representation, e.g., nearest neighbor.

3.1 Univariate version

For every feature (and every category/disjunct) we compute a lower and an upper limit so that the estimated probability that a particular feature value (of an instance

[2]The explicit limits make it possible to retain the tree structure induced by the discrimination algorithm (e.g., ID3) augmenting it with a subtree at each leaf that characterize the category/disjunct represented by that leaf.

belonging to this category/disjunct) belongs to the interval between these limits is $1 - \alpha$. In this way we can control the degree of generalization and, consequently, the above mentioned trade-off by choosing an appropriate α-value. The lesser the α-value is, the more misclassified and less rejected instances. Thus, if it is important not to misclassify instances and a high number of rejected instances are acceptable, a high α-value should be selected.

It turns out that only very simple statistical methods are needed to compute such limits. Assuming that X is a normally distributed stochastic variable, we have that:

$$P(m - \lambda_{\frac{\alpha}{2}}\sigma < x < m + \lambda_{\frac{\alpha}{2}}\sigma) = 1 - \alpha$$

where m is the mean, a is the standard deviation, and λ is a critical value depending on α (e.g., $\lambda_{0.025} = 1.96$).

In order to follow this line of argument we have to assume that the feature values of each category (or each disjunct if it is a disjunctive concept) are normally distributed. As indicated by the experiments below, this assumption seems not too strong for most applications. Moreover, as we cannot assume that the actual values of m and σ are known, they have to be estimated. A simple way of doing this is to compute the mean and the standard deviation of the training instances belonging to the category/disjunct.[3]

3.2 Multivariate version

To implement this method we will make use of multivariate methods to compute a weighted distance from the instance to be classified to the "center" of the category/disjunct. If this distance is larger than a critical value (dependent of α) the instance is rejected. Assuming that feature values are normally distributed within categories/disjuncts we have that the solid ellipsoid of x values satisfying

$$(x - \mu)^T \Sigma^{-1} (x - \mu) \leq \chi^2_p(\alpha)$$

has probability $1 - \alpha$, where μ is the mean vector, Σ is the covariance matrix, x^2 *is the* chi-square distribution, and p is the number features. (For more details, see for instance [Johnson and Wichern, 1992].) Thus, we have to estimate two parameters for each category/disjunct: (i) μ, which contains the mean for each feature, and (ii) Σ, which represents the covariance for each pair of features. In analogy with the univariate case, we estimate these parameters by computing the observed mean vector and covariance matrix of the training instances belonging to the category/disjunct.

The main limitation of both versions of the SD approach is that they are only applicable to numerical features. However, as the Max approach is applicable also to ordered and nominal features, it is possible to make SD more general by combining it with Max to form a hybrid approach able to handle all kinds of ordered features. We would then use SD for numerical features and Max for the rest of the features.

[3]It can be argued that this is a rather crude way of computing the limits. A more elaborate approach would be to compute confidence intervals for the limits and use these instead. This was actually the initial idea but it turned out that this only complicates the algorithm and does not increase the classification performance significantly.

4 Empirical results

The ideas behind the SD approach emerged when working on a real world application concerning the problem of learning the decision mechanism in coin sorting machines described in the introduction. The application of the SD method to this problem was very successful and the results are presented briefly below. A more detailed presentation of this application can be found in [Davidsson, 1996b].

In order to properly evaluate the SD approach it have to be applied to other data sets as well. At the UCI Repository of Machine Learning databases there are no examples of data sets of the desired kind, i.e., data sets consisting of separate training set (with x categories) and test set (with more than x categories). However, we can simulate such data sets in the following way: Take any data set, but leave out one (or more) category during training. Then test the algorithm on instances from all of the categories. At best, the algorithm will classify all the instances belonging to categories present in the training set correctly and reject all those belonging to categories not present in the training set. This approach may at first sight seem somewhat strange as we actually know that there are, for instance, three categories of Irises in the famous Iris data set [Anderson, 1935]. But how can we be sure that there only exist three categories? It might exist some not yet discovered species of Iris. In fact, as pointed out earlier, in most real world applications it is not reasonable to assume that all relevant categories are known in advance and can be represented in the training set.

Due to shortage of space, only the results of one experiment besides the coin sorting application will be presented here. Although better classification performance was achieved using other data sets, I have chosen the Iris data set because it is so well-known. (As discrimination between the three categories is considered relatively easy, it was somewhat surprising that this task turned out to be quite difficult.) Results from several different data sets using different discrimination algorithms can be found in [Davidsson, 1996a]. We will begin with studying the behavior of the SD approach and then compare it with other approaches to learning characteristic descriptions.

4.1 The behavior of the SD approach

In our experiments we have used re-implementations of two different discriminators, ID3 and IBI (a basic nearest neighbor algorithm [Aha *et* al., 1991]), and combined them with both the univariate and the multivariate version of SD. We will here only describe the results of

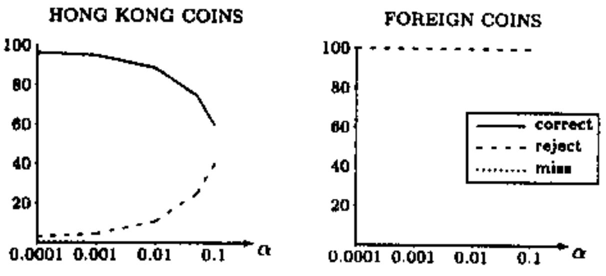

Figure 1: Classification performance of ID3-SDuni as a function of α. Performance on categories present in the training set are shown in the left diagram and the category not present in the training set in the right diagram. (For each α-value, averages in percentages over 10 runs.)

ID3 integrated with the univariate version (ID3-SDuni) and IB1 integrated with the multivariate version (IB1-SDmulti), which are the most natural combinations.

Coin sorting application

In the experiments two databases were used, one describing Canadian coins contains 7 categories (1, 5, 10, 25, 50 cent, 1 and 2 dollar), and one describing Hong Kong coins that also contains 7 categories (5, 10, 20, 50 cent, 1, 2, and 5 dollar). All of the 5 attributes (diameter, thickness, permeability, and two kinds of conductivity) are numerical. The Canada and Hong Kong databases were chosen because when using the manufacturer's current method for creating the rules of the decision mechanism (which is manual to a large extent), these coins have been causing problems. In each experiment 140 (7x20) instances were randomly chosen for training and 700 (2x7x50) for testing.

Figure 1 shows the classification results of ID3-SDuni for some different α-value when training only on Hong Kong coins (which is the most difficult case). To begin with, we can see that all foreign coins (i.e., the Canadian coins) are rejected. However, there were some problems with misclassifications of Hong Kong coins. In this particular application there are some demands that must be met: at most 5% rejects of known types of coins and very few misclassifications (not more than 0.5%). For ID3-SDuni, these requirements are met when α is between 0.001 and 0.0001. This clearly illustrates the advantage of being able to control the degree of generalization.

The results when applying IBl-SDmulti are even more pleasing (see Figure 2). In fact, it does not misclassify any instances! Note, however, that very small a-values must be used to achieve excellent classification behavior.

The Iris database

The Iris database contains 3 types of Iris plants (Setosa, Versicolor or Virginica) of 50 instances each. In each experiment the data set was randomly divided in half, with one set used for training and the other for testing. Thus, 50 (2x25) instances were used for training and 75

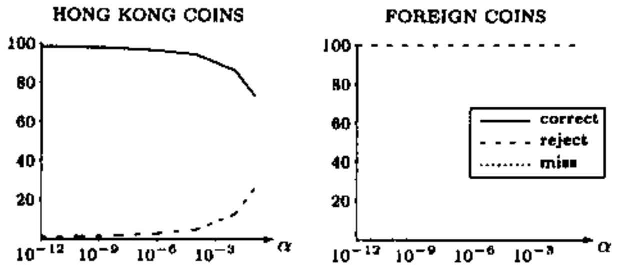

Figure 2: Classification performance of IBl-SDmulti.

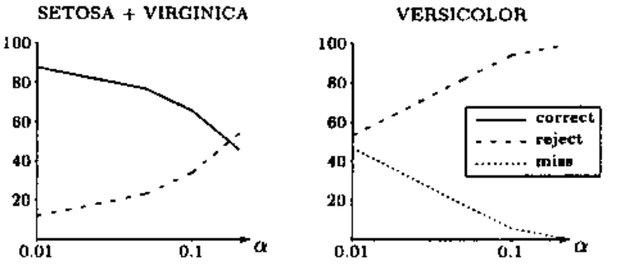

Figure 3: Classification performance of ID3-SDuni.

(3x25) for testing. Figure 3 shows the classification results when the algorithms were trained on instances of Iris Setosa and Iris Virginica (the most difficult case). We see that by varying the α-value it is possible to control the trade-off between the number of rejected and misclassified instances. It is possible to achieve almost zero misclassifications if we choose $\alpha = 0.2$, but then we get a rejection rate of over 50% also for the two known categories.

Figure 4 shows some more encouraging results for IBl-SDmulti. Compared to ID3-SDuni we see that by selecting an appropriate α-value it is always possible to achieve both more correct classifications and less misclassifications using the multivariate approach. Since SD-multi creates acceptance regions that closer match the distribution of feature values, i.e., regions that are smaller but still cover as many instances, fewer instances of categories not present in the training set are misclassified.

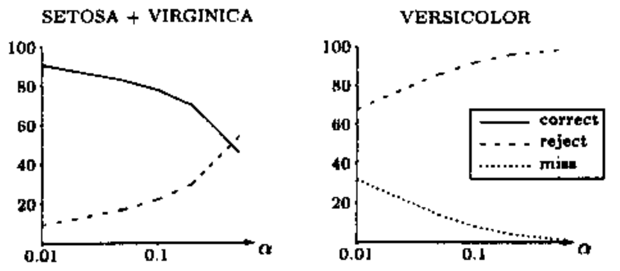

Figure 4: Classification performance of IBl-SD-multi.

	Hong Kong coins			Foreign coins		
	corr.	miss	reject	corr.	miss	reject
ID3	98.3	1.7	0.0	0.0	100.0	0.0
ID3-Max	79.7	0.0	20.3	0.0	0.0	100.0
ID3-SDuni 0.0001	96.3	0.5	3.2	0.0	0.0	100.0
IB1	99.3	0.7	0.0	0.0	100.0	0.0
IB1-SDmulti 10^{-12}	98.7	0.0	1.3	0.0	0.0	100.0
AQ15c	86.1	0.0	13.9	0.0	0.0	100.0
AQ15c conf 0.8	96.9	1.6	1.5	0.0	3.9	96.1
AQ15c conf 0.6	97.7	2.3	0.0	0.0	36.5	63.5

Table 1: Results from training set containing Hong Kong coins (averages in percentages over 10 runs).

4.2 Comparison to other algorithms

We have compared six different algorithms:

- the basic ID3 algorithm (only discriminator)
- the basic IB1 algorithm (only discriminator)
- ID3 combined with the maximum specific description algorithm (ID3-Max)
- ID3 combined with the univariate version of SD
- IB1 combined with the multivariate version of SD
- AQ15c

AQ15c[4] [Wnek *et* al., 1995] is a system that learns decision rules from examples (and counterexamples). It has the ability to learn discriminative or characteristic descriptions depending on how parameters are set. It is by many considered as one of the best algorithms for learning characteristic descriptions. To get AQ15c to learn characteristic descriptions a parameter called *trim* is set to *spec,* which means that the rules learned "are as specific as possible, involving the maximum number of extended selectors, each with a minimum of values" [Wnek *et* al., 1995, page 11],

When using the plain AQ15c algorithm it is not possible to control the degree of generalization. However, for each classified instance AQ15c also outputs a value between 0 and 1 reflecting how confident it is in the classification. Thus, the decision to accept or reject an instance could be based on the degree of confidence, e.g., accept if confidence > 0.8 and reject if confidence < 0.8.

In the following experiments two (or more) results of AQ15c are presented: (i) the plain AQI5c and (ii) the version(s) using the confidence which seems to balance the trade-off best (i.e., we try to present AQ15c as favorable as possible). Also, one or two a-values for each of the SD algorithms that performs "better" than (i) and (ii) respectively has been chosen.

Coin sorting application

The results from the coin sorting application are presented in table 1. First, we see that of the discriminators, IB1 is slightly better than ID3. As a result, the combined

[4]AQ15c is a C language re-implementation of AQ15.

	Setosa + Virginica			Versicolor		
	corr.	miss	reject	corr.	miss	reject
ID3	99.0	1.0	0.0	0.0	100.0	0.0
ID3-Max	71.4	0.0	28.6	0.0	14.8	85.2
ID3-SDuni 0.1	66.4	0.0	33.6	0.0	5.6	94.4
ID3-SDuni 0.01	87.6	0.0	12.4	0.0	47.2	52.8
IB1	100.0	0.0	0.0	0.0	100.0	0.0
IB1-SDmulti 0.05	83.4	0.0	16.6	0.0	14.1	85.9
IB1-SDmulti 0.0001	99.0	0.0	1.0	0.0	75.2	24.8
AQ15c	81.2	0.0	18.8	0.0	70.8	29.2
AQ15c conf 0.75	97.8	0.0	2.2	0.0	98.4	1.6

Table 2: Results from training set containing instances of Iris Setosa and Iris Virginica (averages over 40 runs).

algorithms using IB1 performs better than those using ID3. Both discriminators misclassify, of course, all foreign coins. The plain AQ15c algorithm suffers from the same problem as ID3-Max, it rejects far too many instances of the categories present in the training set (the Hong Kong coins). However, when decreasing the confidence factor the number of misclassification becomes much too high. In short, the SD algorithms are clearly superior to both ID3-Max and AQ15c for this problem.

The only confidence values (except from 1.00 that corresponds to plain AQ15c) produced by AQ15c were 0.8, 0.6, 0.4, 0.2 and 0.0. Therefore, these are the only relevant candidates. This is another disadvantage with AQ15c — you cannot control the generalization completely, you are limited by the confidence values it produce and cannot choose a value in between. Using SD on the other hand, you are able to choose any degree of generalization you want.

Note also that the number of misclassifications for categories present in the training set is reduced when the discriminator is combined with a characterizer.

The Iris database

Table 2 shows the classification results when the algorithms were trained on instances of Iris Setosa and Iris Virginica. Also in this experiment does IB1 discriminate slightly better than ID3. We see that IB1-SDmulti 0.0001 outperforms ID3-Max in every respect and that AQ15c is not adequate for this task — it misclassifies far too many of the class not present in the training set. It is impossible to achieve less than 70% misclassifications. The only confidence values produced by AQ15c were 1.00, 0.75, 0.50, 0.25 and 0.00.

5 How to choose discriminator and characterizer

In theory it is possible to combine an arbitrary discriminator with an arbitrary characterization algorithm. In practice, however, it is often necessary to take some constraints into consideration, such as, the representation language of the category descriptions. For instance, if you need to retain the tree structure constructed by a decision tree algorithm, you are forced to chose SDuni or Max for characterization rather than SDmulti.

Another issue to take into consideration is whether the categories in the domain are likely to be of a disjunctive nature or not. If a category consists of clearly separate clusters of instances and we use a discriminator that does not learn explicitly disjunctive descriptions, e.g., backpropagation and nearest neighbor algorithms, only one acceptance region is computed for the category. This acceptance region will then be unnecessarily large implying too many misclassified instances.

6 Conclusions and future work

We have argued that for most real world applications of learning from examples classification performance can be improved if discriminative and characteristic classification schemes are integrated. The former is used to discriminate between the categories present in the training set, and the latter to characterize these categories against all possible categories. A novel approach for characterization, the SD approach, was suggested. An important property of this approach is its ability to control the degree of generalization continuously which is crucial for most real world applications. Finally, some experimental results were presented that supported the claims that classification performance can be improved by integrating discriminative and characteristic classification schemes and that the SD approach often is a good choice when selecting a characterizer.

Some important issues for future work are: (i) developing strategies for selection of discriminators and characterizers, (ii) developing characterizers in which it is possible to control the degree generalization also for non-numeric features, and (iii) trying to shed some light on the underlying tension between discrimination and characterization.

7 Acknowledgment

I wish to thank R.S. Michalski and E. Bloedorn at the MLIC for providing the AQ15c software.

References

[Aha *et al.,* 1991] D.W. Aha, D. Kibler, and M.K. Albert. Instance-based learning algorithms. *Machine Learning,* 6(1):37-66, 1991.

[Anderson, 1935] E. Anderson. The Irises of the Gaspe Peninsula. *Bulletin of the American Iris Society,* 59:2-5, 1935.

[Davidsson, 1996a] P. Davidsson. *Autonomous Agents and the Concept of Concepts.* PhD thesis, Department of Computer Science, Lund University, Sweden, 1996.

[Davidsson, 1996b] P. Davidsson. Coin classification using a novel technique for learning characteristic decision trees by controlling the degree of generalization. In *Ninth International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems (IEA/AIE-96),* pages 403-412. Gordon and Breach Science Publishers, 1996.

[Holte *et al,* 1989] R.C. Holte, L.E. Acker, and B.W. Porter. Concept learning and the problem of small disjuncts. In *IJCAI-89,* pages 813-818. Morgan Kaufmann, 1989.

[Hutchinson, 1994] A. Hutchinson. *Algorithmic Learning.* Clarendon Press, 1994.

[Johnson and Wichern, 1992] R.A. Johnson and D.W. Wichern. *Applied Multivariate Statistical Analysis.* Prentice-Hall, 1992.

[Michalski and Larson, 1978] R.S. Michalski and J.B. Larson. Selection of most representative training examples and incremental generation of VL hypotheses: The underlying methodology and the description of programs ESEL and AQ11. Technical Report 877, Computer Science Department, University of Illinois, Urbana, 1978.

[Quinlan, 1986] J.R. Quinlan. Induction of decision trees. *Machine Learning,* 1(1):81-106, 1986.

[Quinlan, 1993] J.R. Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann, 1993.

[Rumelhart *et al,* 1986] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning internal representations by error propagation. In D.E. Rumelhart and J.L. McClelland, editors, *Parallel Distributed Processing: Explorations in the Micro structure of Cognition. Vol.1: Foundations.* MIT Press, 1986.

[Smyth and Mellstrom, 1992] P. Smyth and J. Mellstrom. Detecting novel classes with applications to fault diagnosis. In *Ninth International Workshop on Machine Learning,* pages 416-425. Morgan Kaufmann, 1992.

[Wnek *et al.,* 1995] J. Wnek, K. Kaufman, E. Bloedorn, and R.S. Michalski. Selective induction learning system AQ15c: The method and user's guide. Technical Report MLI 95-4, Center for Machine Learning and Inference, George Mason University, 1995.