A Novelty Detection Approach to Classification*

Nathalie Japkowicz
Department of Computer Science
Rutgers University
New Brunswick, New Jersey 08903
nat@paul.rutgers.edu

Catherine Myers and **Mark Gluck**
Aidekman Research Center†
Rutgers University
Newark, New Jersey 07102
{myers,gluck}@pavlov.rutgers.edu

Abstract

Novelty Detection techniques are concept-learning methods that proceed by *recognizing* positive instances of a concept rather than *differentiating* between its positive and negative instances. Novelty Detection approaches consequently require very few, if any, negative training instances. This paper presents a particular Novelty Detection approach to classification that uses a Redundancy Compression and Non-Redundancy Differentiation technique based on the (Gluck & Myers 1993) model of the hippocampus, a part of the brain critically involved in learning and memory. In particular, this approach consists of training an *autoencoder* to reconstruct positive input instances at the output layer and then using this autoencoder to recognize novel instances. Classification is possible, after training, because positive instances are expected to be reconstructed accurately while negative instances are not. The purpose of this paper is to compare HIPPO, the system that implements this technique, to C4.5 and feedforward neural network classification on several applications.

Introduction

Many practical applications of supervised learning are *concept learning* problems, that is, problems that involve discriminating instances according to whether or not they belong to a given class. This class can be thought of as the concept to be learned. Usually, concept learning involves learning correct classification of a training set containing both positive and negative instances of a concept, followed by a testing phase in which novel examples are classified. Good performance often depends on constructing training sets which contain a broad range of positive and negative examples. Many classification methods have been designed under these conditions, including C4.5 (Quinlan 1993) and backpropagation applied to a feedforward neural network (FF Classification) (Rumelhart, Hinton, & Williams 1986).

As an alternative to learning a concept using a broad range of positive and negative training examples, some concept-learning techniques that require mainly—or only—positive training examples have recently been introduced (Petsche, Hanson, & Gluck 1994). Such techniques are grouped as *Novelty Detection* methods (the term stems for the fact that negative inputs are recognized as being novel compared to positive inputs which are more familiar as they belong to the class that was used for training). Novelty Detection techniques proceed by examining instances of a concept, trying to find their commonalities and generalizing from them. These techniques differ from more conventional classification approaches (C4.5, FF Classification) in that they attempt to *recognize* instances of a concept rather than to *differentiate* between instances of both classes.

The advantage of Novelty Detection approaches is that they can be used on problems that cannot easily be addressed by more conventional approaches. Such problems are those for which negative examples are very expensive or difficult to obtain. In machine fault diagnosis, for example, positive examples are plentiful and typically involve recording from the machine during normal operation. Negative examples, however, involve causing the machine to break down in each manner in which future failure is possible so that a recording can be made of each failure type. Monitoring tasks which consist of examining a system's readily available signals and issuing an alarm when a potential problem is detected fall in this category of problems. The need to monitor a system's operation arises frequently and the development of reliable Novelty Detection techniques would provide important benefits for critical military and commercial systems (e.g., helicopter gearboxes, shipboard fire pumps, motors, and generators). Novelty Detection methods, therefore, also need to demonstrate a certain level of reliability. The purpose of this paper is to introduce a particular Novelty Detection technique—Redundancy Compression and Non-Redundancy Differentiation— and demonstrate experimentally that, in addition to requir-

* This paper appears on pp. 518-523 of the Proceedings of the Fourteenth International Conference on Artificial Intelligence held on August 20-25, 1995 in Montreal (Canada).

† Center for Molecular and Behavioral Neuroscience

ing fewer negative examples, this technique is able to classify novel examples more accurately than the conventional classification approaches.

The particular Novelty Detection technique introduced in this paper uses an *autoencoder* (Hinton 1989). An autoencoder is a neural network which learns to map from its inputs, through a narrow hidden layer, to output nodes which attempt to reconstruct the input. Because the network has a narrow hidden layer, it is forced to compress redundancies in the input while retaining and differentiating non-redundant information. To implement our technique, the network is trained to reconstruct as well as possible a training set consisting of positive examples only. After having been trained on positive instances of the concept, the autoencoder should be able to adequately reconstruct subsequent positive instances, but should perform poorly on the task of reconstructing subsequent negative instances of the concept which present different structural regularities. Identifying positive and negative instances of a concept is therefore equivalent to assessing how well such instances are reconstructed by the autoencoder. These same processes have previously been proposed to model computations occurring in the hippocampus (Gluck & Myers 1993), a part of the brain involved in learning and memory.

This paper presents HIPPO, a concept-learning system based on this idea and assesses its performance by applying it to three real-world problems: CH46 Helicopters gearbox fault detection, recognition of promoter regions of DNA, and classification of sonar targets; HIPPO's error rates on these applications are then compared to those produced by C4.5 and FF Classification. The next section describes HIPPO, including its *Redundancy Compression and Non-Redundancy Differentiation* component and its *Threshold-Determination* component. The following section discusses the testing of HIPPO and its results.

HIPPO

Implementing the idea of Redundancy Compression and Non-Redundancy Differentiation results in a device able to issue two sorts of signals: one for recognized data, and an other for novel data. Designing such a device, however, is not sufficient for building a powerful, stand-alone concept-learning system: one also needs to be able to *discriminate* between "recognized" and "novel" signals. Such discrimination is called *Threshold Determination.* This section first describes HIPPO's Redundancy Compression and Non-Redundancy Differentiation component, and then discusses its Threshold Determination component. The last part is an overview of the overall functioning of the system.

Redundancy Compression and Non-Redundancy Differentiation

Redundancy Compression and Non-Redundancy Differentiation is a process believed to occur in the hippocampus (Gluck & Myers 1993). Gluck and Myers have used connectionist models to study the function of the hippocampal region in the brain and its implications for learning and memory behaviors. They suggested that the hippocampus forms new stimulus representations during learning, and that these representations specifically compress redundant information while preserving or differentiating non-redundant information. The system they built based on these considerations was able to accurately predict a range of classical conditioning behaviors observed in normal and hippocampal-damaged animals (Gluck & Myers 1993). Because these compression and differentiation constraints appear to be useful during learning in the brain, Gluck and Myers suggest that they may be useful for machine learning tasks as well.

The Redundancy Compression and Non-Redundancy Differentiation technique uses an *autoencoder*, of the type that was proposed by (Hinton 1989). An autoencoder is an artificial neural network composed of a given number, I, of input nodes; the *same* number of output nodes, O; and a given number, H, of hidden nodes with $H < I = O$. This type of network learns to reproduce its inputs at the output layer, using a multilayer learning algorithm such as *backpropagation* (Rumelhart, Hinton, & Williams 1986), the learning paradigm used in this work. Figure 1 presents an autoencoder with $I = O = 6$ and $H = 3$. In the past, autoencoders have been used for estimating learning algorithms reliability (Pomerleau 1993) and for solving the catastrophic inference problem (Kortge 1990). We now discuss how autoencoders can also be used for Novelty Detection.

In order to be used for Novelty Detection, the autoencoder is trained on positive instances of the concept, using backpropagation. Once trained, the autoencoder can be fed new instances that it tries to reconstitute at its output layer. The quality of reconstruction is evaluated by computing the sum of the absolute error at each corresponding input and output node. i.e., $Error = \sum_{i=1}^{I} |Inp(i) - Out(i)|$ where $Inp(i)$ and $Out(i)$ are the corresponding input and output nodes at position i and I is the size of the input. This error is recorded at various epochs. If after the autoencoder has been sufficiently trained, this error is small, then the instance should be labeled "positive", otherwise, it should be labeled "negative". The part of the system responsible for evaluating the size of the error and labeling the new instances is a semi-automated module called the Threshold-Determination component and will be described in section 2.2.

The phenomena that take place while using an autoencoder can be understood as follows: the narrow internal layer of the autoencoder forces it to generate an

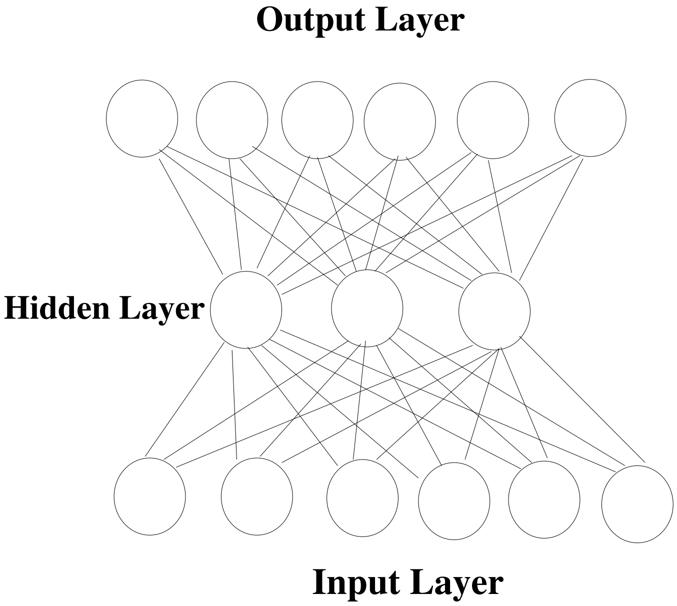


Figure 1: An Autoencoder with $I = O = 6$ and $H = 3$.

internal representation that compresses redundancies in the input pattern while retaining and differentiating non-redundant information. When the autoencoder is specifically used for classification, it is trained on positive instances only. During training, consequently, the autoencoder learns to reconstruct positive data, but does not learn to reconstruct negative data. Since negative data present structural regularities that are different from those of the positive data (i.e., the inputs that are redundant in the positive data may not be redundant in the negative data and vice-versa), reconstruction of negative data, must be done differently. At testing time, therefore, reconstruction of positive data will succeed, whereas reconstruction of negative data will fail, and this success or failure will be the criterion used in classifying new instances.

Threshold Determination

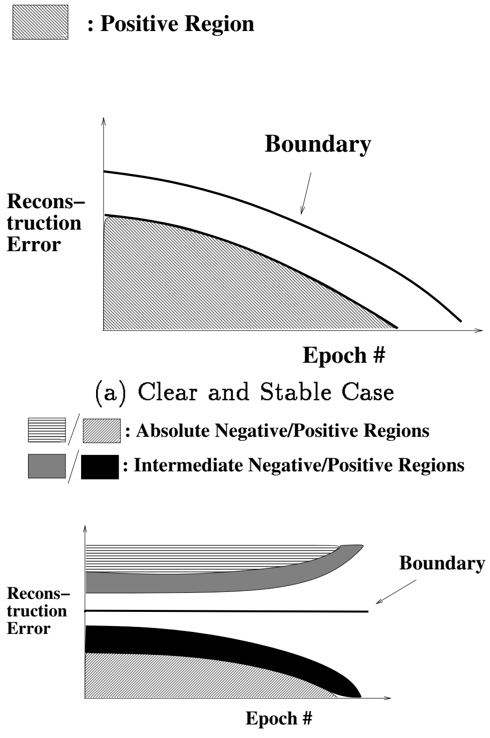
Threshold determination consists of determining a boundary that discriminates between the reconstruction errors of positive and negative data. The Threshold Determination component is a semi-automated component composed of two algorithms: one for the noiseless case, which requires only positive or only negative data and one for the noisy case, which requires both positive and negative data. For every application, one of the two algorithms is manually selected according to the availability of data and the the expected quality of separation between positive and negative reconstruction errors.

Noiseless Case In the noiseless case, the separation between positive and negative data is clear and stable in that the reconstruction error of all the positive instances is much lower than that of all the negative instances after sufficient training took place. In such a case, only positive or only negative instances are necessary.

In the case where only negative training instances are provided, the procedure we built simply computes the lower-bound of the reconstruction error of all the negative training instances at every epoch considered and then relaxes this bound by reducing it by a certain percentage. New instances are subsequently classified by checking whether the reconstruction error of the new instance is higher than that of the relaxed boundary in at least a certain acceptable proportion of the epochs considered. In such a case, the new instance is negative; otherwise, it is positive. The case where only positive training instances are provided was treated in a similar fashion. Figure 2(a) illustrates the noiseless case and shows the boundary that was derived when using only negative data. In the particular case study that uses this algorithm—CH46 Helicopter gearboxes—, the relaxation ratio was set to 25%; the epochs considered are all the recorded epochs that occur after epoch 150; and the acceptable proportion of epochs considered was set to one half.

Noisy Case In the noisy case, the separation between positive and negative data is not clear in that although the majority of positive instances have low reconstruction errors and the majority of negative instances have high reconstruction errors, some positive examples have a high reconstruction error and some negative examples have a low reconstruction error. In such a case, the Threshold Determination component needs to process both positive and negative instances in order to establish a boundary, and will need to decide what data to ignore as exceptional or possibly noisy. The procedure that we built for this case tries to find the epoch that shows the best separation between the reconstruction errors of positive and negative instances among all the epochs considered and at the same time, considers how stable this separation is.

In order to find the best separation, the procedure begins by constructing the boundaries of the *absolute* and *intermediate* positive and negative regions of the epoch versus reconstruction error space, at every epoch considered. Instances that belong to a given class with great certainty have reconstruction errors that fall in the absolute region of this class while instances that belong to this class with less certainty have reconstruction errors that fall in its intermediate region. The absolute negative region is located above the intermediate negative region, while the absolute positive region is located below the positive intermediate region. Absolute and intermediate regions differ from *actual* regions which span the entire negative and positive instance sets respectively. Absolute and intermediate regions are illustrated in Figure 2(b). To construct the negative intermediate region in partic-



(a) Clear and Stable Case

(b) Unclear Case

Figure 2: The two cases of Threshold Determination.

ular, our procedure begins by stating the lower and higher boundaries of the actual negative region, and then proceeds by repeatedly shrinking this region by manipulating its boundaries, until it believes that it found the most accurate intermediate negative region. The region located above the upper boundary of the final intermediate negative region defines the absolute negative region. In the particular case studies that use this algorithm —Promoter and Sonar Targets—, the density level of the final intermediate negative region is such that the lower half of the final intermediate negative region contains less than two fifth of the negative data while half of the negative data contained in this lower half is contained in its lowest fifth. The intermediate positive region is constructed in a similar fashion and the boundary for classifying positive and negative examples is established as the midpoint between the lower boundary of the negative intermediate region and the upper boundary of the positive intermediate region.

In order to find the epoch with the most stable separation, the program calculates how stable the separation is at each recorded epoch. The stability of a given epoch is defined as the slope of the line that goes through the separation of this epoch and the next epoch recorded. The separation that is selected for classification is the best separation whose stability is higher than half the greatest stability encountered.

Overall Functioning of the System

The training of HIPPO is carried out in two phases. In the first phase, the Redundancy Compression and Non-Redundancy Differentiation component is trained with positive instances of the concept. This phase results in the computation of a *specialized autoencoder* which can differentiate between a positive and a negative instance by showing a small reconstruction error in the positive case and a large one otherwise. The second phase of training consists of training the Threshold Determination component. In this phase, the specialized autoencoder is used with positive and/or negative instances. For each instance, the reconstruction error is recorded and fed into the Threshold Determination component which analyzes the reconstruction error of all the instances and issues a *discriminator*. The discriminator can be interpreted as a boundary between positive and negative instances.

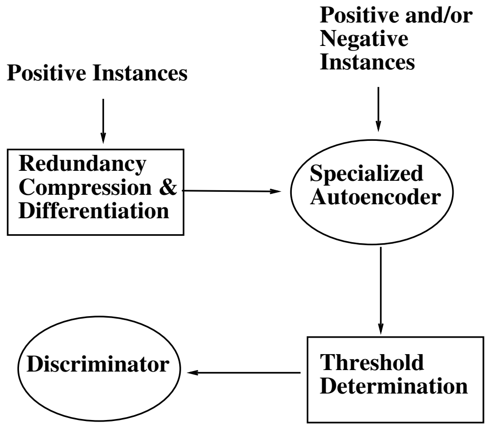
Once the two components have been trained, HIPPO can be used as follows: First, an unlabeled instance can be input to the specialized autoencoder which will issue a reconstruction error. The reconstruction error can then be input to the discriminator which will issue a classification. Figure 3 illustrates the functioning of the overall concept learner.

Experiments

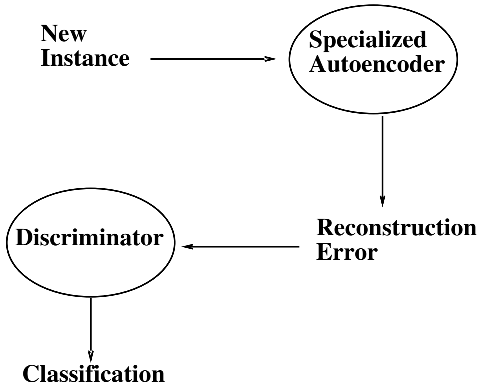
HIPPO was tested in three domains: CH46 helicopter gearbox fault detection, molecular biology promoter recognition, and sonar target classification. Its results are compared to those of two standard approaches to classification: C4.5, a decision tree learning system (Quinlan 1993) and FF Classification, another connectionist learning method (Rumelhart, Hinton, & Williams 1986). We begin by introducing the three domains considered and the methodology used to evaluate the three approaches. We then discuss the results of the experiments.

The Case Studies

The CH46 Helicopter Gearbox data was obtained from NRaD.(Kolesar & NRaD 1994). The CH46 Helicopter problem is a monitoring problem that consists of discriminating between faulty and non-faulty CH46 helicopter gearboxes, according to the whining sound they emit during their operation. The sudden, unexpected failure of CH46 helicopter gearboxes is currently very costly both in terms of lives and equipment. The development of a monitoring system that can identify imminent failures before takeoff or when in flight is of paramount importance. The data for this problem was obtained by pre-processing the vibration time signal of the gearboxes of various faulty and non-faulty helicopters. The complete data set is composed of 18 non-faulty instances and 46 faulty ones which come in the form of 256 long vectors of real numbers. In this particular problem, we chose the non-faulty examples to represent the positive class.



(a) Training Phase



(b) Testing Phase

Figure 3: Overall Functioning of the System.

The Promoter problem takes as input segments of DNA, some subset of which represent promoters. A promoter is a sequence that signals to the chemical processes acting on the DNA where a gene begins. The goal of the problem is to train a classifier to be able to recognize promoters, which are taken to be the positive class. The training set is composed of 100 examples (47 promoters and 53 negatives), each of which is composed of a set of 51 nucleotides, where each nucleotide can take one of four values {a, c, g, or t}. The promoter data was obtained from the U.C. Irvine Repository of Machine Learning and was modified in response to Norton's critique of the biological flaws underlying the original formulation of the data (Norton 1994). In addition, as is usual for this problem when run on a connectionist system, each example was converted into a 204-bit long vector where each nucleotide was represented with 4 bits.

The Sonar Target Recognition problem takes as input the signals returned by a sonar system in the cases where mines and rocks were used as targets. The sonar data was obtained from the U.C. Irvine Repository of Machine Learning though only a subset of 100 instances (47 positive and 53 negative) from this data was used in this particular case study.[1]. The transmitted sonar signal is a frequency-modulated chirp, rising in frequency. The data set contains signals obtained from a variety of different aspect angles. Each instance of this data is represented as a 60-bit long vector. In this particular case study, we chose the signals returned by the mine targets to constitute the positive class.

General Methodology

Connectionist models such as HIPPO and FF Classification are more difficult to train than symbolic models like C4.5. Not only do connectionist models require to be tuned before they can actually be trained but also, their training requires two phases: concept-learning and threshold-determination. No tuning is necessary and a single phase is sufficient for symbolic models.

The learning rate, momentum, and bias for HIPPO and FF Classification were arbitrarilly set to 0.05, 0.9, and 1.0 respectively, and held constant for all three case studies. The number of hidden units and recorded epochs were determined experimentally for each case study on random subsets of the entire data sets. For the helicopter gearbox application, HIPPO used 32 and FF Classification used 50 hidden units. Both systems were run for 200 epochs which were recorded every 10 epochs for the first 190 epochs and every epoch subsequently. For the promoter problem, both systems used 153 hidden units while for the sonar target recognition problem, they used 20 hidden units. In both applications, the system were run for 100 epochs which were recorded every 10 epochs.

The threshold-determination method of section 2.2 was used with both HIPPO and FF Classification. It was tuned on random subsets of the entire data sets.[2] In every case study, the thresholds were established on the same data sets for both systems. For the helicopter gearbox problem, the noiseless method of section 2.2 was selected and applied to 10 negative instances. For both the promoter and the sonar target recognition problems, the noisy method of section 2.2 was selected and applied to 5 positive and 5 negative instances.

In the three domains considered, the three systems were evaluated using 5-fold crossvalidation (Weiss & Kulikowski 1991). At every fold of every experiment, the training set used by C4.5 was divided into a training set for concept-learning and one for threshold-determination for HIPPO and FF Classification. Since HIPPO learns a concept from positive data only, the negative data was eliminated from the

[1] This explains why the results reported in section 3.3 are different from those reported in previous experiments on this data, such as (Gorman & Sejnowski 1988)

[2] For use with FF Classification, the input of the threshold-determination component was taken to be the value of the output node and its output has to be reversed since positive instances are supposed to return a larger signal than negative ones.

concept-learning training set of this system while it was kept for FF Classification. In every experiment, the testing sets of the three systems always corresponded. Note that at every fold of every experiment, HIPPO uses significantly fewer negative data for overall training than the other two systems: for the CH46 helicopter gearbox problem, HIPPO uses 10 negative data while FF Clasification and C4.5 use between 35 and 39 such data. For the other two case studies, HIPPO uses 5 negative instances while FF Classification and C4.5 use between 40 and 44 such instances.

Results

The error rates of HIPPO, C4.5, and FF Classification in the three case studies considered are listed in Table 1. Numbers after each "±" are standard deviations for each of the five-fold averages.

Case Study	HIPPO *Error (%)*	*C4.5 Error (%)*	*FF Classif. Error (%)*
Helicopters	3.125 ± 0.9	15.625 ± 1.9	10.9 ± 1.7
Promoters	20 ± 0.7	35 ± 1.4	20 ± 1.4
Sonar Targets	20 ± 2.7	29 ± 1.8	32 ± 3.2

Table 1: Results for the three case studies.

Table 1 shows that in both the CH46 Helicopter and the Sonar Target recognition case studies, HIPPO performed much better than either FF Classification or C4.5. In the Promoter case study, HIPPO and FF Classification performed equally well and better than C4.5. These comparisons are all statistically significant with $p < .05$, except for the comparison with C4.5 in the sonar target recognition study.

Altogether, this shows that in addition to requiring a much smaller number of negative training data than the other two systems, HIPPO is capable of classifying novel instances more accurately than both C4.5 and FF Classification in all cases except for the Promoter data, where HIPPO's performance is matched by FF Classification's. However, we believe that the limited performance of HIPPO with respect to FF Classification has to do with the weakness of the representation used in this version of the promoter problem (Hirsh & Noordewier 1994; Norton 1994).

Conclusion

This paper has presented a new approach to classification that uses the idea of Novelty Detection and in particular, that of Redundancy Compression and Non-Redundancy Differentiation. The system we introduced—HIPPO— attempts to learn how to recognize concepts, rather than to differentiate between positive and negative instances of a concept. The method works in two phases. In a first phase, a concept is learned from positive instances only and in a second phase, the system learns how to identify positive and negative instances of that concept. HIPPO was tested on three real-world applications and compared with two conventional classification systems: C4.5 and Feedforward Classification. In all applications, HIPPO performed better than C4.5 and in two of them, it performed better than Feedforward Classification (in the third application HIPPO and Feedforward Classification performed equally well). Furthermore, in all applications, HIPPO used a significantly smaller number of negative training data than the other two systems.

The work presented in this paper opens up a large number of possible theoretical and practical issues to consider in the future. It would be useful, in particular, to establish the strength and limitations of our approach more precisely, by experimenting in other domains (both artificial and real) and comparing HIPPO's results with methods other than C4.5 and Feedforward Classification. We could also attempt to improve the two components of HIPPO, using a more refined version of the autoencoder and fully automating the Threshold Determination component. Such studies would contribute to the exploration of this promising new approach to concept-learning which is more accurate than conventional methods and requires fewer negative data for training.

Acknowledgements

This research was supported by the Office of Naval Research through the Young Investigator program and grant N00014-88-K-0112. Parts of this work was conducted at the Centre for Neural Networks at King's College in London and the Laboratoire Laforia at l'Université Pierre et Marie Curie in Paris. We thank Rick Kaye and Brian Davison for helpful comments on earlier drafts of this paper, and Bob Kolesar for his contributions to the helicopter analysis and feature extraction.

References

Gluck, M. A., and Myers, C. E. 1993. Hippocampal mediation of stimulus representation: A computational theory. *Hippocampus* 3(4):491–516.

Gorman, R., and Sejnowski, T. 1988. Analysis of hidden units in a layered network trained to classify sonar targets. *Neural Networks* 1:75–89.

Hinton, G. 1989. Connectionist learning procedures. *Artificial Intelligence* 40:185–234.

Hirsh, H., and Noordewier, M. 1994. Using background knowledge to improve learning of DNA sequences. In *Proceedings of the Tenth IEEE Conference on Artificial Intelligence for Applications*, 351–357.

Kolesar, B., and NRaD. 1994. Helicopter gearbox monitoring. In *NIPS-94 Workshop on Novelty Detection and Adaptive System Monitoring (presented).*

Kortge, C. A. 1990. Episodic memory in connectionist networks. In *The Twelfth Annual Conference of the Cognitive Science Society*, 764–771.

Norton, S. W. 1994. Learning to recognize promoter sequences in E. coli by modeling uncertainty in the training data. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*.

Petsche, T.; Hanson, S.; and Gluck, M. 1994. Workshop on novelty detection and adaptive system monitoring. In *Neural Information Processing Systems. Volume 7*.

Pomerleau, D. A. 1993. Input reconstruction reliability estimation. In *Proceedings of the Fifth Neural Information Processing Systems Conference*.

Quinlan, J. R. 1993. *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann.

Rumelhart, D. E.; Hinton, G. E.; and Williams, R. J. 1986. Learning internal representations by error propagation. In Rumelhart, D. E., and McClelland, J. L., eds., *Parallel Distributed Processing*. Cambridge, MA: MIT Press. 318–364.

Weiss, S. M., and Kulikowski, C. A. 1991. *Computer Systems That Learn*. San Mateo, CA: Morgan Kaufmann.