A Recognition-based Alternative to Discrimination-based Multi-Layer Perceptrons

Todd Eavis and Nathalie Japkowicz

Faculty of Computer Science DalTech/Dalhousie University 6050 University Avenue
Halifax, Nova Scotia Canada, B3H 1W5

Abstract. Though impressive classification accuracy is often obtained via discrimination-based learning techniques such as Multi-Layer Perceptrons (DMLP), these techniques often assume that the underlying training sets are optimally balanced (in terms of the number of positive and negative examples). Unfortunately, this is not always the case. In this paper, we look at a recognition-based approach whose accuracy in such environments is superior to that obtained via more conventional mechanisms. At the heart of the new technique is a modified auto-encoder that allows for the incorporation of a recognition component into the conventional MLP mechanism. In short, rather than being associated with an output value of "1", positive examples are fully reconstructed at the network output layer while negative examples, rather than being associated with an output value of "0", have their inverse derived at the output layer. The result is an auto-encoder able to recognize positive examples while discriminating against negative ones by virtue of the fact that negative cases generate larger reconstruction errors. A simple technique is employed to exaggerate the impact of training with these negative examples so that reconstruction errors can be more reliably established. Preliminary testing on both seismic and sonar data sets has demonstrated that the new method produces lower error rates than standard connectionist systems in imbalanced settings. Our approach thus suggests a simple and more robust alternative to commonly used classification mechanisms.

1 Introduction

Concept learning tasks represent a form of supervised learning in which the goal is to determine whether or not an instance belongs to a given class. As would be expected, the greater the number of training examples, the more reliable the results obtained during the training phase. In addition, however, we must also acknowledge that the success of supervised learning algorithms is at least partly determined by the balance of positive and negative training cases. For training purposes, then, we would consider a data set optimal if, in addition to a certain minimal size, its instances were split more or less evenly between positive and negative examples of the concept in question. This type of division would ensure that our learning algorithms are not unduly skewed in favour of one case or the other.

Unfortunately, such optimality is often hard to guarantee in practice. In many domains, it is neither possible nor feasible to obtain equal numbers of positive and negative instances. For example, the analysis of seismic data in terms of its association with either naturally occurring geological activity or man-made nuclear devices is hampered by the fact that examples of the latter are extremely uncommon (and rigidly controlled). Seismic applications are not the only ones suffering from imbalanced conditions. The problem was also documented in applications such as the detection of oil spills in satellite radar images [Kubat et al., 1998], the detection of faulty helicopter gearboxes [Japkowicz et al., 1995] and the detection of fraudulent telephone calls [Fawcett & Provost, 1997]. Thus, supervised learning algorithms used in such environments must be amenable to these inherent restrictions.

In practice, many algorithms do not perform well when the training set is imbalanced (see [Kubat et al. 1998] for an illustration of this effect). Since a significant number of real-world domains can be described in this manner, it seems logical to pursue mechanisms whose performance suffers less drastically when counter examples are relatively hard to come by. In this paper, we present preliminary results obtained via the use of a Connectionist Novelty Detection method known as auto-encoder-based classification. Essentially, auto-encoder-based classifiers learn how to recognize positive instances of a concept by identifying their common patterns. When later presented with novel instances, the auto-encoder is able to recognize cases whose characteristics are in some way similar to its positive training examples. Negative instances, on the other hand, generally have little in common with the training input and are therefore not associated with the concept under investigation.

Though the auto-encoder as just described has been successful within a number of domains, it has become clear that not all environments are equally receptive to a training phase completely devoid of counter examples. More specifically, auto-encoders tend not to be as effective when negative instances of the concept exist as a subset of the larger positive set. In such cases, the network is likely to confuse counter examples with the original training cases since it has had no opportunity to learn those patterns which can serve to delineate the two. Consequently, the method presented here will incorporate a local discrimination phase within the general recognition-based framework. The result is a network that can successfully classify mixed instances of the concept, despite having been given a decidedly imbalanced training set.

2 Previous Work

Although the imbalanced data set problem is starting to attract the attention of a number of researchers, attempts at addressing it have remained uncoordinated. Nevertheless, these research efforts can be organized into four categories

- Methods in which the class represented by a small data set gets over-sampled so as to match the size of the opposing class.

- Methods in which the class represented by the large data set can be downsized so as to match the size of the other class.
- Methods that internally bias the discrimination-based process so as to compensate for the class imbalance.
- Methods that ignore (or makes little use of) one of the two small classes altogether.

The first method was used by [Ling & Li, 1998]. It simply consists of augmenting the small data set by re-sampling the instance multiple times. Other related schemes could diversify the augmented class by injecting some noise into the repeated patterns. The second method was investigated in [Kubat & Matwin, 1997] and consists of removing instances from the well-represented class until it matches the size of the smaller class. The challenge of this approach is to remove instances that do not provide essential information to the classification process. The third approach was studied by [Pazzani et al., 1994] who assigns different weights to examples of the different classes, [Fawcett & Provost, 1997] who remove rules likely to over-fit the imbalanced data set, and [Ezawa et al., 1996] who bias the classifier in favour of certain attribute relationships. Finally, the fourth method was studied in its extreme form (i.e., in a form that completely ignores one of the classes during the concept-learning phase) by [Japkowicz et al., 1995]. This method consisted of using a recognition-based rather than a discrimination-based inductive scheme. Less extreme implementations were studied by [Riddle et al., 1994] and [Kubat et al., 1998] who also employ a recognition-based approach but use some counter-examples to bias the recognition process. Our current study investigates a technique that falls along the line of the work of [Riddle et al., 1994] and [Kubat et al., 1998] and extends the auto-encoder approach of [Japkowicz et al., 1995] by allowing it to consider counter examples. The method, however, differs from [Riddle et al., 1994] and [Kubat et al., 1998] in its use of the connectionist rather than rule-based paradigm.

Our method is also related to previous work in the connectionist community. In the past, auto-encoders have typically been used for data compression [e.g., Cottrell et al., 1987]. Nevertheless, their use in classification tasks has recently been investigated by [Japkowicz et al., 1995], [Schwenk & Milgram, 1995], [Gluck & Myers, 1993] and [Stainvas et al., 1999]. [Japkowicz et al., 1995] and [Schwenk & Milgram, 1995] use it in similar ways. As mentioned previously, [Japkowicz et al., 1995] use the auto-encoder to recognize data of one class and reject data of the other class. [Schwenk & Milgram, 1995], on the other hand, use it on multi-class problems by training one auto-encoder per class and assigning a test example to the class corresponding to the auto-encoder which recognized it best. Both [Gluck & Myers, 1993] and [Stainvas et al., 1999] use the auto-encoder in conjunction with a regular discrimination-based network. They let their multi-task learner simultaneously learn a clustering of the full training set (including conceptual and counter-conceptual data) and discriminate between the two classes. The discrimination step acts as both a labelling step (in which the clusters uncovered by the auto-encoder get labelled as conceptual or not) and a fine-tuning step (in which the class information helps refine the auto-encoder clustering). Our method is similar to [Gluck & Myers, 1993] and [Stainvas et al., 1999] in that it too uses class information about the two classes and lets this information act both as a labelling and a fine-tuning step. However, it differs in that the auto-encoder is used in a different way for each class.

3 Implementation

Auto-encoder. As stated, an auto-encoder learns by determining the patterns common to a set of positive examples (in the standard case). It then uses this information to generalize to examples it has not seen before. In terms of the training itself, the key component is a supervised learning phase in which input samples (in the form of a multi-featured vector) are associated with an appropriate target vector. The target, in fact, is simply a duplicate of the input itself. In other words, the network is trained so as to reproduce the input at the output layer. This, of course, stands in contrast to the conventional neural network concept learner which is trained to associate positive instances with a target value of "1" and negative examples with a "0". The architectures of the auto-encoder (with 6 input/output units and 3 hidden units) versus that of the conventional neural network (with 6 input units, 3 hidden units and 1 output unit) are illustrated in Figure 1.

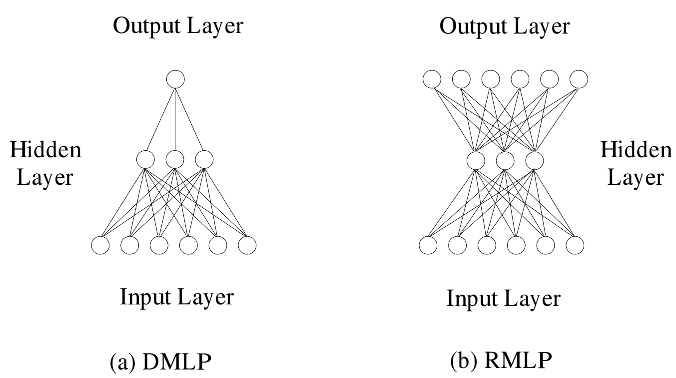

Fig. 1. Examples of Feedforward Neural Networks: (a)discrimination-based DMLP; (b) recognition-based RMLP

Once an auto-encoder has been trained, it is necessary to provide a means by which new examples can be accurately classified. Since we no longer have a simple binary output upon which to make the prediction, we must turn to what is called the "reconstruction error". The reconstruction error is defined as

$$\sum_{i=1}^{k}[I(i) - O(i)]^2 \tag{1}$$

where I(i) and O(i) are the corresponding input and output nodes at position i and k represents the number of features in the vector. In other words, we ascertain the degree to which the output vector — as determined by the trained network — matches the individual values of the original input. Of course, in order to apply the reconstruction error concept, we must establish some constraint on the allowable error for instances that will be deemed to be positive examples of the given concept. To do so, we include a threshold determination component in the training phase. Since we would like the threshold T to be closely associated with the mean of the full set R of individual reconstruction errors, we define the threshold as follows

$$T = mean(R) + Z(std(R)) \tag{2}$$

We use the Z parameter as a means of controlling the range of acceptable reconstruction error values. In essence, it represents the desired confidence interval for the mean reconstruction error and, as such, corresponds to the standard Z value commonly used in statistical analysis. By combining the Z value with the mean and standard deviation of the error distribution, we may efficiently tune our boundary to suit the input at hand.

Extended auto-encoder. In the current study the auto-encoder has been extended to allow for what we call local discrimination. In other words, a small number of negative examples are included in the training set so that the network has the opportunity to determine those features that differentiate clusters of negative instances from the larger set of positive instances. In cases where there is considerable overlap between concept and non-concept instances, it is expected that this additional step may significantly lower classification error. The extension to the new model is relatively straight-forward. As before, target values for positive input are represented as a duplication of the input vectors. In contrast, however, the target vector for negative instances is constructed as an inversion of the input. For example, the three-tuple $<$0.5, 0.6, 0.2$>$ would become $<$-0.5, -0.6, -0.2$>$ at the output layer. During the threshold determination phase, the network output vectors for these negative examples are assessed relative to the vectors that would have been expected had the input actually been a positive example. In other words, the negative reconstruction error is the "distance" between the inverted output and the original input.

Armed with this new information, we are now able to establish both positive and negative reconstruction error ranges. A definitive classification boundary is determined by finding the specific point that offers the minimal amount of overlap. Though this might at first appear to be a trivial task, in practice it is somewhat more complicated than expected. Typically, due to the underlying data imbalance, the range of positive reconstruction errors is much more tightly defined than the range of negative errors (i.e., more compactly clustered around the mean). For this reason, it is necessary to skew the boundary towards the mean of the positive reconstruction error. In our study, this extra step was not required since we employed a "target shifting" technique (see below) that significantly reduced the likelihood of boundary overlap. As a result, we were simply able to utilize the positive reconstruction boundary as described in the preceding section. Once the final boundary condition has been established, classification of novel examples is relatively simple. Instances are passed to the trained network and reconstruction errors calculated. Errors below the boundary are associated with positive examples of the concept, while those above signify non-concept input.

Target Shifting. In theory, the local discrimination technique as just described should produce distinctive error patterns for both positive and negative training input. Unfortunately, initial testing using this basic scheme was quite disappointing. In particular, it proved almost impossible to produce non-overlapping error ranges. An analysis of the raw network output showed that the auto-encoder was indeed inverting the negative training cases. However, it was clear that the negative reconstruction was simply much smaller than expected. The problem was two-fold. First, the inclusion of empty or zeroed feature values effectively reduced the number of vector elements that could contribute to the reconstruction error. For example, if a domain provides twenty distinctive features for each instance, but individual cases rarely have more than five or six non-zero feature values, then the ability of the network to produce distinguishable error ranges is severely curtailed. Second, and perhaps more importantly, the normalization of input prior to training can have a deleterious effect upon threshold determination. In particular, the existence of out-liers in the original input set has a tendency to squash many feature values down towards zero. If these near-zero features belong to negative training examples, then the inverted features will be deceptively close to the non-inverted input. For example, a normalized input value of 0.001 would become -0.001 in a perfectly trained network. Consequently, it is likely that the reconstruction error for many negative training cases will be no greater than their positive counterparts.

To combat this problem, it was necessary to utilize some mechanism that could exaggerate the error associated with negative examples while leaving the positive reconstruction error unchanged. We chose to employ a simple technique by which the entire normalized range of input values was shifted in order to create target output. Positive instances are modified simply by incrementing each element of the input vector by one. Negative input is also incremented but, in this case, the sign of each element is also inverted. For example, the positive input vector <0.2, 0.3, 0.4> becomes <1.2, 1.3, 1.4> while the negative vector <0.2, 0.5, 0.9> becomes <-1.2, -1.5, -1.9>. This approach to target vector generation has two fundamental advantages. First, it maintains the normalized input patterns so that features with large absolute values do not dominate the training phase. Second, and more significantly in the current context, we can ensure that properly recognized negative instances will result in the generation of significantly exaggerated reconstruction errors. Typically, negative instances produce values greater than two for each component of the vector while positive instances contribute errors of less than one per component. (Note: We say "typically" since the network is unlikely to perfectly transform all features into the expected ranges). In the initial implementation, only non-zero vector elements were actually inverted, the belief being that transforming these "empty" features might hamper the network's ability to properly recognize the original input. However, in practice, the primary result of not shifting the zero values was to distribute more evenly target output and, in the process, to bring positive and negative boundaries much closer together. As a result, all values were inverted in subsequent experiments.

Target shifting has proven to be a simple but effective technique for establishing appropriate reconstruction error constraints. As should be obvious, domains exhibiting a higher number of features generally produce more striking differences between positive and negative boundaries. Nevertheless, even for feature-poor domains, it is generally quite easy to define the appropriate ranges.

4 Seismic Data

Description. We applied our technique to the problem of learning how to discriminate between seismograms representing earthquakes and seismograms representing nuclear explosions. The database contains data from the Little Skull Mountain Earthquakes 6/29/92 and its largest aftershocks, as well as nuclear explosions that took place between 1978 and 1992 at a nuclear testing site near the Lawrence Livermore Labs. The long-range motivation for this application is to create reliable tools for the automatic detection of nuclear explosions throughout the world[1] in an attempt to monitor the Comprehensive Test Ban Treaty. This discrimination problem is extremely complex given the fact that seismograms recorded for both types of events are closely related and thus not easily distinguishable. In addition, due to the rarity of nuclear explosions and earthquakes occurring under closely related general conditions (such as similar terrain) that can actually allow for fair discrimination between the two types of events, significant imbalances in the data sets can be expected.[2] Specifically, our database contains more nuclear explosion than earthquake data since the chances of natural seismic activity taking place near the nuclear testing ground are very slim. Nevertheless, there is a strong appeal in automating the discrimination task since, if such a computer-based procedure could reach acceptable levels of accuracy, it would be more time-efficient, less prone to human-errors, and less biased than the current human-based approaches.

The seismic data set used in the study is made up of 49 samples, 31 representing nuclear explosions and 18 representing naturally occurring seismic activity. Each event is represented by 6 signals which correspond to the broadband (or long period) components BB Z, BB N and BB E and the high frequency (or short period) components HF Z, HF N and HF E. Z, N, and E correspond to

[1] Relevant seismic data can be recorded in a station located thousands of kilometers away from the site of the event.

[2] In the more useful setting where seismograms can be transformed so that the surrounding conditions do not need to be constant, large imbalances in the data sets would remain, though this time they would be caused by the scarcity of nuclear explosions and the ubiquity of earthquakes of various types around the world.

the Vertical, North and East components that refer to projections of the seismograms onto the Vertical, North and East directions, respectively. All the signals were recorded in the same locale although the earthquake data is divided into three different classes of events that took place at different locations within that area and at different points in time. Because the broadband recordings were not specific enough, we worked instead with the short period records.

The signal onset was selected manually by inspection of each signal. Since the exact onset could not always be determined, the starting point was uniformly chosen so as to be slightly past the actual onset for all signals. Clipping was done after 4096 recording. This number was chosen because it includes the most informative part of the signals and it is also a power of 2. (Because of the second feature, application of the Fast Fourier transform using the MATLAB Statistical package is faster.) Although the overall files did contain spikes, the parts of the signals selected for this study were not spiky, so no additional procedure needed to be applied in order to deal with this issue. The signals were then de-trended, transforming them so that the collective set exhibited a zero mean. Next, the signals were normalized between 0 and 1 in order to make them suitable for classification. Finally, the signal representation was changed by converting the time-series representation to a frequency representation using MATLAB's Fast Fourier transform procedure. Although some of the earthquake files seemed to contain several events, we only kept the first one of these events in each case.

Experimental Details. All training and testing was conducted within MATLAB's Neural Network Toolbox. As mentioned, data was normalized into a 0-1 range (before target shifting) and features made up of zero values across all input vectors were removed. Network training was performed using the Resilient Backprop (Rprop) algorithm which offered extremely rapid training times in our study (for a more complete description of Rprop, see [Riedmiller & Braun, 1993]).

To assess the impact of the auto-encoding with local discrimination, we chose a pair of comparative tests. In the first case, each of three relevant network architectures — DMLP (conventional discrimination-based MLP), RMLP (basic recognition-based auto-encoder), and XRMLP (auto-encoder with local discrimination) — was trained on a "set" number of seismic records (RMLP, of course, relied only upon positive training instances). We must note, here, that the relatively small size of the data sample made network training and testing somewhat difficult. Splitting the input set into two equal subsets for training/testing and cross-validation would have left too few cases in each of the partitions; results would likely have been too inconsistent to have been of much value. Instead, network parameters were established by using the entire set as a training/testing set. Two thirds of the positive and negative cases were chosen at random and were used for training, while the remaining third went into a test set. Hidden unit counts of 16 for DMLP and 64 for both auto-encoders were established in this manner (Rprop does not use a learning rate or momentum constant). The data set was then re-divided into five folds and, using the hidden unit parameters established in the previous step, three separate test cycles of 5-fold cross-validation were performed. On the full (i.e, relatively balanced) data set, the XRMLP network produced a cross-validated error of 0.161, while the DMLP and RMLP generated 0.212 and 0.387 respectively.

In the second - and more significant - phase of testing, our goal was to directly compare the impact of reducing the number of negative training units upon the error rate of both DMLP and XRMLP. In this phase, we used from 1 to 10 negative training samples and performed 5 network training cycles at each level. (Here, final testing was performed on the full data set since there were simply not enough negative examples to use the previous 5-fold cross-validation technique) Figure 2 is a graphical representation of the results, while Table 1 lists both the mean and standard deviation for each of the separate tests.

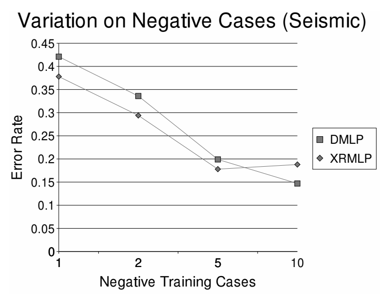

Fig. 2. DMLP vs. XRMLP

There are two points of interest with respect to Table 1. First, the auto-encoder provides a lower error rate when small numbers of negative samples are used; only at ten instances does the DMLP show improved accuracy. Second, even though the mean error rate of the DMLP diminishes as the number of negative samples increases, its standard deviation is much higher than that of the auto-encoder in the last two recordings (i.e., 5 and 10 cases). The implication, of course, is that DMLP is much more dependent upon the specific set of negative training samples with which it is supplied.

Table 1. Negative sample variation

	Architecture	
Negative Samples	DMLP	XRMLP
1	.421 $\pm$0.0	.378 $\pm$.068
2	.336 $\pm$.029	.294 $\pm$.047
5	.199 $\pm$.086	.178 $\pm$.029
10	.147 $\pm$.125	.188 $\pm$.047

5 Sonar Data

Description. The sonar detection problem takes as input the signals returned by a sonar system in those cases where mines and rocks were used as targets. Transmitted sonar signals take the form of a frequency-modulated chirp, rising in frequency. In the current context, signals were obtained from a variety of different aspect angles. Each instance of the data is represented as a 60-bit long vector spanning 90 degrees for the mine and 180 degrees for the rock. Samples are represented as a set of 60 numbers in the range 0.0 to 1.0., where each number represents the energy within a particular frequency band, integrated over a certain period of time. The data itself was obtained from the U.C. Irvine Repository of Machine Learning. In total, there were 111 mine samples (positive class) and 97 rock samples (negative class).

Eperimental Details. As before, experimental results were obtained using MATLAB's Neural Net Toolbox. Though Rprop again provided good performance for the DMLP network, it was not as effective in the XRMLP environment. More specifically, even while using the target shifting technique, we found significant overlap between the positive and negative reconstruction boundaries, so much so that Rprop results proved unreliable at best. Experimentation with a variety of other training functions[3] eventually demonstrated that One Step Secant (OSS) was most appropriate for this particular data set. (Note: DMLP accuracy did not improve with any of these other training functions.) What was most interesting about OSS was the definitive nature of its classification decisions (for further information regarding OSS, see [Batti, 1992]). Though it did not always classify correctly, there was generally little question as to how to assess the output vectors; positive reconstruction errors were typically very small (i.e., less than 5) while negative reconstruction errors were quite large (i.e., greater than 100).

The decisive classification of OSS, however, necessitated some minor changes in the boundary determination phase. Because of the marked difference in the absolute values of the individual positive and negative reconstruction errors, it was possible for a small number of network classification errors to grossly inflate the mean reconstruction error. As such, subsequent testing would be adversely affected in that a number of negative test cases would likely fall inside the inflated boundary and be classified incorrectly. Our solution, therefore, was to prune the original set of reconstruction errors by removing all error values that clearly represented classification errors. In this case the heuristic used was to exclude those values which were more than five times greater than the median value in the reconstruction set.

In terms of the tests themselves, we chose to focus exclusively on the comparison of DMLP and XRMLP. Data was randomly divided into a training set of 108 samples (61 positive, 47 negative) and a cross-validation testing set of 100 samples (50 positive, 50 negative). For both networks, training produced

[3] One Step Secant, Gradient Descent backpropagation, Bayesian Regulation backpropagation, and One-vector-at-a-time training

an optimal hidden unit count of 32, though the OSS mechanism proved to be relatively robust at a variety of hidden unit counts. As was the case during the seismic testing, we were interested in the impact upon classification accuracy as the number of negative training samples decreased. Therefore, we compared the two architectures by training the networks on a small number of negative samples randomly selected from the available training set. Figure 3 graphically displays the results for negative training samples in the range one to ten (using five-fold cross-validation), while Table 2 depicts the same results in the form of a 95% confidence interval.

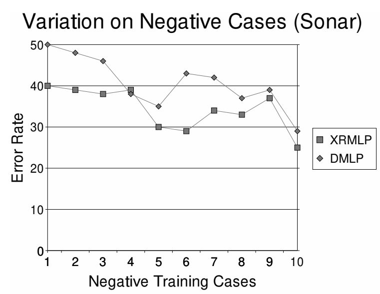

Fig. 3. DMLP vs. XRMLP

As before, the results clearly demonstrate the benefit of the XRMLP mechanism within dramatically imbalanced settings. Inside the approximately 5:1 ratio represented by these tests (47 positive cases versus a maximum of 10 negative training cases), the recognition-based approach extracted more information from the training samples than did the discrimination-based alternative. We should also note that when the two networks were trained in a balanced environment, DMLP performance was superior to the XRMLP (0.22 error versus 0.29), perhaps suggesting XRMLP accuracy does not necessarily benefit from the unlimited addition of negative training cases. Even so, however, we must recognize that the accuracy of XRMLP on a limited training set is relatively close to that of DMLP of a fully-balanced data set.

6 Conclusions and Future Work

In this paper, we have discussed an extension to the auto-encoder model which allows for a measure of local discrimination via a small number of negative training examples. Comparisons with the more conventional DMLP model suggest that not only does the new technique provide greater accuracy on imbalanced data sets, but that its effectiveness relative to DMLP grows as the ratio of positive to negative training cases becomes more exaggerated. In addition, we have noted that the auto-encoder appears to be much more stable in this type of environ-

Table 2. Negative sample variation

	Architecture	
Negative Samples	DMLP	XRMLP
1	.50 ±.08	.40 ±.06
2	.48 ±.03	.39 ±.14
3	.46 ±.03	.38 ±.11
4	.38 ±.07	.39 ±.14
5	.35 ±.08	.30 ±.15
6	.43 ±.07	.29 ±.18
7	.42 ±.13	.34 ±.05
8	.37 ±.11	.33 ±.14
9	.39 ±.07	.37±.16
10	.29 ±.09	.25 ±.09

ment, in that its error rates tend to fluctuate relatively little from one iteration of the network to the next.

The lack of success shown by the basic auto-encoder (i.e., without negative training samples) also demonstrates that some form of local discrimination is important in certain environments. Though such an architecture has proven very effective in other settings, it seems clear that the underlying characteristics of concept and non-concept instances may sometimes be too similar to distinguish without prior discriminatory training.

There are many possible extensions of this work. First, it will be important to assess the accuracy of XRMLP on larger data sets; doing so will allow us to experiment with a wide range of imbalance ratios. (Note: preliminary work in this regard has been promising). Second, a possible approach to the seismic problem that appears promising involves the use of radial basis functions. Third, it would be interesting to compare our method to that of [Stainvas et al., 1999], though our technique should be implemented within an ensemble framework in order for the comparison to be fair. Finally, it could be useful to extend the auto-encoder-based technique described in this paper to multi-class learning (by assigning different goals for the reconstruction error of each class) and to compare this method to the standard multi-class neural network technique.

Acknowledgements

This work was partially supported by a grant from Israel's Atomic Energy Commission and the Council for Higher Education through Dr. Gideon Leonard and Dr. Nathan Intrator at Tel-Aviv University. We would also like to thank Drs. Nathan Intrator and Manfred Joswig for their guidance in using the seismic data and addressing the problem. In addition, we would like to thank the three anonymous reviewers who provided thorough comments on an earlier draft, as well as Michelle Rau-Chaplin for her assistance in preparing the final version of the paper.

References

1. Batti, R., 1992. “First and Second Order Methods for Learning: Between Steepest Descent and Newton’s Method”, Neural Computation, vol. 4, no. 2, pp. 141-166.
2. Cottrell, G. W., Munro, P. and Zipser, D., 1987. "Learning Internal Representations from Gray-Scale Images: An Example of Extensional Programming", Proceedings of the 1987 Conference of the Cognitive Science Society, pp. 462-473.
3. Ezawa, K.J., Singh, M. and Norton, S.W., 1996. "Learning Goal Oriented Bayesian Networks for Telecommunications Management. Proceedings of the Thirteenth International Conference on Machine Learning, pp. 139-147.
4. Fawcett, T. E. and Provost, F., 1997. "Adaptive Fraud Detection", Data Mining and Knowledge Discovery, 1(3):291-316.
5. Gluck, M.A. and Myers, C., 1993. "Hippocampal Mediation of Stimulus Representation: A Computational Theory", Hippocampus, 4(3): 491-516.
6. Japkowicz, N. Myers, C. and Gluck, M.A., 1995. "A Novelty Detection Approach to Classification", Proceedings of the Fourteenth Joint Conference on Artificial Intelligence, pp. 518-523.
7. Kubat, M. and Matwin, S., 1997. "Addressing the Curse of Imbalanced Data Sets: One-sided Sampling", Proceedings of the Fourteenth International Conference on Machine Learning, pp. 179-186.
8. Kubat, M., Holte, R. and Matwin, S., 1998. "Machine Learning for the Detection of Oil Spills in Satellite Radar Images", Machine Learning, 30:195-215.
9. Ling, C. and Li, C., 1998. "Data Mining for Direct Marketing: Problems and Solutions", Proceedings of KDD-98.
10. Pazzani, M. and Merz, C. and Murphy, P. and Ali, K. and Hume, T. and Brunk, C., "Reducing Misclassification Costs", Proceedings of the Eleventh International Conference on Machine Learning, pp. 217-225.
11. Stainvas, I., Intrator, N. and Moshaiov, A., "Blurred Face Recognition via a Hybrid Network Architecture", Proceedings of the conference on Neural Computation in Science and Technology 99.
12. Schwenk, H. and Milgram, M., 1995. "Transformation Invariant Autoassociation with Application to Handwritten Character Recognition", Proceedings of the Seventh Conference on Neural Information Processing Systems, pp. 991-998.
13. Riddle, P., Segal, R. and Etzioni, O., 1994. "Representation Design and Brute-Force Induction in a Boeing Manufacturing Domain. Applied Artificial Intelligence, 8:125-147.
14. Riedmiller, M., and H. Braun, "A direct adaptive method for faster backpropagation learning: The RPROP algorithm," Proceedings of the IEEE International Conference on Neural Networks, 1993.