2002-05-17

Improving Speech Recognition Learning through Lazy Training

Tony R. Martinez
martinez@cs.byu.edu

Michael E. Rimer

See next page for additional authors

Original Publication Citation
Rimer, M., Martinez, T. R., and D. R. Wilson, "Improving Speech Recognition Learning through Lazy Training", Proceedings of the IEEE International Joint Conference on Neural Networks IJCNN'2, pp. 2568-2573, 22.

Authors
Tony R. Martinez, Michael E. Rimer, and D. Randall Wilson

Improving Speech Recognition Learning through Lazy Training

Michael E. Rimer
Brigham Young University
Computer Science Department
Provo, UT 84602, USA
mrimer@axon.cs.byu.edu

Tony R. Martinez
Brigham Young University
Computer Science Department
Provo, UT 84602, USA
martinez@cs.byu.edu

D. Randall Wilson
Fonix Corporation
180 West Election Road
Draper, Utah 84020
randy@axon.cs.byu.edu

Abstract – Multi-layer backpropagation, like most learning algorithms that can create complex decision surfaces, is prone to overfitting. We present a novel approach, called *lazy training*, for reducing the overfit in multiple-layer networks. Lazy training consistently reduces generalization error of optimized neural networks by more than half on a large OCR dataset and on several real world problems from the UCI machine learning database repository. Here, lazy training is shown to be effective in a multi-layered adaptive learning system, reducing the error of an optimized backpropagation network in a speech recognition system by 50.0% on the TIDIGITS corpus.

I. INTRODUCTION

Multi-layer feed-forward neural networks trained through backpropagation have received substantial attention as robust learning models for tasks including classification [17]. Much research has gone into improving their ability to generalize beyond the training data. Many factors play a role in their ability to learn, including network topology, learning algorithm, and the nature of the problem being learned. Overfitting the training data, caused through the use of an inappropriate objective function, is often detrimental to generalization. In applications such as speech recognition where even a small amount of error can be unacceptable it is important to generalize as well as possible.

This work introduces *word training* (WT), a novel technique for training speech recognition networks. Word training, inspired by *lazy training* [15], implements an objective function that seeks to directly minimize word classification error while discouraging overfitting. Lazy training performs successfully on a large OCR dataset and several problems selected from the UCI machine learning database repository, reducing their average generalization error over training of optimized networks by more than 60% using 10-fold cross-validation [17]. An extensively optimized, state-of-the-art backpropagation network achieves word recognition error of 0.12% on the TIDIGITS speech recognition corpus [11]. Word training performs markedly better than optimized standard backpropagation training, decreasing test set error by half, from 0.12% to 0.06%.

An overview of related work and a discussion of objective functions are provided in Section II. The lazy training and the word training algorithms are presented in Section III. Experiments and results are given in Section IV. Analysis and discussion are in Section V. Further work and conclusions are presented in Section VI.

II. RELATED WORK

The speech recognition problem is very complex and has received much attention in machine learning literature. Many learning models have been developed to cope with the difficulty of this problem. Often, neural networks have been utilized to provide a solution. However, neural networks are prone to overfit to the training data, which is detrimental to robust generalization. *Hidden Markov models* (HMMs) traditionally perform as well or better than neural networks at speech recognition [14]. Word training achieves results comparable to HMMs.

A. Critique of current training techniques

To generalize well, a learner must have a proper objective function. Most learning techniques incorporate an objective function of minimizing *sum-squared-error* (SSE). The validity of using SSE as an objective function to minimize error relies on the assumption that sample outputs are offset by inherent gaussian noise, being normally distributed about a cluster mean. For learning function approximation of an arbitrary signal, this presumption often holds. However, this assumption is invalid for classification problems, where the target vectors are class codings (i.e., arbitrary nominal or boolean values representing designated classes).

Cross-entropy (CE) assumes *idealized* class outputs (i.e., target values of zero or one for a sigmoid activation) [13] and is therefore more appropriate to classification problems. However, error values using SSE and cross-entropy have been shown [9] to be inconsistent with ultimate sample classification accuracy. That is, minimizing CE or SSE is not necessarily correlated to high recognition rates. Numerous experiments in the literature provide examples of networks that achieve little error on the training set but fail to achieve the best possible accuracy on test data [2, 18]. This is due to a variety of reasons, such as *overfitting* the data or having an incomplete representation of the data distribution in the training set. There is an inherent tradeoff between fitting the (limited) data sample perfectly and generalizing accurately over the entire population.

B. Shortcomings of search methodologies

More fundamentally, the above objective functions provide mechanisms that do not reflect the true goal of classification learning, which is to achieve high recognition rates on unseen

0-7803-7278-6/02/$10.00 ©2002 IEEE

data. In [9], a monotonic objective function, the *classification figure-of-merit* (CFM), is introduced for which minimizing error remains consistent with increasing classification accuracy. Networks that use the CFM as their criterion function in phoneme recognition are introduced in [9] and further considered in [5]. They are, however, also susceptible to overfitting. The question of how to prevent overfitting is a subtle one. When a network has many free parameters, not only is learning fast, but local minima can often be avoided. On the other hand, networks with few free parameters tend to exhibit better generalization performance. Determining the appropriate size network remains an open problem [8].

The problem of overfitting has received much attention in the literature. Methods of addressing this problem include using a holdout set to stop training early [20], cross-validation [2], node pruning [7, 8], and weight decay [21], among others. These techniques approach optimal solutions given the bias of standard backpropagation learning but do not consider possible enhancements to the bias itself. Node pruning seeks to improve accuracy by simplifying network topology, rather than alleviating the problems common to larger topologies, for example. Methods for overcoming problems in the inductive bias inherent to training with backpropagation generally involve forming network ensembles. Ensemble techniques, such as *bagging* and *boosting* [12], or *wagging* [3], are more robust than single networks when the errors among the networks are not positively correlated.

There is evidence that the size of the weights in a network plays a more important role to generalization than the number of nodes [4]. A simple method of reducing overfitting is to provide a maximum error tolerance threshold, d_{max}, which is the smallest absolute output error to be backpropagated. In other words, no weight update occurs for a given d_{max}, target value, t_k, and network output, o_k, if the absolute error $| t_k - o_k | < d_{max}$. This threshold is arbitrarily chosen to represent a point at which a sample has been sufficiently approximated. With an error threshold, the network is permitted to converge with much smaller weights [19].

III. WORD TRAINING METHOD

This work addresses overfitting exhibited by previous backpropagation solutions by applying *lazy training*, a conservative form of training, to the learning process (see Section III.C). Similar to CFM, it requires that a reduction in error correlate to increasing accuracy. However, CFM does not prevent weight saturation, which is often detrimental to accuracy [4]. Lazy training only backpropagates an error signal from output nodes that endanger classification accuracy. This approach allows the model to approach a solution more conservatively and discourages overfitting.

A. Phoneme training algorithm

Speech recognition is a complex problem, and a standard approach involves simplifying the problem by breaking it up into smaller, simpler ones. Word recognition is broken into the simpler problem of phoneme recognition. The signal is divided into small time slices called *frames* and features derived from each frame are input into the recognizer (see Figure 1). The recognizer then outputs the probability of each phoneme being uttered during that frame. Often, several contiguous frames are considered simultaneously, as in the multi-layer time-delay neural network in [10]. Phonemes are identified and combined through a proper linguistic model to derive words. However, derived features of a speech signal are often noisy and speaker dependent. Hence, it is difficult to achieve a satisfactorily high phoneme recognition rate at each frame and produce a reasonable solution.

Therefore, a decoder is stacked onto the phoneme recognizer to provide a more holistic solution. The decoder receives the outputs of the phoneme recognizer and combines the outputs over time to make a more educated guess as to what word or phrase has been spoken. Pairings of adjacent possible phonemes are validated or prohibited according to the linguistic model, and the overall most-likely sequence of phonemes is output as the response. Additional elements such as a lexicon can be incorporated into the decoder to constrain possible responses to produce more intelligent solutions. The decoder can be made even more sophisticated to combine probable words together into entire utterances according to a language model.

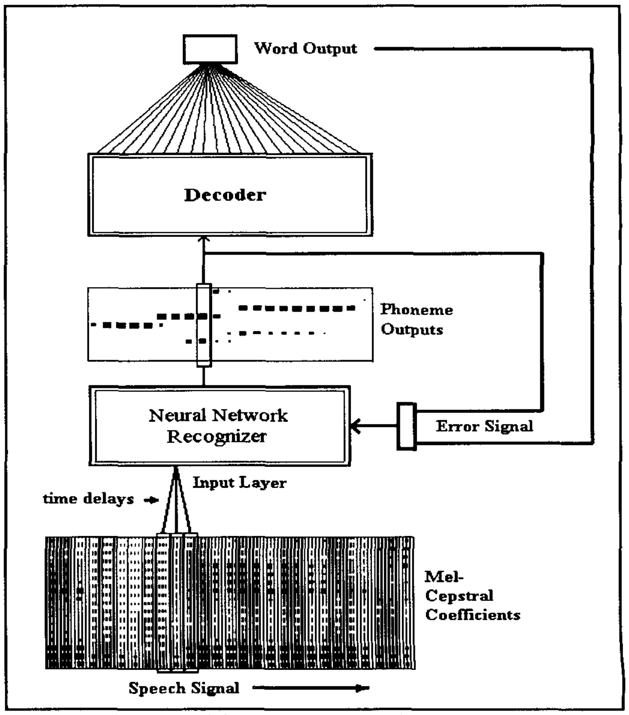

Figure 1. Word training system with neural network and decoder.

Phoneme training involves presenting a series of utterances to the network. Each utterance is divided into temporal frames and features derived from the signal that are input into the network. Each frame is labeled with the phoneme being spoken during that time. The network is often trained using backpropagation with a cross-entropy objective function.

B. Lazy training paradigm

Due to the reasons stated in Section II, a neural network classifier often overfits the training data. The tendency to overfit is further aggravated because labeled data points in this problem space are sparse. The problem is compounded since phonemes blend together, and it is problematic to label minute time slices accurately. It is therefore desirable to incorporate a recognizer that will overfit as little as possible in order to produce the highest possible generalization accuracy.

Overfitting a neural network is often equated with saturating the weights. It follows that overfit is reduced by letting the weights be as small as possible in the solution. This ideal can be approached through the following method.

For each frame considered by the recognizer during training, only those outputs that are credited with classification errors are updated through backpropagation. The result is training without idealized target outputs of 0 and 1, providing a learning mechanism that is reminiscent of constraint satisfaction and reinforcement learning, where the network outputs learn to interact with their (changing) environment (the behavior of the decoder based on the values of the output nodes). As this forces networks to learn only when explicit evidence is presented that their state is a detriment to classification accuracy, we have dubbed this technique *lazy training* (not to be confused with *lazy learning* approaches [1]). Backpropagation training often uses an objective function that tends to a *saturation* of the weights. That is, it tends to encourage larger weights in an attempt to output a value approaching the limits of 1 or 0. The ramifications of this are discussed further in Section V. Lazy training is biased toward simpler solutions, meaning that smaller weights (even approaching zero) can be used to provide an acceptable solution.

Two or more output nodes can in effect collaborate together to decide how learning is to proceed at any given point. More specifically, interaction among outputs allows a *dynamic error threshold* to be implemented. That is, when one output presents a sufficient solution in an area of the problem space, other outputs do not need to work at redundantly modeling the same local data. Consequently, they are able to specialize and break a complex problem up into smaller, simpler ones. This provides for a more conservative form of training that converges with smaller network weights, hence with less overfitting and greater generalization accuracy.

The lazy training methodology has been successfully utilized to significantly reduce error on OCR data and on several problems from the UCI repository of machine learning databases [6,15]. We implement it here for speech recognition to show further advantages of this training style. In past experiments, lazy training was performed on N separate single-output networks (one for each class in the problem). Here we show how it can successfully be used on a single N-output network. A single network provides a more compact, simpler, faster solution than many separate networks in learning a problem with several output classes.

Also, we illustrate that lazy training learns effectively when there is a level of indirection necessarily involved in arriving at a solution. In this case, while the network learns to output phoneme confidences, these confidences do not provide the actual solution, but are used by the phoneme decoder to derive the words spoken. High phoneme accuracy is therefore not necessarily the goal of training, but instead high word recognition rates. Word training (WT) is the name we give to training with an objective of directly increasing word recognition accuracy (possibly at the expense of phoneme accuracy). The method for deducing the network phoneme error from word error is presented in the following sub-section.

C. Word training algorithm

In word training the network decoder is involved in the training process. The decoder gathers the network outputs on all the frames of an utterance. When the decoder outputs a recognized word sequence, the output is compared against the target word sequence. If the output utterance matches the target, no error signal is propagated through the network at all (see Fig. 1, Error Signal). The network performs adequately within the system, and refraining from updating the weights discourages overfitting. When a discrepancy between the output and target exists, then the network weights are updated only on those time frames where the word errors occur.

Let N be the number of network output nodes (distinct class labels). Let o_k be the output value of the k^{th} output node of the network ($0 \le o \le 1$, $1 \le k \le N$). Let T designate the target output class for a given frame and c_k signify the class label of the k^{th} output node. For target output nodes, $c_k = T$, and for non-target output nodes, $c_k \neq T$. Non-target output nodes are called *competitors*. Let $o_{T\max}$ denote the highest-outputting target output node. Let $o_{C\max}$ denote the value of the highest-outputting competitor. The error, ε_k, back-propagated from the k^{th} output node of the network is defined as

$$\varepsilon_k \equiv \begin{cases} \tau_U - o_k & \text{if } c_k = T \text{ and } (o_{C\max} \ge o_{T\max}) \\ \tau_L - o_k & \text{if } c_k \neq T \text{ and } (o_k \ge o_{T\max}) \\ 0 & \text{otherwise} \end{cases} \qquad (1)$$

where τ_U and τ_L are upper and lower target values such that $0 \leq \tau_L < o_k < \tau_U \leq 1$. Thus, the target output generates an error signal only if there is some competitor with an equal or higher value than $o_{T\max}$, signaling a potential misclassification. Non-target outputs generate an error signal only if they have an output equal to or higher than $o_{T\max}$, indicating they are responsible for the misclassification.

The rate of convergence is partly dependent on the values used for τ_U and τ_L. Note that changing either τ is effectually equivalent to altering the learning rate. A τ closer to the current output value o_k implies a smaller error signal and will result in slower, but steadier convergence that more closely approximates the true error gradient than values near 0 or 1.

Word training of a network proceeds at a different pace than with standard backpropagation phoneme training. Training only the nodes that directly contribute to classification error of a word allows the model to relax more gradually into a solution, learning only as much as it needs to and thereby discouraging overfitting. This approach is reminiscent of training with an error threshold; however whereas a fixed error threshold causes training to stop at a pre-specified point, word training dynamically halts at the first possible point for a given frame at a given point in time. Weights are updated only through necessity. Without the decoder, a phoneme can be considered "learned" with any output value, providing competitors output lower values. Using a decoder, even more flexibility is possible, since the target output on a phoneme can be lower than its competitors and a word still be correctly identified.

Additionally, overfitting is minimized in a word trained network because outliers (noisy frames) have minimal detrimental impact to the decision surface's accuracy. This is because the target output is only required to output a value that is negligibly higher than the output representing the neighboring class, as illustrated in Figure 2b. This is in contrast to classical gradient descent training, where hard target values of 0 and 1 are required (translating to pushing the decision surface as far away as possible) even for outliers as illustrated in Figure 2a. Hence, in testing, samples close to the outlier belonging to the competing class (represented by the question mark) have a much better chance of being correctly classified.

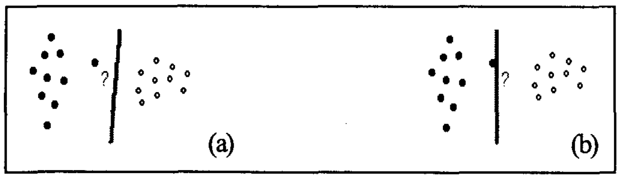

Figure 2: Overfit decision surface (a) and lazy-trained surface (b).

D. Enlarging the margin

When lazy training, it is common for the highest outputting node in the network to output a value only slightly higher than the second-highest-firing node (see Figure 3). This is true for correctly classified samples (above 0 in Figure 3), and also for incorrect ones (below 0). This means that most training samples remain physically close to the decision surface throughout training. An error margin, μ, can be introduced during the training process that serves as a confidence buffer between the outputs of target and competitor nodes. Under the sigmoid function, the error margin is bounded by [−1, 1]. For no error signal to be backpropagated from the target output, an error margin requires that $o_{C\max} < o_{T\max} - \mu$. Conversely, for a competing node k with output o_k, the inequality $o_k < o_{T\max} - \mu$ must be satisfied for no error signal to be backpropagated from k.

During the training process, μ can be increased gradually and might even be negative to begin with, not expressly requiring correct classification at first. This gives the networks time to configure their parameters in an even more uninhibited fashion. Then μ is increased to an interval sufficient to account for the variance that appears in the test data, allowing for robust generalization. The value of μ can also be decreased, and remain negative as training is concluded to account for noisy outliers (see Section V.A).

At the extreme value of μ equal to 1, lazy training becomes standard SSE training, with target values of 1.0 and 0.0 required for all positive and negative samples, respectively.

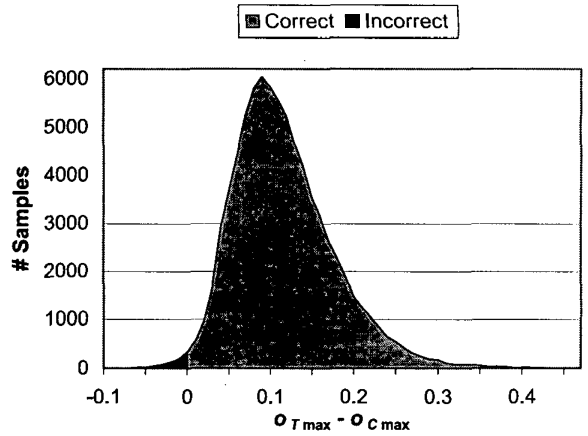

Figure 3: Network output margin of error after lazy training.

IV. EXPERIMENTS

The performance of phoneme versus word training models has been evaluated on a subset of the TIDIGITS data corpus consisting of over 17,000 utterances and sampled at 11 kHz, containing 50,000 spoken digits, partitioned into roughly 15,000 training samples, 1,000 validation samples and 1,000 test samples. Each sample is partitioned into 10 ms frames. The features generated for input to the network are standard mel-cepstral coefficients and their derivatives.

A. Parameters

We compared fully connected feed-forward network trained through on-line backpropagation maximizing cross-entropy on single frames against word-trained networks trained on utterances. In the experiments presented, networks contained a single hidden layer comprised of 50, 100, or 200 hidden nodes. Weights were initialized to small random values. The same initial weights were used for each training method. The learning rate began at 0.05 and a harmonic decay frequency of 5 epochs was used. In these tests a τ_U of 1 and τ_L of 0 were used for faster lazy training, and μ was 0. Training was halted after 150 epochs, many epochs after training error ceased to decrease.

The backpropagation network used is state-of-the-art. Its topology, objective function and learning parameters were optimized through extensive experimentation over a period of several years.

B. Results

Table I displays the test results of standard CE back-propagation training (BP) versus word training (WT). Accuracies are shown in percent. Highest column values are shown in bold, with the highest value for the other learning technique italicized. Note that high word accuracy is our prime goal. High sentence accuracy is a desired consequence, and phoneme accuracy is ultimately irrelevant.

TABLE I
RESULTS ON SUBSET OF TIDIGITS DATA CORPUS

Method, Hidden Nodes	**Phoneme**	**Base phoneme**	**Word**	**Sentence**
BP 200	**79.33**	**91.93**	*99.88*	*99.60*
BP 100	74.58	89.48	99.73	99.10
BP 50	66.40	84.66	99.71	99.00
WT 200	*51.50*	*76.04*	**99.94**	**99.80**
WT 100	47.96	74.03	99.82	99.40
WT 50	46.23	72.05	99.79	99.30

V. ANALYSIS AND DISCUSSION

Table I shows that networks generated through word training have the capability of cutting word error in half from 0.12% for standard phoneme backpropagation training to 0.06% for word training. These tests show that, although word training experienced much lower phoneme accuracy, word accuracy was increased and the amount of overfit was reduced (see Section V.C). The highest accuracies were achieved with a 200-node hidden layer. Larger networks show no further improvement. Interestingly, as smaller hidden layers are used, word and phoneme accuracy degrades more gracefully for word training than for CE training. When the training process concentrates directly on word accuracy instead of on learning phonemes, not directly responsible for word accuracy, more network parameters are free to learn a better solution.

A. Lazy training analysis

When networks are lazy-trained, instead of pushing the sample outputs to one end of the output range or the other, the vast majority remains spread out just slightly above the decision boundary. Output distribution is roughly gaussian, reflecting an actual gaussian data distribution, with a larger variance than appears from standard backpropagation, but only a fraction of the classification error. This suggests that the decision surface is much smoother and that network weights are not saturated.

Training set accuracy is largely preserved on the test set. Since the outputs learn together, their solutions are highly correlated and their solution transfers well to unseen data. Error is 50.0% less than with phoneme-trained networks, presenting a strong case for lazy training on complex data sets where backpropagation networks tend to overfit.

Lazy training also assists in the case of noisy data and inaccurate or uncertain phoneme labeling. In this case, the output representing the more accurate phoneme can fire roughly equal to the falsely labeled phoneme, rather than forcing it all the way down at 0. Even though the correct phoneme does not fire the highest value among the outputs, it fires nearly that high, enabling the decoder to more easily produce the correct answer.

B. Network complexity

The network outputs the majority of values at about 0.5. At first, it seems counter-intuitive that networks outputting only around 0.5 will generalize so well. Ordinarily, training networks together allows a classifier to become more complex, prone to overfitting. According to Occam's razor, adding parameters to a network, beyond the smallest correct solution for a given problem, can be a detriment to the generalization ability of the network. This is similar to the claim that a network with higher learning capacity tends to "memorize" noise in the data, which is an undesirable trait.

Recently, however, it has been illustrated how the number of nodes in a network is not as influential as the *magnitude* of the weights [4]. The topology, rather, serves more as a mechanism that lends itself to solving of certain problems, while the weights represent how tightly the network has fit itself to the (admittedly incomplete) training data distribution. Network complexity is further defined in [20] as the number of parameters and the *capacity to which they are used in learning* (i.e., their magnitude). In light of this, it is understandable why complex networks and lazy training, which allows networks to have small weights, perform so well together. Although the WT network has a high number of parameters, lazy training prevents further weight updates once frames are correctly classified and results in low complexity. Hence, the possibility of overfitting is reduced in the training process.

The networks used in our experiments had 130 inputs, 50, 100, or 200 hidden nodes and 199 output nodes, with 16,450, 32,900, and 65,800 weight parameters, respectively. The rows of Table II list the average magnitude of the weights in networks initialized with small random weights, during phoneme training, and during word training, respectively. The particular values shown are taken following the epoch with the highest word accuracy on the holdout set. The columns denote the average weight from input to hidden nodes, and from hidden to output nodes, respectively. The word-trained network has weights that are twice as large as the initial random values, while standard training produces weights four times larger. The lazy-trained network is a simpler solution than the network produced by standard backpropagation training.

TABLE II
AVERAGE NETWORK WEIGHTS

Method	Hidden Weights	Output Weights
Initial	.132	.150
Standard	.491	.567
Lazy	.280	.256

VI. CONCLUSION AND FUTURE WORK

Word training reduces overfitting in gradient descent backpropagation training, increasing the probability of discovering better solutions. Its advantages in word recognition over standard backpropagation phoneme training have been demonstrated in a speech recognition system. A word-trained network reduces word recognition error by half over an optimized backpropagation network on the TIDIGITS corpus, a large real world application.

For the word training nets presented, the learning parameters of the optimized backpropagation network were used. No attempt was made to optimize them for lazy training. Since standard backpropagation and lazy training vary significantly in their search technique, it would be expected that different parameter values would perform optimally with each objective function. Different settings on parameters such as τ_U, τ_L, and μ will be tested to further improve generalization accuracy. Word training will be applied to other problems that are broken into smaller pieces and then merged together, such as text recognition, using networks for OCR.

ACKNOWLEDGEMENTS

This research was funded in part by a grant from *fonix* Corp.

VII. REFERENCES

[1] David W. Aha, editor, *Lazy Learning*, Kluwer Academic Publishers, Dordrecht, May 1997.

[2] Andersen, Tim and Tony R. Martinez, "Cross Validation and MLP Architecture Selection", *Proceedings of the IEEE International Joint Conference on Neural Networks IJCNN'99*, CD Paper #192, 1999.

[3] Andersen, Tim and Martinez, Tony, "Wagging: A learning approach which allows single layer perceptrons to outperform more complex learning algorithms", *Proceedings of the IEEE International Joint Conference on Neural Networks IJCNN'99*, CD Paper #191, 1999.

[4] Bartlett, Peter L., "The Sample Complexity of Pattern Classification with Neural Networks: The Size of the Weights is More Important than the Size of the Network", *IEEE Trans. Inf. Theory*, 44(2), 1998, pp. 525-536.

[5] Barnard, Etienne, "Performance and Generalization of the Classification Figure of Merit Criterion Function", *IEEE Transactions on Neural Networks*, 2(2), March 1991, pp. 322-325.

[6] Blake, C.L. & Merz, C.J. (1998). UCI Repository of machine learning databases, http://www.ics.uci.edu/~mlearn/MLRepository.html. Irvine, CA: University of California, Department of Information and Computer Science.

[7] Castellano, G., A. M. Fanelli and M. Pelillo, "An empirical comparison of node pruning methods for layered feed-forward neural networks", *Proc. IJCNN'93-1993 Int. J. Conf. on Neural Networks*, Nagoya, Japan, 1993, pp. 321-326.

[8] Castellano, G., A. M. Fanelli, and M. Pelillo, "An iterative pruning algorithm for feed-forward neural networks", *IEEE Transactions on Neural Networks*, Vol. 8 (3), 1997, pp. 519-531.

[9] Hampshire II, John B., "A Novel Objective Function for Improved Phoneme Recognition Using Time-Delay Neural Networks", *IEEE Transactions on Neural Networks*, Vol. 1, No. 2, June 1990.

[10] H. Hild and A. Waibel. "Connected Letter Recognition with a Multi-State Time Delay Neural Network." Neural Information Processing Systems (NIPS-5), 1993.

[11] R. Gary Leonard and George Doddington. (1993). TIDIGITS speech corpus, http://morph.ldc.upenn.edu/Catalog/LDC93S10.html. Texas Instruments, Inc.

[12] Maclin, R and Opitz, D, "An empirical evaluation of bagging and boosting", *The Fourteenth National Conference on Artificial Intelligence*, 1997.

[13] Mitchell, Tom, *Machine Learning*. McGraw-Hill Companies, Inc., Boston, 1997.

[14] Rabiner, Lawrence and Juang, Biing-Hwang, *Fundamentals of Speech Recognition*. Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1993.

[15] Rimer, Michael E., Anderson, Timothy L. and Martinez, Tony R., "Improving Backpropagation Ensembles through Lazy Training", Proceedings of the *IEEE International Joint Conference on Neural Networks IJCNN'01*, pp. 2007-2112, 2001.

[16] Rimer, Michael, E., "Lazy Training: Interactive Classification Learning," Masters Thesis, Brigham Young University, 2002.

[17] Rumelhart, David E., Hinton, Geoffrey E. and Williams, Ronald J., "Learning Internal Representations by Error Propagation", Institute for Cognitive Science, University of California, San Diego; La Jolla, CA, 1985.

[18] Schiffmann, W., Joost, M. and Werner, R., "Comparison of Optimized Backpropagation Algorithms", *Artificial Neural Networks*, European Symposium, Brussels, 1993.

[19] Schiffmann, W., Joost, M. and Werner, R., "Optimization of the Backpropagation Algorithm for Training Multilayer Perceptions", University of Koblenz: Institute of Physics, 1994.

[20] Wang, C., Venkatesh, S. S., and Judd, J. S., "Optimal stopping and effective machine complexity in learning", in Cowan, J. D., Tesauro, G., and Alspector, J., editors, *Advances in Neural Information Processing Systems*, vol. 6, Morgan Kaufmann, San Francisco, 1994, pp. 303-310.

[21] Werbos, P., "Backpropagation: Past and future", *Proceedings of the IEEE International Conference on Neural Networks*, IEEE Press, 1988, pp. 343-353.