Dynamically Weighted Ensemble Neural Networks for Classification

Daniel Jiménez (djimenez@bluebonnet.uthscsa.edu)

The University of Texas Health Science Center at San Antonio
Department of Rehabilitation Medicine
7703 Floyd Curl Drive
San Antonio, TX 78284

Abstract

Combining the outputs of several neural networks into an aggregate output often gives improved accuracy over any individual output. The set of networks is known as an ensemble *or* committee. *This paper presents an ensemble method for classification that has advantages over other techniqes for linear combining. Normally, the output of an ensemble is a weighted sum whose are weights fixed, having been determined from the training or validation data. Our ensembles are weighted* dynamically*, the weights determined from the respective* certainties *of the network outputs. The more certain a network seems to be of its decision, the higher the weight.*

1. Introduction

Ensemble methods combine the outputs of several neural networks [1]. The output of an ensemble is a weighted average of the outputs of each network, with the ensemble weights determined as a function of the relative error of each network determined in training [1]; the resulting network often outperforms the constituent networks. There is a growing body of research into ensemble methods, for example, improvements in performance can result from training the individual networks to be decorrelated with each other [2, 3] with respect to their errors. We present a classification technique that differs in that the ensemble weights are determined dynamically, i.e., upon each propagation through the network, as opposed to statically, as part of the training algorithm. The weights are proportional to the *certainty* of the respective outputs. The certainty of a network output measures how close the output is to one or the other of the target values. For example, suppose we have trained two backpropogation networks to output 1 if the input is in a particular class and 0 otherwise. Suppose we present a previously unseen input to the networks and the outputs are 0.6 and 0.9 for the first and second, respectively. With a threshold for decision of 0.5, both outputs would lead us to conclude that the input is in the class, but the first network seems much less certain than the second.

2. Past Work

This section summarizes some of the past work done in ensemble methods, in terms of classification. Consider a population of n networks trained on a set $A = (x_m, y_m)$ of labeled instances of a binary classification problem.

2.1. The Naive Classifier

Let the function computed by the ith network be $f_i(x)$. If the networks are trained to output 0 or 1 for a negative or positive classification, we can use a threshold of 0.5 on the output of a network to decide the class for an instance of the problem. The naive approach usually used is use a cross validation set [4] $CV = (x_l, y_l)$ and pick the network f_{Naive} that minimizes the mean squared error (MSE) on CV. The mean squared error for each network is:

$$MSE[f_i] = E_{CV}[(y_m - f_i(x_m))^2]$$

This techniques throws out any knowledge contained in the other networks. Although f_{Naive} has the best overall performance on the cross validation set, some of the other f_i's may lead to correct classifications where f_{Naive} doesn't.

2.2. The Basic Ensemble Method

A simple approach to combining network outputs is to simply average them together. The basic ensemble method (BEM) output is defined:

$$f_{\text{BEM}} = \frac{1}{n}\sum_{i=1}^{n} f_i(x)$$

This approach by itself can lead to improved performance [1], but doesn't take into account the fact that some networks may be more accurate than others. It has the advantage that it is easy to understand and implement [5] and can be shown not to increase the expected error [5].

2.3. The Generalized Ensemble Method

A generalization to the BEM method is to find weights for each output that minimize the MSE of the ensemble. The general ensemble method (GEM) is defined:

$$f_{\text{GEM}} = \sum_{i=1}^{n} \alpha_i f_i(x)$$

where the α_i's are chosen to minimize the MSE with respect to the target function f (estimated using the cross validation set) and sum to 1. Define the error $\epsilon_i(x)$ of a network f_i as $\epsilon_i(x) = f(x) - f_i(x)$. Define the correlation matrix $C_{ij} = E[\epsilon_i(x)\epsilon_j(x)]$. Then we must find weights α_i that minimize

$$MSE[f_{\text{GEM}}] = \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j C_{ij}.$$

It can be shown [1, 5] that the optimal choice for the α_i is

$$\alpha_i = \frac{\sum_{j=1}^{n} C_{ij}^{-1}}{\sum_{k=1}^{n}\sum_{j=1}^{n} C_{kj}^{-1}}.$$

This method yields better performance than BEM, and can be shown to always give a better estimate than naive classifier [1]. However, this method depends on a reliable estimate of C and the fact that C is non-singular so that it can easily be inverted [1]. In practice, errors are often highly correlated, thus rows of C are nearly linearly dependent so that inverting C leads to significant roundoff errors. Some techniques to avoid this include ignoring networks whose errors are highly correlated with others [1], using specialized techniques for the inversion of near-singular matrices, and training the networks to be decorrelated with each other [6, 3].

3. Dynamic Ensemble Method

3.1. Certainty

If the output of a neural network $y = f_i(x)$ can be interpreted as the probability that an instance x is in a class, then as y approaches 1, we feel more certain that the instance is in the class. As y approaches 0, we become more certain that the instance is not in the class. Let us quantify this notion; define the *certainty* $c(y)$ of a neural network output:

$$c(y) = \begin{cases} y & \text{if } y \geq 1/2 \\ 1-y & \text{otherwise} \end{cases}$$

The certainty rises for outputs $y < 0.5$ as y falls, and for outputs $y \geq 0.5$ as y rises. We say one network output y_1 is *less certain* than another y_2 if $c(y_1) < c(y_2)$. Note that the certainty behaves symetrically with respect to positive and negative decisions; the certainty of an output of 0.1 is the same as that of an output of 0.9, but the decision they are certain about is different.

3.2. Dynamically Averaging Networks

If instead of choosing static weights derived from f_i's performance on a sample of the input space, we allow the weights to adjust to be proportional to the certainties of the respective network outputs, we might achieve better performance. Define the dynamically averaged network (DAN):

$$f_{\text{DAN}} = \sum_{i=1}^{n} \omega_i f_i(x)$$

where the ω_i's are:

$$\omega_i = \frac{c(f_i(x))}{\sum_{j=1}^{n} c(f_j(x))}$$

The ω_i's sum to 1, so f_{DAN} is a weighted average of the network outputs. The difference is that the weight vector is recomputed each time the ensemble output is evaluated, to try to give the best decision for the particular instance under consideration, instead of statically choosing weights that give an optimal decision with respect to a cross validation set. Each network's contribution to the sum is proportionate to its certainty. A value close to 0.5, for instance, would contribute very little to the sum while a very certain value of 0.99 (or 0.01) among many less certain values would dominate the sum. This method is similar to the idea of using agreement among a set of classifiers to obtain a measure of confidence in the decision [2] but the confidence level (certainty) of each classifier itself is used to obtain the final decision.

3.3. History

The dynamic ensemble method grew out of an *ad hoc* method used for locating anatomical markers on digitized three-dimensional shapes of below-the-knee (BK) amputees' residual limbs [7]. The naive method of picking the neural network having the lowest MSE on a validation set from a "crop" of networks was originally used. The results obtained were unsatisfactory since the method was less accurate than another technique involving finding areas of surface curvature [7] [8]. When the f_{BEM} method was used, accuracy improved. When the f_{DAN} method was used, accuracy matched and in some cases surpassed the surface curvature technique [7].

Table 1. Mean squared error and percent correct for the data sets

Classifier	f_{Naive}	f_{BEM}	f_{GEM}	f_{DAN}
MPTN MSE, % correct	0.06176, 92.47%	0.05168, 93.02%	0.05214, 93.12%	0.05281, 93.16%
HDFB MSE, % correct	0.05443, 93.68%	0.04621, 93.89%	0.04654, 93.89%	0.04718, 93.99%
Voting MSE, % correct	0.03983, 94.88%	0.03972, 95.12%	0.03997, 95.12%	0.04057, 95.12%
Cancer MSE, % correct	0.02078, 97.62%	0.02076, 97.62%	0.02076, 97.62%	0.02063, 97.62%

4. Results

Each of the four ensemble methods was implemented and tried on several data sets. We tested each of the four methods on data from these shapes and two other data sets acquired from the University of California at Irvine Machine Learning Repository, a database containing sets of data for use testing machine learning algorithms. The results presented in Figure 1 were obtained using a technique similar to ten-fold cross-validation. Each dataset was divided into ten pairs of training and testing sets. Six feed-forward networks with sigmoidal activation functions and four hidden units were trained for each training set, then tested as an ensemble for each method for the testing set. Each network was trained with standard backpropogation for 50 iterations with a learning rate of 0.3, then for 300 iterations with a learning rate of 0.1 using a dual PentiumPro 200MHz computer running Linux 2.0 and custom neural network software. The mean squared error and percentage correct were averaged over each of the ten testing sets and are reported in Figure 1.

4.1. The MPTN and HDFB data sets

MPTN stands for "midpoint of the patellar tendon." HDFB stands for "head of the fibula." Knowing where both of these anatomical features are is an important component in the computer aided design (CAD) and computer aided manufacture (CAM) of BK prostheses [7]. Typically, a prosthetist uses a laser-scanning imager or plaster cast digitizer to obtain the surface geometry of the patient's residual limb [7]. Since the resulting shape has no information in it about the underlying bones, prosthetists typically have to place markers for these anatomical features on the shape either in a CAD program or on the original shape itself in a way the digitizer can recognize. These placements are somewhat error-prone and tend to vary from one prosthetist to another. The goal of the research presented in [7] was to automatically recognize these features, eliminating human error and saving time for the prosthetist. This was the first step in an ongoing project at the University of Texas Health Science Center at San Antonio Department of Rehabilitation Medicine to automate the design and manufacture of lower-limb prosthetic devices. The MPTN and HDFB datasets represent thirty digitized shapes, each rotated through 96 distinct angles, so there are 2880 samples in each set. The patterns are labeled with a 0 if the feature is on the left side of the pattern, and a 1 if the feature is on the right side. Each pattern is a 16×16 greyscale map representing the roughly cylindrical shape "unwrapped" into two dimensions, so there are 256 inputs for each sample. Neural networks are trained on these data and, with some post-processing, the features can be accurately found for new shapes [7]. These shapes were used for testing each of the ensemble methods. Although each shape contributed 96 patterns to the data set, no pattern from the same shape was in both a training and the corresponding testing set for the ten-fold testing. The f_{DAN} classifier correctly identified 93.16% of the instances in the MPTN set and 93.99% instances in the HDFB set, outperforming each of the other ensemble techniques.

4.2. The Wisconsin diagnostic breast cancer data set

This data set, first used in [9], consists of 569 samples with 30 features computed from a digitized image of a breast mass describing characteristics of the cell nuclei. They are labeled as either benign or malignant; we used 0 to represent benign and 1 to represent malignant. The predictor in the original article has an estimated accuracy of 95.7%. Figure 1 shows the accuracy of each ensemble method; in each case, the accuracy was 97.62% with the lowest MSE from the f_{DAN} classifier.

4.3. The voting record of the U.S. Congress

Another data set from the UCI Repository contains the voting record on sixteen key issues for each congressman in the United States House of Representatives in the second session of the 98th Congress in 1984 [10]. Each sample is labeled with the congressman's party affiliation, either Republican or Democrat. The inputs to the neural network for each vote was 1 if the vote was "yea," 0 for "nay" and 0.5 for something that was neither "yea" or "nay" (for instance, "present"). The reported accuracy of the classifier in the original dissertation was from 90% to 95%. Each of the ensemble methods was tested with the ten-fold method. The f_{Naive} estimator classified 94.88% of the congressmen correctly as either Republican or Democrat. Each of the other methods classified 95.12% congressmen correctly.

5. Conclusions and future directions

The f_{DAN} classifier performs at least as well as the other ensemble methods in classifying the patterns in the data sets we tried. It adjusts its weights dynamically, giving it an edge over ensemble architectures whose weights are fixed as a part of training. We believe it is a valuable addition to the repetoire of techniques available to researchers. In the future, we would like to investigate the possibility of combining *a priori* knowledge like the correlation matrix with dynamically adjusted weights to see if this might increase the performance of the classifier.

References

[1] Perrone, M.P., and Cooper, L.N., "When networks disagree: ensemble methods for hybrid neural networks," *Neural Networks for Speech and Image Processing* by R.J. Mammone, ed., Chapman-Hall, 1993.

[2] Tumer, K. and Ghosh, J., "Classifier Combining: Analytical Results and Implications," Proceedings of the National Conference on Artificial Intelligence, (workshop on Integrating Multiple Learned Models), Portland, August 1996.

[3] Rosen, B.E., "Ensemble learning using decorrelated neural networks," *Connection Science*, 8, 3-4, pp. 373-384, 1996.

[4] Krogh, A. and Vedelsby, J., "Neural network ensembles, cross validation, and active learning," *Neural Information Processing Systems*, volume 7. MIT Press, 1995.

[5] Bishop, C.M., *Neural Networks for Pattern Recognition*, pp. 364-369, Oxford University Press, 1995.

[6] Tumer, K. and Ghosh J., "Error Correlation and Error Reduction in Ensemble Classifiers," *Connection Science, Special issue on combining artificial neural networks: ensemble approaches*, volume 8, numbers 3 & 4, pp 385-404, December 1996.

[7] Jiménez, D., Darm, T., Rogers, B., and Walsh, N. "Locating anatomical landmarks for prosthetics design using ensemble neural networks." Proceedings of the 1997 International Conference on Neural Networks, volume 1, pp. 81-87., Houston, TX, 1997.

[8] Chan, R.B., Rovick, J.S., and Childress, D.S., "Surface Curvature Analysis for Enhanced Computer-Aided-Design of Prosthetic Sockets," Proceedings of the 15th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 1993.

[9] Street, W.N, Wolberg, W.H. and Mangasarian, O.L. "Nuclear feature extraction for breast tumor diagnosis." IS&T/SPIE 1993 International Symposium on Electronic Imaging: Science and Technology, volume 1905, pp. 861-870, San Jose, CA, 1993.

[10] Schlimmer, J. C. "Concept acquisition through representational adjustment." Doctoral dissertation, Department of Information and Computer Science, University of California, Irvine, CA., 1987.