
by B. Samanta*, S. Bandopadhyay†, R. Ganguli†, and S. Dutta†

Synopsis

This paper investigates the performance of the neural network technique in a placer deposit situated at the Nome district in Alaska. Initially a single neural network model was constructed to estimate the gold grade. However, at a later stage, an ensemble model consisting of multiple networks was also constructed via the Adaboost algorithm. The reason behind the use of the ensemble model of the Adaboost algorithm was to examine if the ensemble model provided superior performance to the single neural network. As indicated by other studies, Adaboost did not have a better performance than a single neural network in this specific application. This may be due to the high noise inherent in the gold data used in this project. Consequently R² values of gold prediction are also poor for both the single and multiple networks.

Introduction

Nome placer offshore gold grade estimation is a challenging problem in light of extreme ore heterogeneity. However, this type of phenomenon is quite common in placer ore grade variation. Further, the characterization of spatial ore grade variability and subsequent ore grade modelling of the deposit becomes problematic due to the unavailability of adequate samples. Being located in the Arctic, the sampling is cumbersome and expensive. As a result, ore samples are taken at wide intervals. Therefore, any estimate of ore reserve might be associated with a low level of confidence. Recognizing this inevitability, ore reserve estimation should be carried out using a suitable technique

Geostatistics and neural networks are the two most popular techniques used for ore grade estimation. The efficacy of one method over the other has not been clearly established. In some applications, the superiority of neural networks over the ordinary kriging technique has been demonstrated (Samanta et al. 2003 b). On the other hand, Samanta et al. (2003 a), showed no improvement of neural network techniques over traditional geostatistical techniques. Use of the neural network for ore grade estimation is still an open area of research. Therefore, the emphasis of this study is put on the detailed investigation of the neural network in placer gold grade estimation. For this purpose, not only was a single neural network model used, but also the possibility of improved neural network performance using an ensemble of neural network models through the Adaboost algorithm was examined.

Neural network for ore grade estimation

A neural network model is an artificial intelligence technique which can mimic the operation of a human nervous system. Neural networks have capabilities that go beyond algorithmic programming and work very well for non-linear input-output mapping. It is this property of non-linear mapping by neural networks that can be used for ore grade estimation. Several researchers have applied neural networks for grade estimation, including Dowd and Sarac (1994); Singer and Kouda (1996); Yama and Lineberry (1999); and Koike and Matsuda (2003). In grade estimation, the spatial co-ordinates and other attributes can be used as the input nodes, while the grade attributes can be used as output nodes. If any spatial relationship is present in the deposit, the network is expected to capture the spatial continuity via the network connections between the input and the output nodes. The relationship between the input and the output is constructed through a set of interconnected neurons. There are weights on each of the interconnections and it is these weights that are updated during the training process to ensure that the inputs produce an output which is close to its actual value.

* Indian Institute of Technology, Kharagpur, India.
† School of Mineral Engineering, University of Alaska Fairbanks, Fairbanks, USA.
© The South African Institute of Mining and Metallurgy, 2005. SA ISSN 0038–223X/05.00 + 0.00. Paper received Mar. 2004; revised paper received Feb. 2005.
A comparative study of the performance of single neural network vs. Adaboost

value, with an appropriate training rule being applied to adjust the weights. Through the training process, the network learns from examples and in doing so acquires some capabilities for generalization beyond the training data. Therefore, given a set of inputs such as easting and northing coordinates, and water-table-depth, the network will be able to estimate the output grade attributes. The working principle of neural network has been discussed at length in a number of textbooks (Bishop, 1995; Hagan et al., 1995). However, for the reader's convenience, a brief overview of the neural network technique is presented in the Appendix.

There are some fundamental differences in the working principle of neural networks and geostatistics. In the neural network paradigm, spatial ore variability is captured through the non-linear input-output mapping via a set of connection weights. The neural network appears to work like a parametric non-linear global fitting model. Hence, the neural network is expected to provide improved performance if there is the presence of a non-linear spatial trend in the data variability. On the other hand, geostatistics works like a non-parametric local fitting model, which restricts the fit of the model to a local neighbourhood of data points. In geostatistics, a kriging model prediction is obtained by linearly weighting the data points nearby to the unknown point. The weights to the nearby data points are assigned based on their relative distances to the unknown point and the spatial correlation structure of the deposit. The spatial correlation structure of a deposit is determined through variogram modelling. Further, geostatistical techniques are based on the assumption of stationarity. The neural network does not rely on such an assumption.

Multiple networks and the Adaboost algorithm

Use of multiple networks and combining their outputs to get a single output is quite common in neural network literature (Hasan and Salamon, 1990; Baxt, 1992; Ali and Pazzani, 1996). This method is popularly known as the ensemble approach. There are several advantages to using an ensemble to model a single data-set. First, each neural network in the ensemble model follows more or less the true output mapping function. Conceptually, if one assumes that the output of an individual neural network of the ensemble consists of a true output plus a random error component with zero mean, then the combination of the outputs from the individual networks results in averaging of the random error components. Hence, it ensures reduction of the estimation error. Second, a single best network might be overfitted, meaning that the network behaves poorly with unseen data. An ensemble of networks might reduce the overfitting, by combining different networks with different architectures. Third, the input-output relationship represented by a set of data values with distinct natures might not be captured adequately by a single network. It is possible to train individual networks using the data having an identical nature from the entire set of data, and then combining the outputs of an individual network to get a final improved ensemble output.

Although the advantages of ensemble network were recognized by several researchers, it is believed that no attempt except by Dutta et al. (2003) was made to apply an ensemble network for ore grade estimation. Their study revealed that an ensemble network performed slightly better than a single best neural network. Furthermore, in their application of an ensemble network for ore grade estimation, they selected the different networks by changing the network architectures and the number of hidden neurons; while training data set was identical for each of the networks. In contrast, the objective of the Adaboost algorithm is to select different training data (patterns) sets for different networks, while fixing the network architecture for each of the networks.

Bragging, or bootstrap aggregating (Breiman, 1996), and Boosting (Schapire, 1990; Drucker et al., 1993; Freund and Schapire, 1996, Sharkey, 1999) is a popular technique used to create an ensemble network using different training data-sets. In bagging, each network is independently trained on ‘n’ samples picked randomly with replacement from the ‘n’ original samples of the training set. Each neural network is thereby trained on different but overlapping subsets of the original training set, and will therefore, give different predictions. Final prediction is the average of all the individual networks of the ensemble.

Under boosting algorithm, each network in the ensemble model is trained sequentially in step by step. In boosting, specifically in Adaboost algorithm, a neural network is first trained on n data values (samples) picked randomly with replacement from the n original samples of the training set. Note that since the samples are picked with replacement, a particular sample value may be selected for more than once. By extension, it is also possible that a sample value may not be selected at all. After the network is trained, all the data values in the original training set are passed through the network and the errors on individual data values are noted. At this stage, it is assumed that if the errors on some data values are high, then the network has not learned the characteristics of those data values. Therefore, those data values are highly weighted so that they are more likely to be picked as data values of the training set for the second network in the ensemble model. As a result, the second network will learn those characteristic features that were not learnt by the first network. Hence, as one proceeds in constructing individual networks, data values that are difficult for previous networks are more likely to appear in the training sets for successive networks in the ensemble. Thus, different networks in the ensemble are better for different characteristics of the data-set. To obtain a final prediction, individual networks are combined using the weighted median, whereby those networks that are more confident about their predictions are weighted more heavily.

This paper used the Adaboost algorithm for constructing the individual networks in the ensemble and the final prediction for ore grade was obtained by combining the output of the individual networks. A brief description of the Adaboost algorithm is presented here:

The basic idea behind the boosting/Adaboost algorithm lies in the PAC (probably approximately correct) learning model. In Adaboost, the selection probability of a sample (data value) in the training set for a particular network in the ensemble model depends upon the performance of the prior networks on that sample. According to Drucker (1997), the neural network in the Adaboost algorithm is first trained using the samples selected from the training set in such way
A comparative study of the performance of single neural network vs. Adaboost

that each sample has an equal chance of selection. Once the network is trained, each sample of the original training set is passed through this network so that a prediction for each sample is obtained. The error of each sample is then evaluated using a suitable error function. Generally, the mean squared error is used as an error function. However, other error functions such as, absolute error, absolute percentage error, and exponential error, can also be used depending upon the problem. It may be recognized that the errors for individual samples are scaled so that the maximum error does not exceed 1.0. This is normally done by dividing the individual error of a sample with the maximum error in the data-set.

Average error for the particular network can then be calculated using a weighted average error approach e.g. average error \( \bar{L} = \frac{\sum w_i L_i}{\sum w_i} \), where the weight \( (w_i) \) for a sample is assigned from the weighting distribution of the data-set (e.g. probability of selection of a sample at that stage) and the error \( L_i \) is a user-defined error function such as the squared error of sample i. Confidence \( (\beta_i) \) on the prediction for a particular network can be calculated using the following formula (for the \( t^{th} \) network):

\[
\beta = \frac{\beta_t \times L_t}{1 - \frac{1}{T} \sum L_t}
\]

Low \( \beta \) means high confidence in the prediction. Updating for the weighting distribution of the samples is done by using the following equation:

\[
\beta_t(L_t) \leftarrow \frac{1}{1 - \beta}\beta_{t-1} - L_{t-1}
\]

The process is continued till the average error for a particular network is below 0.5. In this way, multiple neural networks are generated sequentially.

In essence, for a particular input vector \( X \), \( T \) neural networks thus generated make predictions \( h_t, t=1,\ldots, T \) neural networks. Final prediction is obtained by combining the output of the \( T \) predictors using the following formula:

\[
H_{\text{final}} = \inf_{y \in Y} \left[ \sum_{j=1}^{c} \log \left( \frac{1}{\beta_j} \right) \right] \geq \frac{1}{2} \log \left( \frac{1}{\beta} \right)
\]

This can be re-stated as:

\[
H_{\text{final}} = \inf_{y \in Y} \left[ \sum_{j=1}^{c} \log \left( \frac{1}{\beta_j} \right) \right] \geq \frac{1}{2}
\]

The above expression is the weighted median approach and is also sometimes used in geostatistics (especially to remove outlier effects). When an output or prediction is desired for an input, each network \( h_t \) of the ensemble makes a prediction \( y_t \). These predictions are then ordered \( y^{(1)} < y^{(2)} < y^{(3)} ,\ldots,y^{(T)} \) based on their magnitude. Note that each of these networks has a confidence \( (\beta_t) \) associated with them. Starting with the first network (in the ordered networks), the normalized log \( (1/\beta_t) \) (Equation [4]) is summed till the cumulative sum exceeds 0.5. The network at which the summation exceeds 0.5 is chosen as the predictor.

Nome placer gold deposit

For this study, an offshore placer gold deposit of Nome mining district was selected. The Nome district is located on the south shore of Seward Peninsula at about latitude 64°30’ N and longitude 165°30’W. It is approximately 840 km west of Fairbanks and 860 km north-west of Anchorage. Gold is the main constituent of the deposit. Other valuable metals including iron, copper, bismuth, molybdenum, lead and zinc are also reported in Nome district. Because of the extent and richness of the Nome gold resources, the area was studied extensively, and geological, geophysical and geo-chemical characteristics of near-shore gold deposits are well documented in the published literature (Rusanowski, 1994).

For grade estimation, gold assay values are available in the form of a few exploratory drill-hole samples. The samples are not only scanty but sparse. They are also located on an irregular grid. For grade estimation purposes, the lease boundary is divided into 9 blocks—Coho, Halibut, Herring, Humpy, King, Pink, Red, Silver and Tomcod. The present study focused on the King block of the deposit.

This study is based on the 275 drill-hole information that is available from the King block. Drill-hole information contains northing and easting co-ordinates, water-table depth, gold values (in mg/m³) and other relevant information. The northing co-ordinate, easting co-ordinate and water-table depth were used as input variables, and the gold attribute was used as the output variable.

Gold distribution in the King block demonstrates some special features. Characterization of the spatial distribution of the data is therefore deemed essential for meaningful ore grade modelling. Towards this goal, a preliminary statistical analyses of the King data set was conducted through the spatial and histogram plots, and varyography study. Figure 1 presents the spatial grade distribution of the gold concentration. From spatial distribution of the data, it can be seen that low, medium and high values are intermingled with each other. No distinct patch of ore grade zone could be identified in the deposit. This suggests that the spatial distribution of gold concentration may have poor structural correlation. In fact, the omni-directional variogram (Figure 2) indicates a small regional component in the spatial distribution of gold concentration. A large proportion of spatial ore variation results from the nugget component. Also, it may be obvious that gold values are highly skewed towards the right side as it can be observed in the histogram plot shown in Figure 3. However, this is a common phenomenon for gold grade distributions. A high proportion of low grade values and a small proportion of high values are essentially identifiable in the histogram plot. This particular frequency distribution of the gold values poses a major difficulty in neural network modelling, especially for data division.

Data division for neural network modelling

The neural network is a very flexible model. Although, the neural network is extremely capable of recognizing any type of difficult relationship between input and output data, however, it is also susceptible to fitting with the noisy component of the data. The phenomenon of the fitting with the noise is called the overfitting of the neural network. An
A comparative study of the performance of single neural network vs. Adaboost

overfitting neural network model shows very good performance on the training data. However, when its performance is measured on the unseen data, it may show very poor performance. Therefore, the success of neural networks lies in their generalization, meaning how the network performs on unseen data. For this purpose, the entire data-set is divided into three sets: the training, the validation and the test set. The network is trained using the training data-set; however, its performance is monitored on the validation data set. The network training is stopped when the minimum error is reached on the validation data to restrict the overtraining of the model. The overtraining of the neural network generally leads to overfitting. Note that this method of training is popularly known as early stop training.
In order to build a legitimate neural network model, these three subsets of the data should be statistically similar. Conventional methods of data division randomly select the members of the samples into the three subsets. Since the data are highly skewed with a high proportion of low and a small proportion of high values, random division of the data might disproportionately pick only low and high values into these subsets. Different subsets may thus be generated with dissimilar statistical properties. An experimental study agrees with this possibility as 1/4th of randomly selected data divisions resulted in statistically unequal subsets. This means out of 4 random data divisions, at least one data division will be unacceptable. To address the problem, an approach of data segmentation combined with genetic algorithm was used in this study to generate statistically similar subsets of data (Ganguli et al., 2003 a). Detailed discussion of the data segmentation and the genetic algorithms can be found elsewhere (Samanta et al., 2003 b). A brief description is, however, presented here for completeness.

The entire data-set was first divided into three segments based upon the visual inspection of the histogram plot. The first segment consists of the low gold values ranging from 0–100 mg/m³, the second with medium values ranging from 100–300 mg/m³, and the third with high values ranging from 300–6072 mg/m³. After the data segmentation, a genetic algorithm was applied separately in each of the segments. Genetic algorithms employing evolutionary intelligences order the samples in such a way that the three data subsets are constructed with similar patterns, resulting in similar statistical characteristics of the subsets. The ordering of the samples for each segment was done in a way that the first 50% of samples are included into the training set, the second 25% into the validation set, and the remaining 25% are included into the test set. Hence, by processing the three segments through the genetic algorithms, statistically similar data subsets were reproduced (including having similar low, medium and high gold values). This resulted in 137 training samples, 67 validation samples, and 71 test samples.

Statistical properties of the three subsets generated above are presented through the histogram plots (Figure 4) and summarized in Table I. From the histogram plots, it is observed that three data subsets have an almost identical frequency distribution shape. From the table, it can be also seen that the statistical properties of the entire data-set and the three subsets are approximately identical.

Grade estimation

A feed forward multi-layer neural network shown in Figure 5 was used for the neural network modelling. This network has five slabs: one input slab, 3 hidden slabs and 1 output slab. Each slab consisted of varying types of activation functions. The advantages of using different activation functions are that many different characteristics of data could be well captured by this network. It is important to have this capability since a particular activation function might be effective only for some data patterns, and may not work as well for other data patterns. Slab 1 has three input neurons (easting and northing coordinates, and water-table depth), while slab 5 has a single output neuron for gold concentration. Slabs 2, 3 and 4 were used as hidden slabs. Slab 2 used the Gaussian activation function, Slab 3 used tanh functions, whereas Slab 4 used the Gaussian Complement activation function. The output slab used a logistic function. The hidden layer slabs 2, 3, and 4, each have 5 neurons. A MATLAB program applying the Levenberg-Marquardt back-propagation algorithm was developed and used for the neural network training purpose.

While applying the Adaboost algorithm, the first network was trained using randomly picked samples with replacement from the original training set, where each sample had an equal probability of selection in training of the network. A similar procedure was also followed for choosing the validation data for the first network. Training was stopped when the network reached the minimum mean squared error in the validation data set as described earlier. Then all the 137 samples in the original training data-sets were passed through this first network and the errors on individual samples were noted. Based on observed errors on the original training samples by the first network, the weighting distri-
A comparative study of the performance of single neural network vs. Adaboost

Distribution of the training data-sets was changed so that the difficult samples (higher mean squared error) received higher weight for the selection in the next round of training in the second network.

A second network was then trained using the new training set generated according to the modified weighting distribution of the samples. Since the statistics of the new training set were changed due to modification of weighting distribution, the weighting distribution for the validation data-set should also be changed accordingly so that the new validation data-set had similar statistics to that of the new training set. In order to accomplish this, the original 67 samples in the validation data set were also passed through the first network and the errors on validation samples were observed. The weighting distribution of the validation data-set was changed using the same procedure as the training set. Since the second network was trained using only the difficult data values from the first, the error rate generated by the second network on both the training and the validation data-sets was higher, while training and validation data sets were passed through this network. At this stage, the average weighted error (mean squared error) was higher than 0.5. Therefore, the Adaboost algorithm stopped after two iterations resulting in two neural networks in the ensemble model. The final estimate was produced by combining the outputs of these two networks using the weighted median approach described earlier.

Results

Table II presents the results of the boosting algorithm and the single best neural network on the test data used for prediction purposes. For comparison of the results, four statistical indices, bias (mean error), mean absolute error, \( R^2 \), and mean squared error were used. Bias is the average error of estimation and an indicator for over-estimation or under-estimation. Mean absolute error measures the average absolute deviation between actual and the estimate. \( R^2 \) is an evaluator for the percentage of variation of the actual data explained by the predicted data. Mean squared error is the mean of squared deviation between the actual and the estimation. It is a measure of accuracy for estimation, which takes into account of both the bias and the error variance. It is observed from the table that the single neural network and the multiple networks of the Adaboost algorithm performed almost equally for the gold data-set.
Another alternative approach was also explored using a modified weight update criterion. Initially, each sample was assigned equal weight as it is usually done for training the first network. However, the second network was trained using the samples selected from the modified weight update rule. In the modified weight update rule, the samples were ranked in ascending order according to their squared errors. This means that the largest error in the sample set received the highest rank. Weight of a sample (selection probability) was calculated according to its rank such that the sum of weights for all the samples in the set became 1.0. Successive networks were trained by selecting the samples according to this update rule. This approach does not require any convergence criterion to be set for the Adaboost algorithm. Instead, an ensemble of multiple neural networks was selected on an ad hoc basis (i.e. the number of iterations of neural networks was not based on the Adaboost algorithm but was decided arbitrarily). Experimenting with various number of iterations, an ensemble of 8 networks produced the best result using this approach. The results of this approach are presented in Table II (Adaboost_modi). It is seen that this approach performed slightly better than the original boosting algorithm. The bias and the mean squared error are reduced.

Figures 6, 7 and 8 present the scatter plots of the actual vs. predicted values using various methods (Adaboost, Adaboost_modi, and single neural network) for the test data set. From the figures, it can be seen that none of the methods was able to predict extreme gold values.

<table>
<thead>
<tr>
<th>Statistics</th>
<th>Adaboost</th>
<th>Adaboost_modi</th>
<th>Single network</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bias</td>
<td>35.20</td>
<td>1.63</td>
<td>25.43</td>
</tr>
<tr>
<td>Mean absolute error</td>
<td>341.30</td>
<td>337.19</td>
<td>351.20</td>
</tr>
<tr>
<td>Mean squared error</td>
<td>345.370</td>
<td>336.580</td>
<td>351.240</td>
</tr>
<tr>
<td>$R^2$</td>
<td>.19</td>
<td>.20</td>
<td>.19</td>
</tr>
</tbody>
</table>

Table II
Comparative performance of Adaboost algorithm and single neural network on test data set

Figure 6—Actual vs. predicted for gold concentration using neural network

Figure 7—Actual vs. predicted for gold concentration using the Adaboost algorithm
A comparative study of the performance of single neural network vs. Adaboost

Also obvious from this analysis is that the advantages of the boosting algorithm over the single network are not clearly evident in this study. One of the reasons for underperformance of the Adaboost algorithm might be the inherent spatial nature of the gold data-set. The gold data values are highly variable, and there is very little regional (spatial) component observed in the gold data-set as projected in the variogram model. The regional component is exceedingly subdued by the nugget component. Therefore, the first network in the Adaboost algorithm is suitable enough to capture the spatial continuity, however small, observed in the data. The so-called difficult data on which successive networks are trained perhaps bear no additional information, except that they represent the noise component in this case. As a result, the successive networks are not contributing anything, perhaps only mapping the noise component. Therefore, the benefit of multiple networks in the Adaboost algorithm is not predominant in this study.

Furthermore, the $R^2$ values of the predictions for the neural network models (single and ensemble) were substantially poor. Given the nature of gold variation, however, where the spatial distribution of gold values had a significant contribution from the nugget effects, thus, poor $R^2$ values are reasonable.

Summary

In this paper, the applicability of the neural network for ore grade estimation was investigated in the Nome placer gold deposit A feed-forward neural network was used for this purpose. The neural network was trained using early-stop training to acquire a good generalization property. Prior to neural network modelling, the data set was divided into training, validation and test subsets using an integrated approach of data segmentation and genetic algorithm, so that each of the subsets had similar statistical properties. An attempt was also made to construct an ensemble model of multiple networks using the Adaboost algorithm to verify any improved performance of ensemble approach. In applying the Adaboost algorithm, multiple neural network outputs were combined using a weighted median approach to obtain the final output grade. Subsequently, the performance of multiple networks over a single network was evaluated. Performance evaluation of multiple networks constructed by the Adaboost algorithm over a single neural network revealed that both types of modelling approach worked equally well for this specific case study. It is also revealed that the performance of the neural network for this particular case study is generally poor. This might be due to the low spatial correlation structure present in the gold data.

References

A comparative study of the performance of single neural network vs. Adaboost

KORE, K. and MATSU DA, S. Characterizing Content Distributions of Impurities in a Limestone Mine Using a Feedforward Neural Network, Natural Resources Research, 2003, vol. 12, no. 5, pp. 229–233
RUSANOWSKI, NOME Offshore Gold Placer Project, Nova Natural Resources Corporation, 1994, pp. 292

Appendix

Neural network operational mechanism

Neural networks process information in interconnected layers. A layer is simply a group of elements designated as neurons. There is the input layer consisting of the inputs, the one or more hidden layers consisting of a suitable number of neurons, and the output layer consisting of the outputs. Thus, a neural network has three or more layers (typically three or four). A typical network architecture is presented in the Figure A.1 for ease of explanation. The network present in the figure has 3 layers. The inputs (elements in the input layer) are first manipulated to determine the value of the elements of the hidden layer. It should be noted that while the input layer and the output layer have a fixed number of elements for a given problem, the number of elements in the hidden layer is arbitrary.

Each input $x_i$ is assigned a weight $w_j$ and connected to element $j$ of the hidden layer. Each input is connected to each node of the hidden layer just as each node of the hidden layer is connected to each node of the output layer.

Mathematically, each hidden layer element is equal to the sum of the product of the inputs and their assigned weights. The output of each hidden layer element is obtained by applying a function $f_j$ to the sum (i.e. the input of the element). As in the case of the input layer, this output, in turn, is weighted ($w_{jk}$) and sent as inputs to each of the output elements. The final output, i.e. each output element, $y_k$, is obtained by applying a function $f_k$ to the sum of the inputs from the hidden layer. Therefore,

$$y_k = f_k \left( \sum \sum w_{jk} f_j \left( \sum w_{ij} x_i \right) \right) \quad [A.1]$$

In the present case, as there is only one output, the generic output will be referred to as $y_k$ rather than $y_j$. The functions $f_j(*)$ and $f_k(*)$ are typically sigmoidal as shown below:

$$f_x = \frac{1}{1 + e^{-x}} \quad [A.2]$$

However, other activation functions like tanh, Gaussian, Gaussian-complementary might be used as described in the text.

Learning

From the preceding discussion, it becomes apparent that the obtained output depends on the weights of the interconnections. Therefore, it is paramount that these weights be determined correctly so that they are representative of the property. The process of determination of weights is called ‘learning’ or ‘training’. Learning is a process by which the neural network adjusts its weights to reflect the change in the environment, i.e. during learning, it assimilates the data patterns. Known input-output examples are used for this purpose, i.e the sample data-set for the property is used training. Assume that there are $n$ sample points. The network starts learning with sample point number one. It begins with arbitrary weights; typically, all weights are set to random

![Figure A1—A neural network architecture with one input, one hidden and one output layer](image-url)
A comparative study of the performance of single neural network vs. Adaboost

numbers with a mean of zero and a standard deviation of one. In most cases, it doesn't really matter what the initial weights are. Using the initial weights, the network computes the output \( y \) as described previously. Depending on the output, weights are adjusted based on their contribution to the error. This process of propagating the effect of the error onto all the weights is called back propagation. Using the adjusted weights, the network then computes an output for sample number two. The weights are adjusted once again to reflect the error. Ultimately, it goes through all the samples, adjusting the weights on each occasion. When a neural network is done training on a sample set, it is said to have completed an epoch.

Typically, the weights after an epoch are still unsatisfactory. To make the network satisfactory, the learning/training process is restarted with the initial weights being the final weights of the previous epoch. When the error for an epoch is deemed satisfactory, the network is ‘trained’.

**Back propagation**

For correcting the weight \( w_j \) connecting the hidden layer node \( j \) to the output, assume that the error gradient (a measure of the contribution of the node towards the total error) for that node \( (e_j) \) was known. The correction in the weights should then be proportional to this measure. If \( w_j \) is the weight connecting the jth node to the output, the corrected weight is given by:

\[
w_j(\text{new}) = w_j(\text{old}) - \Delta w_j \tag{A.3}
\]

where, the correction \( \Delta w_j \) is given by:

\[
\Delta w_j = e_j \cdot I_j \tag{A.4}
\]

where, \( I_j \) is the total input to the node, \( e_j \) being the gradient. This method is known as the gradient descent method. A learning rate \( (l) \) is incorporated to control the amount of correction. This is shown below:

\[
w_j(\text{new}) = w_j(\text{old}) - lr \cdot e_j \cdot I_j \tag{A.5}
\]

\( e_j \) for a layer \( s \) can be represented by:

\[
e_j^s = f'(I_j^s) \cdot \sum_k e_{kJ}^{s+1} \cdot w_{kj}^{s+1} \tag{A.6}
\]

Starting with the output layer, the equations above are used iteratively to correct all weights. The advantage in using the sigmoidal function is that its derivative is easily computed (making programming easier):

\[
f'(x) = f(x) \cdot (1 - f(x)) \tag{A.7}
\]

Therefore, \( \text{(A.6)} \) reduces to:

\[
e_j^s = I_j^s \cdot (1 - I_j^s) \cdot \sum_k e_{kJ}^{s+1} \cdot w_{kj}^{s+1} \tag{A.8}
\]

**Application of the obtained model**

Once the network is trained, the weights are no longer modified, i.e. the network is available for prediction. The various inputs are fed into the network and predictions made based on the final weights. ◆