ELSEVIER

Neurocomputing 38–40 (2001) 1051–1057

NEUROCOMPUTING

www.elsevier.com/locate/neucom

Minimizing memory loss in learning a new environment

Khalid Al-Mashouq[a,*], Yaser Abu-Mostafa[b], Khaled Al-Ghoneim[a]

[a]*EE Department, King Saud University, P.O. Box 800, Riyadh 11421, Saudi Arabia*
[b]*Caltech, Pasadena, CA, USA*

Abstract

Human and other living species can learn new concepts without losing the old ones. On the other hand, artificial neural networks tend to "forget" old concepts. In this paper, we present three methods to minimize the loss of the old information. These methods are analyzed and compared for the linear model. In particular, a method called network sampling is shown to be optimal under certain condition on the sampled data distribution. We also show how to apply these methods in the nonlinear models.

Keywords: Memory loss; Catastrophic interference; Merging networks

1. Introduction

Living species have an inherent capacity to adapt, through learning, to new environments. Learning is an ongoing process that extends over the whole period of life [1]. The learning of a new concept usually does not result in "catastrophic interference" [3]; i.e. it does not lead to a great loss of the old knowledge. In artificial neural networks, this is usually different. It is of a great interest to study why biological systems are more efficient in this aspect than artificial ones.

Normally in artificial neural networks, we are given a function or a "concept" to be learned from examples. The examples are either ready as a batch or received online during training. In both cases, the training period is limited and once completed, the adaptable parameters (or weights) are fixed. After that, any new data set cannot be incorporated into the network without losing, or forgetting, a considerable portion of the old concept.

* Corresponding author. Fax.: + 966-1-480-7350.
E-mail address: mashouq@ksu.edu.sa (K. Al-Mashouq).

0925-2312/01/$ - see front matter
PII: S0925-2312(01)00400-3

Starting training from scratch with the old and new data is not practical mainly because

- In many real world problems, training takes exorbitant amount of time. Repeating the training on the new combined data set would consume even longer time.
- In some occasions, the old data is not available or expensive to store.

The main objective of this paper is to study methods to incorporate new knowledge in a pre-trained neural network with minimal distortion to the old knowledge. We can formalize our problem as follows. We are given a network, N_1 that has been trained with a fixed training set (X_1, Y_1) obtained from a certain environment. X_1 is the set of input patterns and Y_1 is the set of associated target values. A new training set is now available (X_2, Y_2). This new set is coming from an environment "correlated" with the old one. For example, the first set could be male speech samples and the second one could be female speech samples. Another example is credit card fraud data where the original data were comes from the last 10 years and the new data correspond to the current year. The question is: How to incorporate the new data into network N_1 with minimal loss of the knowledge about the old data?

2. Three possible approaches

Here we consider three approaches to reduce the loss of the old knowledge.

1. Create a new network N_2 and train it with the new data set (X_2, Y_2). Then combine the outputs of the two networks possibly with different weights.
2. Train the old network N_1 using the new data set (X_2, Y_2) with the condition that the old network weights are minimally disturbed. This should be similar to weight decay [2], but instead of minimizing the norm of the weight, we minimize the distance between the output of the old network and the new network due to the new data.
3. Generate a new data set (X_3, Y_3) that represents, as closely as possible, the knowledge in N_1. To do that we propose the technique of sampling the old network N_1. Sampling means that we generate, at random, data points X_3, apply X_3 to N_1 and assign the output to Y_3. In this scheme, care must be taken to sample X_3 with the appropriate distribution. Based on the VC dimension principle [5], the generalization ability of N_1 will affect the accuracy of the new target data Y_3. The number of points in the sampled data can be chosen to reflect our confidence in the new data in comparison with the old one.

We will analyze these three approaches mathematically for the linear neural network model. We also show when the equivalence could occur between these three apparently different approaches. Although our mathematical derivation is confined to the linear model, we will point out possible links to the nonlinear neural network models. Simulation examples with the linear and nonlinear models are provided to illustrate these links.

3. Linear model

Consider the following simple case. We are given (X_1, Y_1) a training set of n_1 samples. If we use a linear 1-layer neural net, then the weight vector w_1 that achieves the minimum mean square error, $E_1\{(x^{\mathrm{T}}w - y)^2\}$, is given by [6]

$$w_1 = R_1^{-1}P_1, \tag{1}$$

where $R_1 = E_1\{xx^{\mathrm{T}}\}, P_1 = E_1\{xy\}$, x the input vector, y the desired output label and E_1 represents the estimated expectation over (X_1, Y_1) data set distribution. When we have both (X_1, Y_1) and (X_2, Y_2) available, we can optimally obtain the new weights from (1) but now the expectations are estimated over the two sets. For simplicity, the number of samples in the second data set n_2 is assumed equal to n_1. In other words, we are giving equal weights to the two data sets. Therefore, the optimal weight vector, w^*, is given by

$$w^* = (R_1 + R_2)^{-1}(P_1 + P_2), \tag{2}$$

where $R_2 = E_2\{xx^{\mathrm{T}}\}, P_2 = E_2\{xy\}$ and E_2 represents the estimated expectation over (X_2, Y_2) data set distribution.

Approach 1: Output combining. If (X_1, Y_1) is not available, or we do not want to train from scratch, we can use the first approach. We apply (1) to find w_2 using (X_2, Y_2) data set as

$$w_2 = R_2^{-1}P_2. \tag{3}$$

Now, the output of the two nets can be combined, or equivalently we find a merged network with weight, w, given by

$$\begin{aligned} w &= w_1 + w_2 \\ &= R_1^{-1}P_1 + R_2^{-1}P_2. \end{aligned} \tag{4}$$

Note that the equivalence between this formula and outputs combining is valid only in the linear model. For nonlinear neural network models, such as the multilayer network, this is in general not true. In other words, merging the two networks (corresponding to the old and new data sets) in one network of the same size does not have the same effect as combining their outputs.

Approach 2: Weight constraint. In the second approach, we want to obtain a weight vector, w which minimizes the following objective function

$$E_2\{(x^{\mathrm{T}}w - y)^2 + \alpha(x^{\mathrm{T}}w - x^{\mathrm{T}}w_1)^2\}, \tag{5}$$

where α is a normalization factor. This objective function is intended to make w represent the new function while closely resembling the old one. Increasing the value of α makes the weight vector more biased towards w_1. Here again for simplicity we assume $\alpha = 1$. One can show that to minimize the objective function in (5), w has to be

$$w = R_2^{-1}(P_2 + R_2 w_1). \tag{6}$$

If we substitute (3) into (6) we obtain $w = w_1 + w_2$, the same output combining approach. We insist that this equivalence is limited to the linear model.

Approach 3: Network sampling. The third approach requires that we generate a third data set (X_3, Y_3) by sampling as described above. The issue of the distribution of X_3 will be discussed later. The target value associated with x is given by $y = x^{\mathrm{T}} w_1$. Thus we have two data sets (X_2, Y_2) and (X_3, Y_3). The optimal weight vector is obtained similar to (2) as

$$w = (R_2 + R_3)^{-1}(P_2 + P_3). \tag{7}$$

Note that

$$P_3 = E_3\{xx^{\mathrm{T}} w_1\} = R_3 w_1 = R_3 R_1^{-1} P_1$$

or

$$w = (R_2 + R_3)^{-1}(P_2 + R_3 R_1^{-1} P_1). \tag{8}$$

It is obvious that if one manages to make the sampling points distribution of X_3 resemble that of X_1, then (8) will yield the same optimal solution in (2). If one chooses $X_3 = X_2$, then the solution in (8) will be equivalent to the weight constraint in (6). Similarly, one can verify that the combining output (and weight constraint) approach solution in (4) will converge to the optimal solution as R_2 gets closer to R_1.

4. Examples with linear and nonlinear models

Here we present two examples to demonstrate the use of the three approaches given above. In the first example, we consider the linear model solution. The second example shows the applicability of the three approaches to the multilayer neural network. Note that the first and second approaches are equivalent only in the linear model.

4.1. Linear model example

In this example we consider two equally probable classes labeled $y = +1$ and -1. The input vector $x \in \mathbf{R}_2$, has the following distributions $p(x \mid y = 1)$ and $p(x \mid y = -1)$, which are 2-D uncorrelated Gaussian with unity variance and mean $(1, 1)^{\mathrm{T}}$ and $(-1, -1)^{\mathrm{T}}$, respectively. Fig. 1 shows the centers of these two Gaussian distributions. Applying (1), we can easily find the (normalized) weight vector $w_1 = (1, 1)^{\mathrm{T}}$. If we receive a new data set (X_2, Y_2) with the same Gaussian covariance but centered at $(-1, 1)$ for the label $y = 1$ and centered at $(1, -1)$ for $y = -1$, again using (1) we find $w_2 = (-1, 1)^{\mathrm{T}}$. The optimal weight for the two combined data sets is obtained using (2), which is $w^* = (0,1)^{\mathrm{T}}$.

It is interesting here to note that the output combining approach results in $w = w_1 + w_2 = (0,2)^{\mathrm{T}}$ which is equivalent to the optimal solution. Moreover, if we use the network sampling approach as in (8) we get $w = w^*$ whenever $R_3 = R_1$ or $R_3 = R_2$. In fact, one can find much more choices of R_3 which yield the same optimal

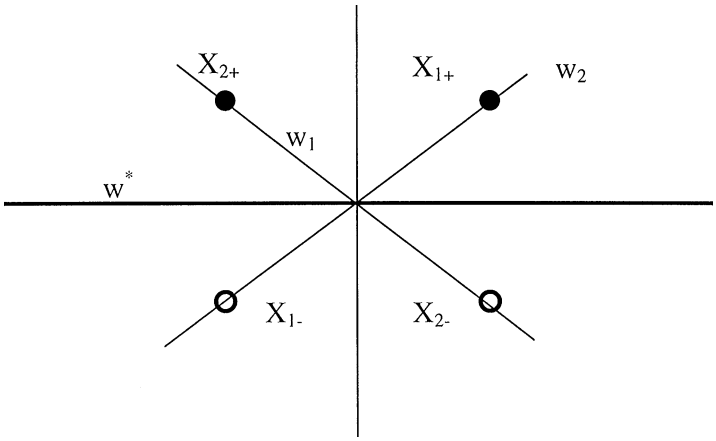

Fig. 1. Separation lines for the old data set, w_1, new data set, w_2, and optimal solution w^*.

solution. This eases the condition on X_3 distribution and makes the sampling task concerned only about R_3 and its "class" of choices.

4.2. Multilayer neural network example

All of the three approaches can be used to the nonlinear models such as the multilayer neural network. In the output combining approach we combine the output of the two networks, N_1 and N_2, after trained with (X_1, Y_1) and (X_2, Y_2), respectively. For the weight constraint approach, we have to do a slight adjustment to the training algorithm such as the backpropagation algorithm. The error term should be

$$e = (f_2(x) - y) + \alpha(f_2(x) - f_1(x)),$$

where $f_1(x)$ and $f_2(x)$ are the output of N_1 and N_2, respectively, due to the input x. For the network sampling approach, we perform a standard training procedure with a mixture of (X_1, Y_1) and (X_3, Y_3), where Y_3 is the output of N_1 due to X_3.

We chose the ionosphere data set, as explained in [4], to perform our experiments. This data set contains 351 samples corresponding of radar return from the ionosphere. Each sample has 34 attributes. The target is either bad return (no free electrons) or good return (free electrons exist). We select the first 100 samples (with their class type) as (X_1, Y_1) and the last 100 samples as (X_2, Y_2). The remaining samples (X_t, Y_t) are used for testing. A 2-layer neural network with 3 hidden nodes is chosen as our basic structure.

First we obtain N_1 which is trained with (X_1, Y_1). Upon testing with (X_t, Y_t), N_1 yields 11% misclassification rate. Similarly, N_2 results in 12.58% misclassification. When we train with both (X_1, Y_1) and (X_2, Y_2) we get 7.9% misclassification. The output combining of N_1 and N_2 results in 12.58% error, while the weight constraint approach achieves 10% error. When we sample N_1 using $X_3 = X_1$, and train with the two data sets (X_2, Y_2) and (X_3, Y_3), we obtain 7.9% classification error. This suggests

that the weight sampling approach could out perform the other two approaches. Another advantage of this approach is that it gives flexibility in putting more emphasis on the old data set by incorporating more training samples from the old network.

5. Discussion and conclusions

In this paper, we discussed the problem of training neural networks without losing the old knowledge stored in them from previous training data. Given a trained neural network and a new training set, we discussed three possible approaches training the network to reflect both the old and new data. For the linear case, we showed that the first two approaches are equivalent. Namely, combining the outputs of the old network with a fresh network trained on the new data is the same as training a fresh network such that the error is a mixture between satisfying the new targets and being close to the output of the old network. The third approach depends on selecting a third dataset of input samples, and setting their target to the output of the old network. This approach will yield optimal solutions with the careful choice of the sampling distribution. We demonstrated our results using two examples. Our future work will concern methods of sampling data for the third approach. Without having the original training set, the only knowledge about the distribution can be inferred from the old network. We plan to compare the targets of the new data with the outputs of the old network to predict the sampling distribution. The nonlinear case will be also be the subject of our future work.

References

[1] C.R. Gallistel, The Organization of Learning, MIT Press, Cambridge, MA, 1990.
[2] S.J. Hanson, L.Y. Pratt, Comparing biases for minimal network construction with backpropagation, in: D. Touretzky (Ed.), Advances in Neural Information Processing Systems, Vol. 1, Morgan Kaufmann, San Diego, CA, 1989, pp. 177–185.
[3] N.E. Sharkey, A.J.C. Sharkey, 1994. Understanding catastrophic interference in neural nets, Department of Computer Science Research Report CS-94-4, University of Sheffield, UK.
[4] V.G. Sigillito, S.P. Wing, L.V. Hutton, K.B. Baker, Classification of radar returns from the ionosphere using neural networks, Johns Hopkins APL Tech. Digest 10 (1989) 262–266.
[5] V. Vapnik, A. Chervonenkis, On the uniform convergence of relative frequencies of events to their probabilities, Theory Probab. Appl. 16 (1971) 264–280.
[6] B. Widrow, S. Stearns, Adaptive Signal Processing, Prentice-Hall, Englewood Cliffs, NJ, 1985.

Khalid Al-Mahsouq received B.Sc. (Hons.) and M.S. degrees from King Saud University, Riyadh, in 1983 and 1986, respectively. He received the Ph.D. from University of Southern California, Los Angeles, CA, in 1991. In December 1991 he joined the Department of Electrical Engineering of King Saud University, where he is currently an associate professor. His research interests include neural networks, signal processing and mobile communications.

Yaser Abu-Mostafa received a B.Sc. degree with honors from Cairo University in 1979, and MSEE from the Georgia Institute of Technology in 1981, and a Ph.D. from Caltech in 1983. At Caltech he won the Clauser Prize for the most original doctoral thesis. He has been teaching at Caltech since 1983, and was recognized in 1996 with the Richard P. Feynman Award for Excellence in Teaching. He became a full professor in 1994.

Khaled Al-Ghoneim received the B.S. degree in computer engineering with First Class Honors from King Saud University (Riyadh, Saudi Arabia) in 1988. In 1992, he completed the M.S. degree with a thesis on Optical Parallel Multiplication at Carnegie Mellon University. He received a Ph.D. degree also from CMU in 1996 in the field of statistical pattern recognition. Since then he has been with the computer engineering department at King Saud University (Riyadh, Saudi Arabia) as an assistant professor. His research interests include pattern recognition, machine learning, signal and speech processing, and neural networks. He is a member of IEEE, ACM, and the Saudi Computer Society.