The Use of Backpropagating Neural Networks in Coordination with Logistic Spline Coefficients to Obtain the Change Point

Elisa M. Maldonado-Colberg
Advisor: Daniel McGee

Department of Mathematics
University of Puerto Rico, Mayagüez Campus
Mayagüez, P.R. 00681-5000
elisa@math.uprm.edu
mcgee@cs.uprm.edu

Abstract

This article presents one phase of an algorithm that uses backpropagating neural networks and their ability to find logistic regression coefficients as a useful tool that may outperform other conventional methods for finding the change point of a data set. To seek this objective, we first demonstrate the capacity of a one layered neural network to find logistic regression coefficients. Then, we added another layer to the network in order to determine the two discriminants it found would allow us to approximate the change point. A recursive algorithm was created restricting symmetric data to successively smaller regions that bound the change point. Finally, another algorithm was developed to deal with asymmetric data cases.

1. Introduction

A significant barrier in modeling data sets is the existence of fundamental changes in the behavior of the data. For example, when modeling the probability of dying in an automobile accident based on age, the probability of dying in an automobile accident increases as the age of the person increases until the age of 23 when the probability decreases as age increases. Hence 23 would be the change point of the data and it would be appropriate to use one model to describe the data when the age is less than 23 and another model to describe the data when the age is greater than 23. Our research focuses on the use of neural networks and its associated logistic regression coefficients to determine the existence and the location of a change point.

Two layer backpropagating neural networks can be interpreted as finding various logistic regression equations in the first level and finding an appropriate balance between these in the second. Our intention is to use the first layer of the neural networks to indicate approximately where the change point is and then to create an algorithm which will recursively approach the change point using the weights in the second level of the neural network.

2. Background

1.2 **Neural Networks**

In general, a neural network's goal is to find a boolean function which receives a vector as an input and returns a boolean function as an output. The input vector's components are, generally, features or traits and the function's output is 1 if it determines that the object belongs to a certain class of objects or endpoints. Otherwise, the function's output is 0.

The processing elements of neural networks are often sigmoid functions such as:

$$\text{Sigmoid} = \frac{1}{1 + e^{-(\vec{a}\,\vec{x} + b)}} \quad \textbf{eqn: 1}$$

2.2 Neural Networks and Probabilities

Neural networks are most often used with relatively clean data where their purpose is to determine classification. This term is used to describe data which can be easily and clearly classified into different sets. Graphically, the sets or populations are separated by a noticeable gap. In this case, the network outputs zero for a pattern belonging to a certain class. Otherwise, its output equals to one.

However, it is difficult to classify elements of different sets where the data is partially or completely mixed up. Graphically, no noticeable gap is shown between different sets of elements. Neural networks are often being used for "fuzzy data" sets as well where degree of belonging to a set is measured. This "fuzzy belonging" can be perceived as the probability that an item belongs to a certain set.

2.3 Change Point Problem

When we seek to model a population, we often find that there are in fact two distinct populations with distinct behaviors. Trying to model such populations with a single model is difficult; therefore, we need to separate those populations and model them separately.

Because of this, the change point problem consists on finding a transfer point in which the population undergoes a significant change. Once the change point has been identified, the two subpopulations can be modeled separately with whatever modeling scheme is deemed appropriate.

The most common techniques for obtaining change points are sequential searches or Monte Carlo sampling techniques based on a maximization of the maximum likelihood equations.

3. Experiments

3.1 Experiment 3 –Finding the Change Point with Symmetric Data

A C++ program was written to train and implement two layered backpropagating neural networks with the objective of seeing whether the discriminants obtained in the first level of the neural network would help in obtaining a *change point*. Once the *change point* is found, we can use two different representation models (logistic splines, for example) for the data on each side of this point.

Using as input ten different Matlab files containing uniformly distributed random numbers, ten simulations were performed by modeling a probability distribution in the xyz space using the interval x,y=[0,8] for its representation. The value of z=0 or 1 was given by P(z=1)=.25(x+y) for x+y in [0,4] and P(z=1)=2-.25(x+y) for x+y in [4,8]. The goal for the neural network was to obtain two *discriminants* that would be eventually used to find the *change point*.

Once the discriminants are calculated, the data that resides far from the change point are not crucial when determining where the change point is precisely located. For this reason, we wish to create an algorithm to successively restrict data about the change point until we have a very narrow region surrounding the change point with the second level of the neural network providing equal weight to the two sides. The algoritm consists of the following. If $\begin{pmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \end{pmatrix}$ are the coefficients obtained by the first level of the neural network trained with two input values (the third is always constant) and $\begin{pmatrix} y_1 & y_2 & y_3 \end{pmatrix}$ are the coefficients of the second level of the neural network, the following algorithm will be implemented.

1. When the neural network is trained, the approximate change point is the line
$$y_1\left[\left(\frac{x_{11}}{x_{11}+x_{12}}\right)x+\left(\frac{x_{12}}{x_{11}+x_{12}}\right)y+\left(\frac{x_{13}}{x_{11}+x_{12}}\right)\right]$$
$$+y_2\left[\left(\frac{x_{21}}{x_{21}+x_{22}}\right)x+\left(\frac{x_{22}}{x_{21}+x_{22}}\right)y+\left(\frac{x_{23}}{x_{21}+x_{22}}\right)\right]=0$$
2. The following equation will be converted to the form $x+ay=b$
3. The region over which the data is tested will be reduced to those *(x,y)* satisfying $b-4*(0.5)^i \le x+ay \le b+4*(0.5)^i$ where i is the number of the iteration.
4. Retrain neural network with the revised data set. Go to 1.

Finally, the region or interval should be enclosed iteratively within the change point based on the two discriminants and the weights of each provided in the second level of the neural network.

3.2 Experiment 3 –Finding the Change Point with Asymmetric Data in n Dimensions

When we dealt with symmetric data, an algorithm was developed to successively restrict data about the change point until we had a very narrow region surrounding the change point with the second level of the neural network providing equal weight. This technique was refined because for asymmetric data it is estimating the change point as the mean of the two discriminants calculated in the first level of the neural network.

If $\begin{pmatrix} X_{11} & X_{12} & X_{13}...X_{1n} \\ X_{21} & X_{22} & X_{23}...X_{2n} \end{pmatrix}$ are the coefficients obtained by the first level of the neural network trained with two input values (the third is always constant), $\begin{pmatrix} Y_1 & Y_2 & Y_3 \end{pmatrix}$ are the coefficients of the second level of the neural network and n is the dimension of the data:

1. The distance between the discriminant $L: Ax+By+C=0$ and a point $P_i(x_i, y_i)$ is given by the formula:
$$d(P_i, L) = \frac{|Ax_i + By_i + C|}{\sqrt{A^2 + B^2}}$$

2. When the neural network is trained, the location of the change point can be found using the following equation:

$$Y_2 * d(P_i, L) = Y_1 * d(P_i, L)$$

Using the coefficients $\begin{pmatrix} X_{11} & X_{12} & X_{13}...X_{1n} \\ X_{21} & X_{22} & X_{23}...X_{2n} \end{pmatrix}$ and $\begin{pmatrix} Y_1 & Y_2 & Y_3 \end{pmatrix}$ obtained in the first and second level of the neural network respectively and substituting the formula for $d(P_i, L)$ in both sides of the equation, we have:

$$Y_2 * \frac{X_{11}x + X_{12}y + X_{13}z + ... + X_{1n-1}m + X_{1n}}{\sqrt{X_{11}^2 + X_{12}^2 + X_{13}^2 + ... + X_{1n}^2}} = Y_1 * \frac{X_{21}x + X_{22}y + X_{23}z + ... + X_{2n-1}m + X_{2n}}{\sqrt{X_{21}^2 + X_{22}^2 + X_{23}^2 + ... + X_{2n}^2}}$$

After multiplying and rearranging similar terms:

$$(\frac{Y_2X_{11}}{div_1} - \frac{Y_1X_{21}}{div_2})x + (\frac{Y_2X_{12}}{div_1} - \frac{Y_1X_{22}}{div_2})y + (\frac{Y_2X_{13}}{div_1} - \frac{Y_1X_{23}}{div_2})z + ... + (\frac{Y_2X_{1n-1}}{div_1} - \frac{Y_1X_{2n-1}}{div_2})m = (-\frac{Y_2X_{1n}}{div_1} + \frac{Y_1X_{2n}}{div_2})$$

where $div1 = \sqrt{X_{11}^2 + X_{12}^2 + X_{13}^2 + ... + X_{1n}^2}$ and $div2 = \sqrt{X_{21}^2 + X_{22}^2 + X_{23}^2 + ... + X_{2n}^2}$.

This change point's location equation can be converted to the form x+ay+bz+…cm = d dividing both sides by $(\frac{Y_2X_{11}}{div_1} - \frac{Y_1X_{21}}{div_2})$. Then, the coefficients are:

a = $\frac{(\frac{Y_2X_{12}}{div1} - \frac{Y_1X_{22}}{div2})}{divisor}$, b = $\frac{(\frac{Y_2X_{13}}{div1} - \frac{Y_1X_{23}}{div2})}{divisor}$,

c= $\frac{(\frac{Y_2X_{1n-1}}{div1} - \frac{Y_1X_{2n-1}}{div2})}{divisor}$ d= $\frac{(-\frac{Y_2X_{1n}}{div1} + \frac{Y_1X_{2n}}{div2})}{divisor}$,

where $divisor = \frac{Y_2X_{11}}{div1} - \frac{Y_1X_{21}}{div2}$.

Experiment 3.3: Parallelization of Change Point Problem

A sequential C++ program is being written to solve the change point problem for symmetric and asymmetric data using backpropagating neural networks and their associated logistic splines.

Since parallel implementation schemes offer an attractive way of speeding up processes and

calculations in artificial neural networks, we intend to later parallelize the developed sequential code, using the Message Passing Interface (MPI) library.

MPI is a library of functions that can be called from a C, C++ or Fortran program. The foundation of this library is a small group of functions that can be used to achieve parallel programming.

The performance and running time of both codes, sequential and parallel, will be measured, analyzed and compared using benchmarks made in C++ using the MPI library functions.

The combination of these techniques allow a variety of two and three dimensional plotting of the datasets used as inputs.

4.Results

4.1. Experiment 1

Table 4.1 Discriminants Found in the Second Level of the Neural Network

No. of Simulation	Discriminants
1	2.43, 6.64
2	2.66, 5.98
3	2.66, 6.63
4	2.18, 6.75
5	2.87, 6.14
6	2.78, 6.47
7	2.43, 6.73
8	2.54, 6.09
9	2.61, 6.44
10	2.86, 6.07

The neural network is always calculating an approximation of the expected discriminants (which are 2.0 and 6.0 for this particular case). As we were not interested in an extended training session and as the coefficients of the first level of the neural network use all data to obtain these coefficients, these results are not surprising.. I.e., the network is calculating the coefficients with data on both sides of the change point despite the fact that the parameters are appropriate for only one side. This fact highlights the importance of the change point in building two representation models on each side of it. The most important element of this data is that the two discriminants do provide an indication of where the change point resides.

Table 4.3 Change Point

No. of Iterations	Change Point
1	x+1.06y=4.3
2	x+1.007y=4.15
3	X+1.003y=3.97

Theoretically, the change point resides at the line x+y=4.0. As table 4.3 shows, each iteration successively approaches the actual changepoint. Although we haven't explored the rate of convergence, with this simulation it appears to converge with order 1.

4.2 Experiments 2 and 3

See Discussion Section.

5. Discussion

Simulations with two variables have verified the effectiveness of backpropagating neural networks to quickly approximate changepoints in simple simulations involving 3 variables.

The algorithm is currently being tested with simulations using asymmetric data in higher dimensions and no final results have been produced. Simultaneously, the sequential algorithm is being parallelized using shared and distributed memory algorithms implemented with the MPI (Message Passing Interface) library functions. The algorithm's parallelization is expected to reduce its execution time by half.

Should the algorithm continue to perform as well as it has performed this far, we will then proceed to use the procedure with experimental data from the Puerto Rico medical study.

Finally, we will compare this method with various other solutions to the change point problem.

References

"Measures and Meanings of Blood Pressure", Michael H. Alderman, "The Lancet", Jan.15,2000

"Applied Regression Analysis"; Norman R. Draper, Harry Smith, John Wiley and Sons,1998.

"Index-Mortality Relationship: A Case Study"; Durazo-Arvizu R., McGee D., Li Z, and Cooper R., Journal of the American Statistical Association, Vol. 92, No.440, pp.13121319, 1997