University of Montana

ScholarWorks at University of Montana

Graduate Student Theses, Dissertations, & Professional Papers

Graduate School

2003

Software system for spatio-temporal prediction and analysis

Chellam Balasundaram Chellam
The University of Montana

Permission is granted by the author to reproduce this material in its entirety, provided that this material is used for scholarly purposes and is properly cited in published works and reports.

Please check "Yes" or "No" and provide signature

Yes, I grant permission ✓

No, I do not grant permission ______

Author's Signature: Rachelle

Date: 18 NOV 2003

Any copying for commercial purposes or financial gain may be undertaken only with the author's explicit consent.

Yes

8/98

A SOFTWARE SYSTEM FOR SPATIO-TEMPORAL PREDICTION AND ANALYSIS

BY

CHELLAM BALASUNDARAM CHELLAM

Bachelor of Engineering, Electronics and Communication Engineering.

Government College Of Engineering, India

Presented in partial fulfillment of the requirements for the

Degree of Master of Science

The University of Montana

2003

Approved by:

David W. Opitz

Chairman, Board of Examiners

Dean of the Graduate School

11/24/03

Date

Chellam, Chellam MS Dec 2003 Computer Science

A Software System for Spatio-Temporal Prediction and Analysis

Director: David W. Opitz DWO

Aerial imagery obtained through satellites has become a critical source of information in many fields (such as agriculture, climate studies, and mineral exploration). This has resulted in large spatio-temporal image databases. While there exists software systems to explore and exploit spatial databases, these systems fail to incorporate temporal information. Our research is to develop a software system with a flexible interface that can predict the future from a time-series of spatial data. This software system integrates suitable machine learning and visualization techniques to explore and exploit spatio-temporal data. The software is developed as an ArcGIS extension, a popular, commercial, GIS software suite.

Acknowledgements

This thesis could not have been written without the assistance, help, encouragement and support of many people. The foremost among them is my advisor. I thank Dr. David Opitz, my guru for guiding me through the adventurous and wonderful trail of research. I also thank him for providing me with the necessary infrastructure through the Machine Learning Research Lab. He displayed incredible patience as my process was interrupted repeatedly by my personal struggles. If not for his advice, commitment, and inspiration this thesis would not have been written.

Being an interdisciplinary research, I want to thank Dr.Ramakrishna Nemani for the valuable discussions, which helped me to solve many issues in regard to satellite imagery and GIS. I also want to thank Dr. Jessie Johnson for his critical evaluation of my thesis write up.

I have to give credit to my parents. I always saw my dad as a role model when it comes to scientific research. Being a scientist himself, my dad played a crucial role in showing me the joy of learning and being curious. My mom's love was a great pillar of support through the rough road of graduate study.

I thank my friend Konda, for his meaningful contribution to this thesis work. I thank Will for his review and corrections on thesis write up.

I also want to thank the computer science department's staff Eric, Kathy and Linda.

Table of Contents

List of Figures

1.0 INTRODUCTION

Satellite imagery is widely used to predict and analyze the Earth's environmental and climatic changes. Satellites accumulate imagery in the range of few hundred terabytes daily. Such large databases are a valuable source of spatial information that can be used for predicting the future. Forecasting future events from a database is called time series prediction. Much work has been done in this research topic [Georg Dorffner 94]; however, there is a need for applying proven time series prediction techniques on spatial data over time. The focus of this thesis is to develop a flexible system that allows a user to explore and exploit the latent information in spatial data over time.

In recent years, the Geographic Information System (GIS) community has developed a gamut of GIS tools for exploring huge spatial databases. Proper GIS analysis requires both spatial and temporal data as attributes change over time; yet few applications are available to help GIS specialists work on temporal data. For example, a health department may need to forecast the growth of an epidemic in terms of the number and the location of diseased cases. A city-planning department might want to know the future number of urban houses and their geographic location. Climatology scientists may need to predict when and where a given cyclone might hit the coast. The above cases are examples of the importance of exploring spatial data over time. Time series prediction (i.e., the prediction of future events from the analysis of past events) is a well-explored field [Georg Dorffner 94]. However *spatial* time series prediction has been for the most part neglected. As a result, in this thesis we develop a software system for spatial time series prediction. This software system uses inductive learning, which is the task of approximating a target function with the help of classified examples. Key components of successfully applying inductive learning algorithms to a problem are, (a) finding appropriate training examples, and (b) defining an effective input representation for the examples.

The software system developed has been designed with a flexible user interface that allows the user to easily interact with inductive learning algorithms. This includes the ability to reduce the number of inputs fed into the learning algorithm, select the type of learning algorithms, vary the parameters of the learning algorithm, etc. The system starts by reducing the dimension of raw inputs by applying an unsupervised learning algorithm and thereby deriving meaningful groups of data points or clusters. Feeding enormous (irrelevant) amounts of data or irrelevant input variables increases the solution space of the problem and hence the search for the optimal solution becomes difficult. Unsupervised learning addresses this problem by giving us groups of data points, which are relevant and similar. The interface allows the user to set learning parameters and easily pick relevant inputs. After the user sets up the problem, the model is trained and is used to predict and analyze the time series.

We tested our system on the Earth's surface temperature images taken for each month over a period of 20 years. Our clustering algorithm gave results that effectively delineated the land and sea regions. The learning models showed the ability to exploit information from the past 6 to 12 months to increase predictive accuracy.

The rest of the paper is organized as follows. First we provide a brief introduction to induction algorithms, and different topologies of neural networks. We include in our discussion how the learning algorithms were adapted for the unique problem of time series prediction. In next section we talk about time series prediction and analysis. This is followed by a discussion on unsupervised learning algorithm and how it is adapted to our problem. Finally, we present a detailed description of our software system followed by experiments and future work.

2.0 BACKGROUND

2.1 INDUCTIVE OR SUPERVISED LEARNING ALGORITHM

Inductive learning is the task of drawing general principles from a set of experimental data and making correct predictions on future data based on the induced general principles. These general principles can be described by a mathematical function called a hypothesis (also known as the target function). Given a sufficiently large set of training examples, representing a proper hypothesis space, an inductive learner can successfully generalize beyond the observed data [Mitchell, 1977]. This process of learning from classified examples is also called as supervised learning. The objective is to classify the new unclassified instance by learning the pattern associated with the classified training examples. Back-propagation neural networks are a popular example for *inductive* or *supervised* learning [Werbos, 1995].

Important points about supervised learning include:

- A problem should be aptly defined with relevant attributes or feature lists.
- Training examples must be chosen such that they are representative of the concept to be learned.

- The target function, which describes the learnt hypothesis, must be chosen meticulously because it determines the knowledge learned and how well it can predict the future instances.
- The toughest part in implementing inductive learning techniques is the initial choice of hypothesis space; it has to be large enough to contain a solution for the learning problem, yet small enough to ensure good generalization from a small number of examples [Mitchell, 1991]. Bias is the a-priori preferential ordering of hypotheses. Normally, bias is introduced by the selection of appropriate set of features by a human expert.

2.2 TIME SERIES ANALYSIS and PREDICTION

Time series is a set of time stamped data entries. A time series allows a natural association of data collected over intervals of time. A mathematical explanation of time series is, $\overline{X}(t)$ is a sequence of vectors that are dependent on time t. The components of the vectors can be any observable variable, such as, the temperature of a land region, the population in a state, etc. Theoretically, X can be viewed as a continuous function of time variable t, but for practical purposes it is viewed as discrete time steps [Georg Dorffner 94]. The time duration between each time step is called sampling time period.

Time series processing can be divided into four broad categories:

- Forecasting of future developments of the time series.
- Classification of time series into one of several classes.
- Description of a time series in terms of the parameters of a model.
- Mapping of one time series onto another.

In this thesis, we focus on the forecasting of future developments in time series. Such as the prediction of temperature on the California coast, and the mapping of one time series onto another, such as finding the correlation between California coastal temperature and California land temperature over the same period of time.

Typically when it comes to handling and analyzing time series, a simple predictor is used, such as a linear or exponential regression or some form of smoothing such as a moving average [Makridakis, et al 98]. For complex problems, the most widely used method is Auto Regressive Moving Average models (ARMA). ARMA models are linear functions of previous variables. However, when the process is non-linear and with relevant and sufficient data, neural networks will produce a more accurate model than the ARMA model [Box, et al 1994].

2.3 BACK-PROPAGATION

Back-propagation neural network [Rumelhart, et al 1986][Werbos 94] [Rumelhart, et al. 1988] is a popular supervised learning algorithm that is applied to pattern classification problems. Figure 1 depicts a simple back-propagation neural network with six input units, three hidden units and two output units. Each interconnection is governed by weights, which are altered during the course of learning. Back-propagation is a training algorithm that learns the proper weights for a multi-layered network with a fixed set of units and interconnections [Mitchell, 1977].

Back-propagation is designed for learning structural static patterns in a data set. Learning static patterns means the learner can induce the pattern or the function irrespective of the order of the training examples fed into the learner. When we have data sets across time intervals, we are trying to learn the dynamic pattern, which means, that the pattern evolves over a period of time and hence the order of training examples determines the pattern or function learned. For this thesis, then, back-propagation needs to be adapted to learn evolving patterns, as opposed to static patterns. We achieve this by using time-delayed input and window.

For example, lets assume T1, T2 and T3 are inputs to the network and T4 is the output. Then the window size is 3, because we have only three time stamps. The next training step would include T2, T3, and T4 as inputs and T5 as output, and so on. The network is trained in this fashion. In order to predict Tn we have to input Tn-1, Tn-2 and Tn-3.

It is also possible to sample inputs on a periodic basis. For example we can have T1, T3 and T5 as input and the output be T7. In this case, the delay or sample period is 2. In our project we call the input window (*HISTORY*) and sampling time period for time delay (*INCREMENT*).

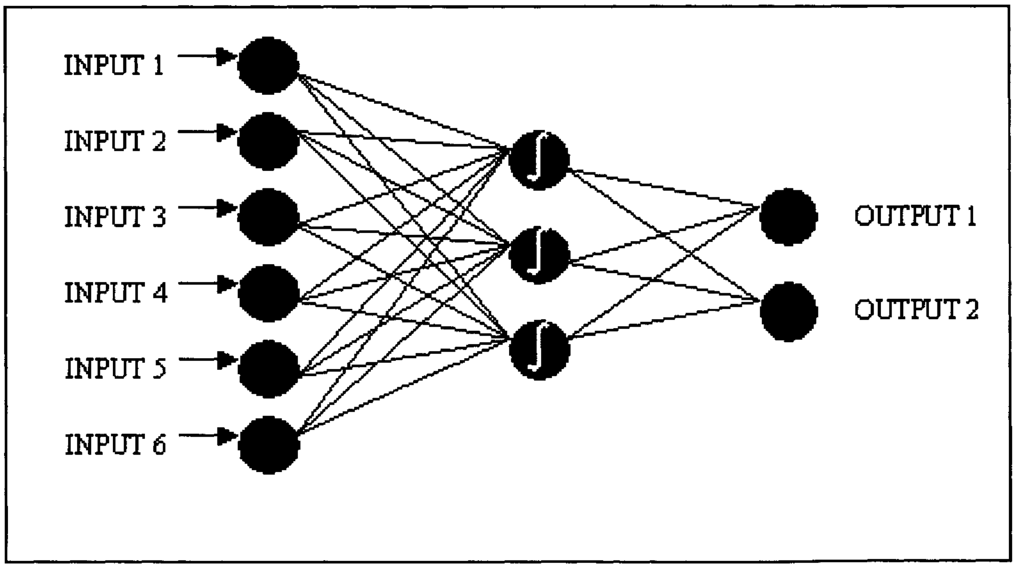

FIGURE 1. BACK-PROPAGATION NETWORK WITH SIX INPUT UNITS, THREE HIDDEN UNITS AND TWO OUTPUT UNITS

2.4 RECURRENT NEURAL NETWORK

Another technique for times series prediction problems are recurrent neural networks, which are trained without a window and time delay. Two common architectures are Elman networks [Elman 90] and Jordan Networks [Georg Dorffner 94].

Figure 2 shows the block diagram of the operation involved in the recurrent neural network. The algorithm for the recurrent neural network is similar to that of back-propagation algorithm, except that at each instance of training, a feed back loop of later hidden and output units are fed along with the current instant's input. With Elman networks, only later hidden units are fed as additional inputs, whereas with Jordan networks, both hidden and output units are included.

The general steps to these networks are:

1. Copy inputs for time *t* to the input units.
2. Compute hidden unit's activation using net input from input units and copy layer.
3. Compute output unit's activation as usual.
4. Copy new hidden unit's activation to Context Layer for an Elman Network (dotted lines Figure 2) and copy both hidden and output units activation of previous instant to Context Layer for the Jordan Network (dotted lines Figure 2).

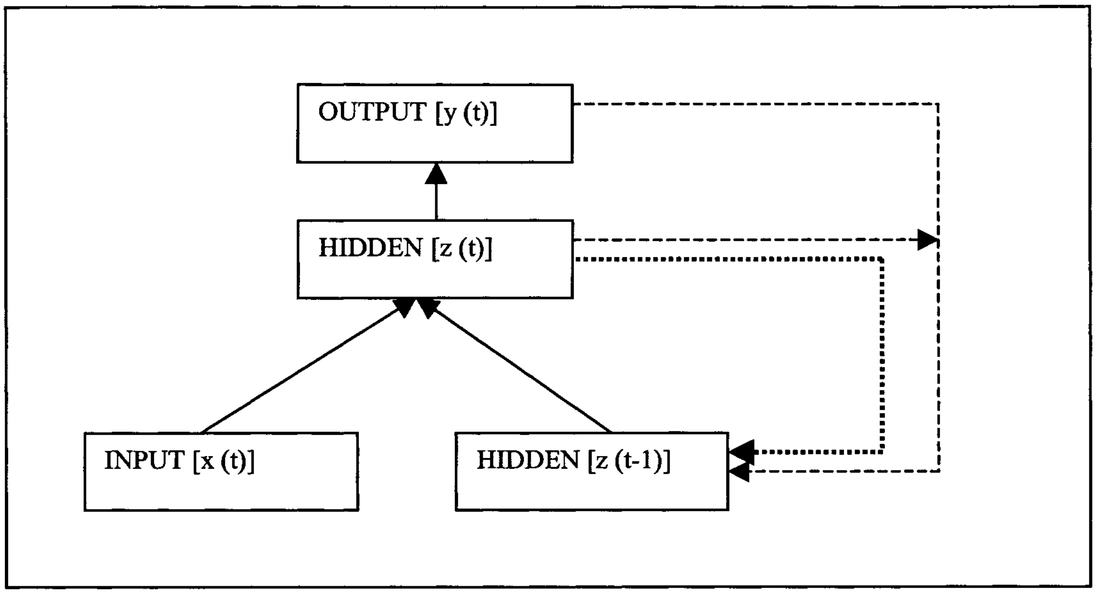

Figure 2. Working of recurrent neural network (dashed line represents Jordan network and dotted lines represent Elman network).

2.5 UNSUPERVISED CLASSIFICATION: CLUSTERING ALGORITHM

As an alternative to supervised learning, unsupervised learning algorithms are those in which training examples are not used, the responsibility of classifying the data into new patterns lies with the unsupervised learner. Data analysis procedures can be categorized either as exploratory or confirmatory based upon the availability of appropriate models for the data source. Cluster analysis (which falls into the exploratory category) is the organization of a collection of patterns into clusters based on similarity.

Data clustering techniques are used for classifying observed data into clusters, which satisfy two main criteria:

- Each cluster is homogeneous; that is, data that belong to the same group are similar to each other.
- Each cluster should be unique; that is, data that belong to one cluster should be different from the data belonging to other clusters.

An example of clustering is depicted in Figure 3. The input patterns are shown in Figure 3(a), and the desired clusters are shown in Figure 3(b). Here, points belonging to the same cluster are given the same label.

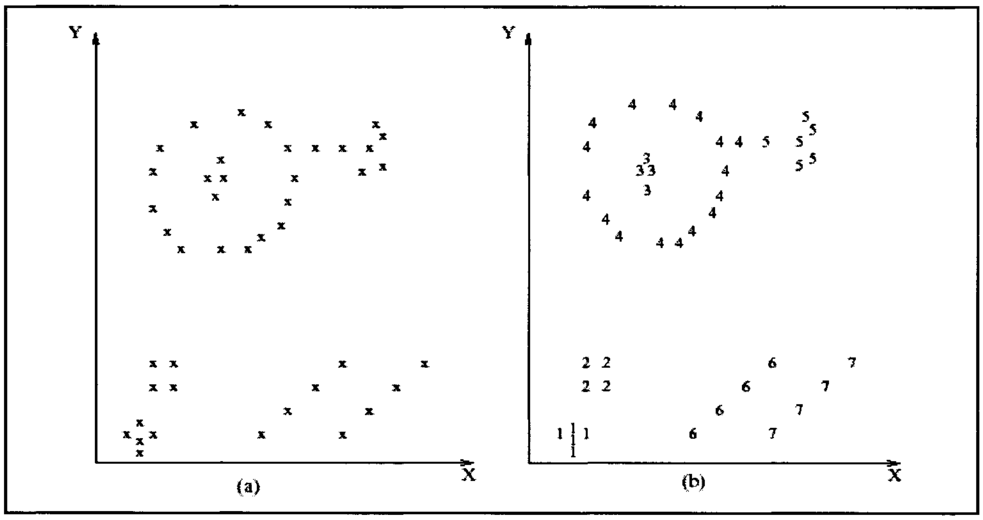

Figure 3(a) The distribution of data points "x" 3(b) Shows the clusters formation

For our system, we choose K- Means clustering [Jain and Dubes 88] because it is a popular and simple clustering algorithm (shown in Table 1).

K Means algorithm's goal is to find K clusters. To start with, we select K random cluster centroids. We then take each data point and find the Euclidean distance between the point and the K centroids. The data points are assigned to the respective cluster with which it has the least Euclidean distance. Once the new data point is added to the cluster, the particular cluster's centroid is recalculated (by finding the average of all the data points, which includes the new data point too). The algorithm repeats steps 2 to 4 until convergence is reached. Convergence occurs when all the centroids remain constant.

Goal: To cluster data into groups of similar data points. (1) Randomly choose K cluster centers or centroids. (2) Assign each point to the closest cluster center or centroid based on the shortest Euclidean distance from the reference point to the K centroid. (3) Recompute the cluster centers using the current cluster memberships by taking the average of all the points in each cluster. (4) If a convergence criterion is not met, go to step 2.

Table 1 K - Means Clustering Algorithm

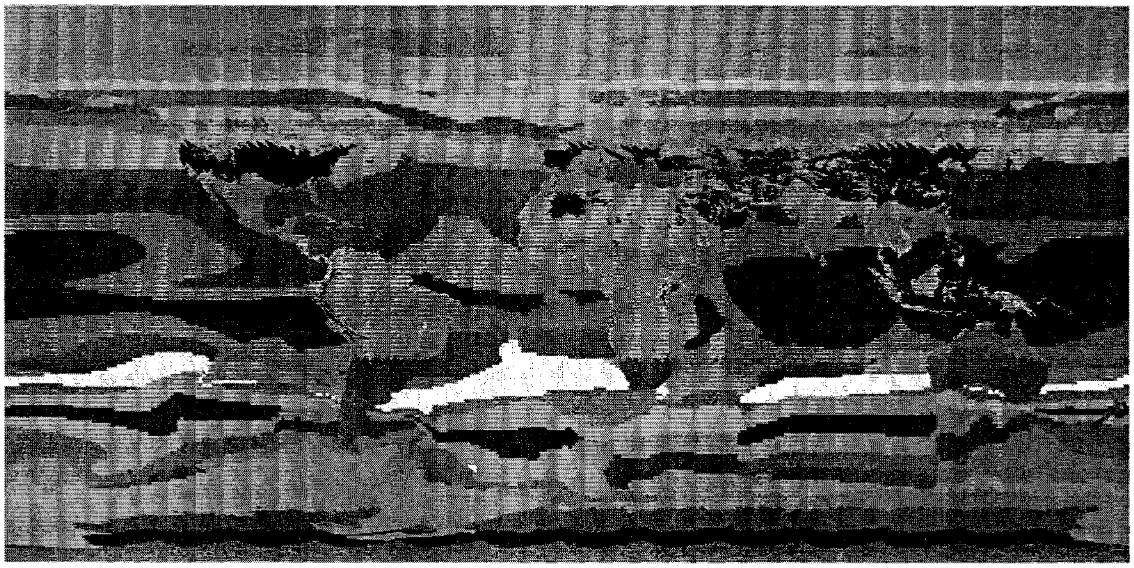

Figure 4. Image with 50 clusters

Figure 4 represents a clustered image of 20 years of satellite surface temperature data after applying the K-Means clustering algorithm. The aim of data clustering techniques is to bring order or structure to the huge mass of data. In our case, we want to classify regions of similar behavior or regions with similar temperature range of values. By applying data clustering we eventually achieve the image segmented (zones) into oceans and land into smaller pieces that are relatively homogeneous.

2.6 GIS (Geographic Information Systems)

GIS is a system of hardware and software used for storage, retrieval, mapping, and analysis of geographic data. Spatial features are stored in a coordinate system (latitude/longitude, state plane, etc.) that references a particular place on the Earth. Descriptive attributes in tabular form are associated with spatial features. Spatial data and associated attributes in the same coordinate system can then be layered together for mapping and analysis. GIS can be used for scientific investigations, resource management, and development planning.

ArcGIS, from Environmental Systems Research Institute, Inc (ESRI) is a family of software products that forms a complete GIS capability. ArcGIS is a complete, single, integrated system for geographic data creation, management, integration, and analysis. ArcGIS provides a catalog for browsing and managing data, on-the-fly coordinate and datum projection, metadata creation, customization with built-in Visual Basic environment, new editor tools, support for static annotation, cartographic tools, multi-user editing, advanced analysis, Internet services, and high performance spatial database services. ArcGIS is the market leader in GIS software both in the educational and commercial world. For these reasons, we developed our system as an extension to ArcGIS.

3.0 SYSTEM DESCRIPTION

3.1 DESCRIPTION OF SOFTWARE SYSTEM BLOCK DIAGRAM

Figure 5 shows the block diagram of the software system. The raw data is first clustered into groups of meaningful clusters using the K-Means clustering algorithm (with K = 50 clusters). The clusters are then pre-processed to represent the data points in terms of statistical parameters (mean, standard deviation/variance, minimum and maximum). Once the inputs have been preprocessed, the learning networks (backpropagation, Elman and Jordan) are trained to develop predictive models.

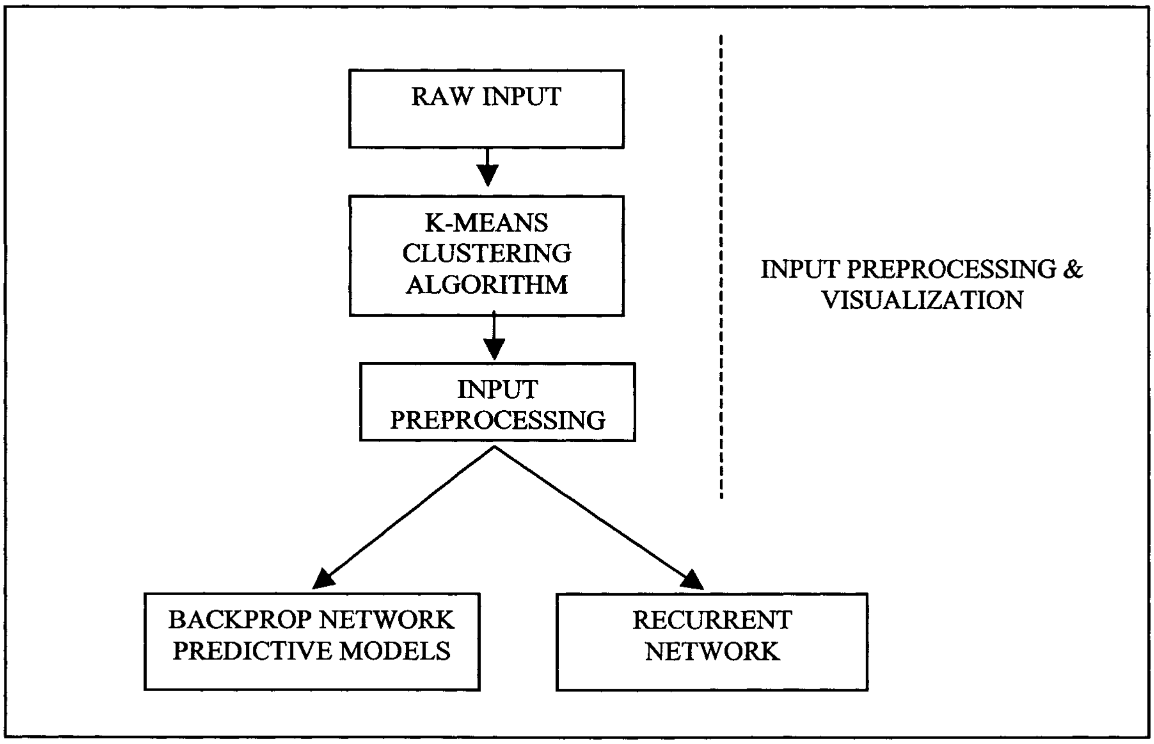

Figure 5 Software system block diagram of the operation involved.

3.2 INTERFACE DESCRIPTION

Figure 6 shows the interface of the "ClusterViewer", where the user can view the clustered image. The offline-generated image with 50 clusters is loaded into the interface "ClusterViewer" in the CLUSTER IMAGE panel. The user selects (by clicking the relevant clusters on the image) the clusters of interest to be used as input and target clusters for training the neural network.

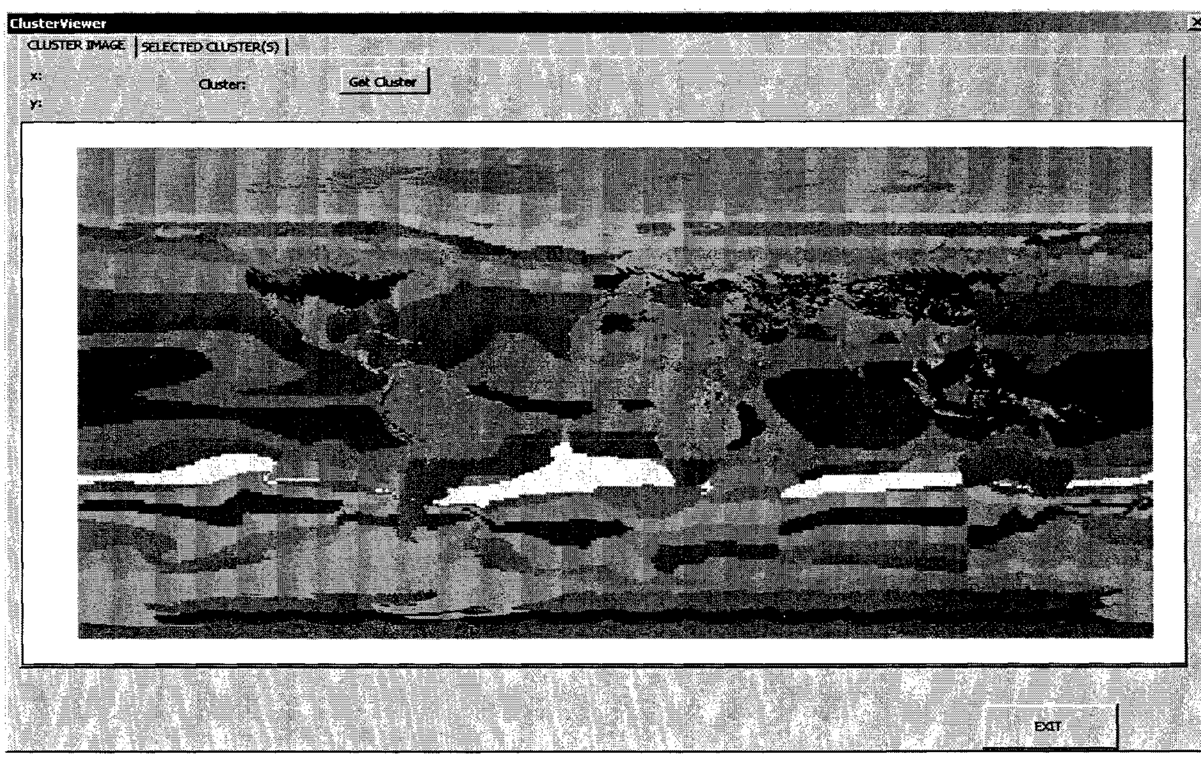

Figure 6. "ClusterImage" panel allows the user to select the clusters (by clicking) for experiments.

The panel "SELECTED CLUSTERS" (see Figure 7 below) displays the clusters clicked by the user in the previous step. This panel allows the user to calculate the statistics of each selected cluster across time (which can be selected from the combo boxes). Once the statistical values are calculated, the next step is to generate a pattern file upon which to train our neural network.

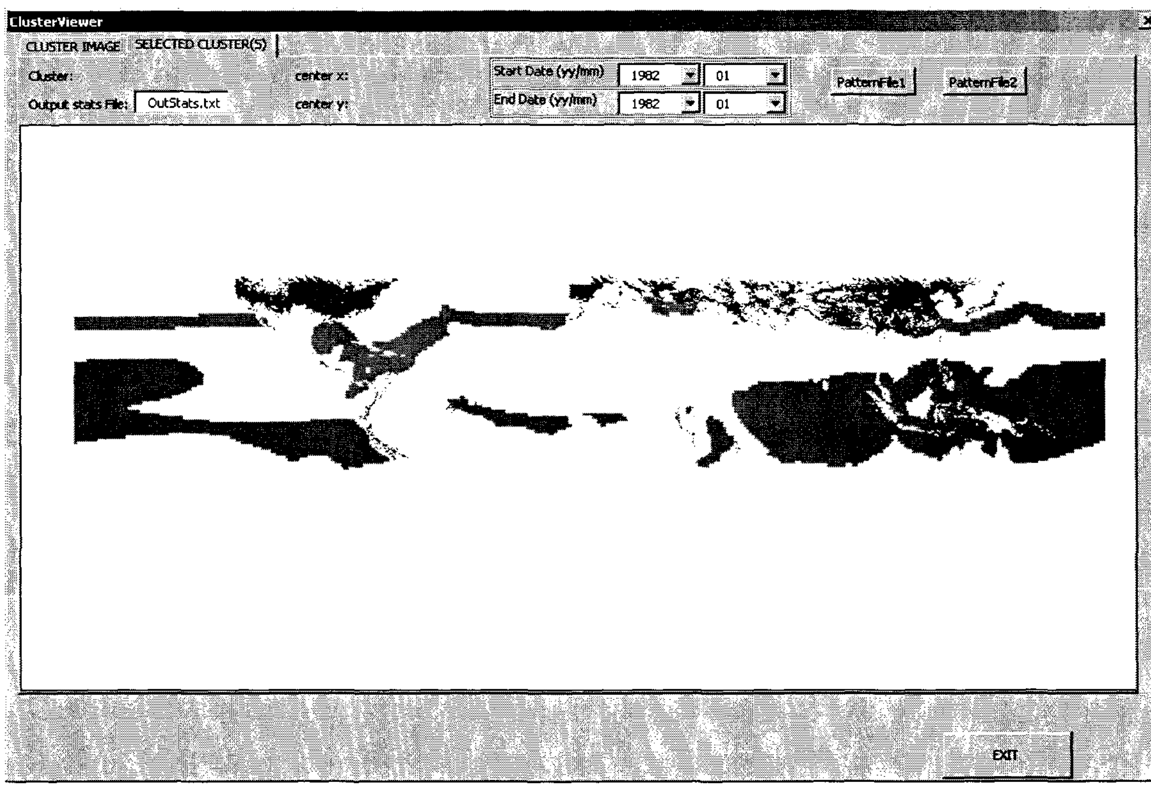

Figure 7.Selected clusters are displayed in the "Selected Cluster" panel and the time window is set for the statistical parameters to be calculated.

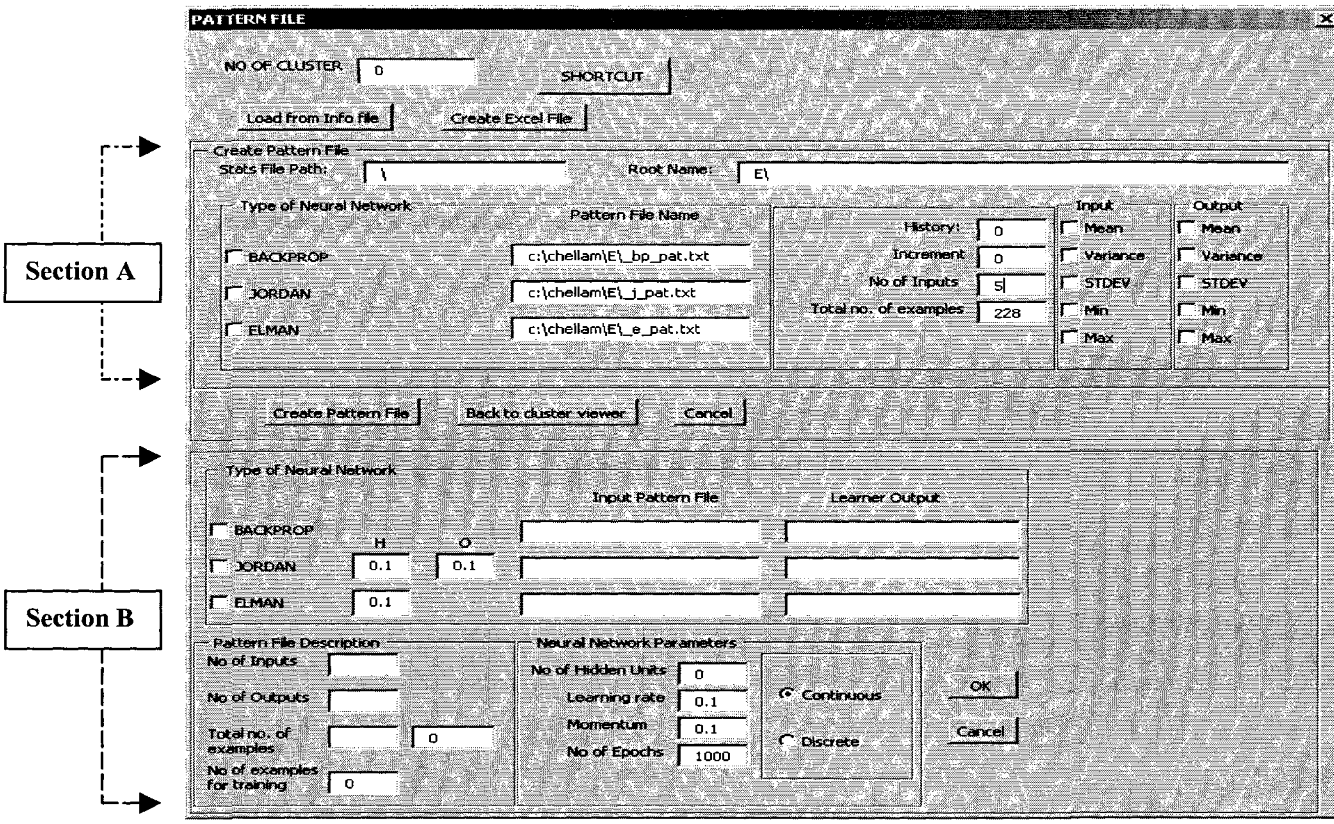

Figure 8. Interface for setting parameters for the learners

Since we are using three different time series prediction networks, we need to generate three different kinds of pattern files. The parameter boxes to be filled in Figure 8 are:

- Status File Path : The name of the statistics files generated in the previous step.
- Root Name : The path name where the user wants to store the generated pattern files.
- Type of Neural Network checkbox
  - BACKPROP : Checking this box generates the BACKPROP network pattern file
  - JORDAN : Generates the Jordan network pattern file
  - ELMAN : Generates the Elman network pattern file
- Other Parameters

- HISTORY : With the BACKPROP pattern file, this refers to how many past time intervals should be taken into account in generating training instances.
- INCREMENT : With the BACKPROP pattern file, the time delay involved in selecting the inputs with the given history.

- Output & Input checkbox
  - Checking the corresponding checkbox items in the input and output column means the user is selecting the particular statistical values as input or target attributes for all training instance in the pattern file.

Description of Section B

The second half of the interface shown in Figure 8 deals with the following:

- Selection of three learning networks and entering their corresponding pattern files for training
- Varying the learning parameters such as learning rate, momentum, number of hidden units and number of epochs
- Entering the file names to store the output or predicted values by the learning models and setting the number of training and testing examples in the pattern file
- The weights to be entered for the recurrent loops inputs in the recurrent neural networks
- A graphic interface, which allows the user to compare and contrast the predicted results with the original value (Figure 9).

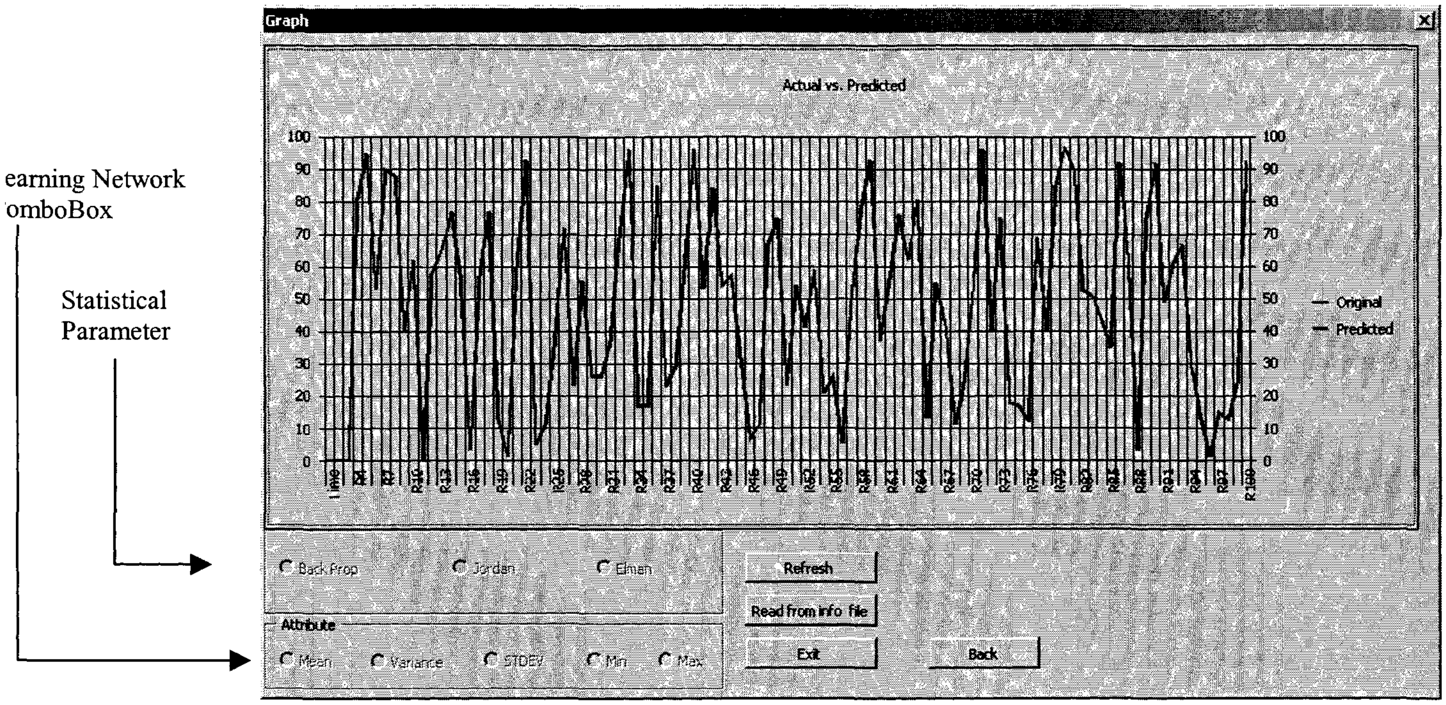

Figure 9. Graphical representation of the three-learning model's results

4.0 EXPERIMENT

4.1 DESCRIPTION OF DATA SET

For this thesis, we use whole Earth surface temperature images taken across the time scale from Jan 1982 to Dec 2000 for each month (228 months in total). Each surface temperature image consists of a 720 by 1440 pixel matrix. Each pixel in the image is represented by 2 bytes.

4.2 METHODOLOGY

For the experiments, we ran K-Means to find 50 clusters. We represented each clusters with the statistical parameters of mean, standard deviation / variance, minimum and maximum. We trained the three Learning Algorithms (Backpropagation, Elman and Jordan recurrent neural network) to the values set in Table 2. For the recurrent neural networks, the feedback loop weight was set to 0.10.

Learning Rate	0.01
Momentum	0.30
Number of Hidden Layers	10
Number of Epoch	1000

Table 2.Parameter values for the three learners

In the experiments conducted, the specific task is to predict the mean value of the land clusters with the help of the remaining clusters (49). Out of 50 clusters, four land clusters (which represents the important geographical regions Missoula, India etc.,) were selected manually (see Table 3).

Cluster	Important Region Encompassed
C1	Parts of India, Africa and Southern United States
C2	Parts of South America, Africa and Australia
C3	Parts of North Africa and Middle East
C4	Parts of Northern United States, Canada and China

Table 3. The clusters and the geographical region covered by the cluster.

We represent the selected clusters as C1, C2, C3 and C4 for convenience. Figure 10 visually presents the selected land clusters or regions. To create the learning model, the four land clusters were predicted using the remaining clusters (i.e., the remaining 49 clusters except for the target cluster in each context). Hence, the total experiments run were 49x4 for different combinations of histories, intervals, input values, and learning algorithms. The root mean square error (RMS) is calculated to find the accuracy of the learning network in predicting the target. Mean square error is the average of the square of *errors* over N (where N is the number of variables and the *error* is the difference between predicted and original). The error-calculation method allows us to determine the amount of variance between the expected and actual output of a neural network. Once the error has been determined, the network can be trained so that the error will likely be lower next time.

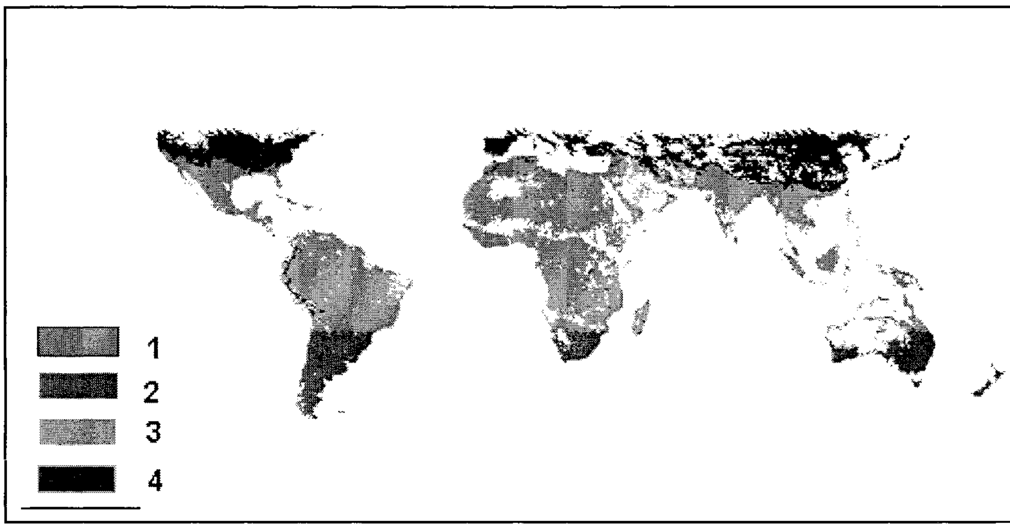

Figure 10. The selected land clusters 1, 2, 3 and 4

We tried various combinations of input parameters of the four statistical parameters (mean, standard deviation, maximum and minimum) to predict the target cluster's mean value. We found that using all parameters resulted in lower error than using a subset of the parameters. Hence, all experiments presented use all four parameters as input.

4.4 RESULT

Table 4 shows the summation of 49 error rates between an output land cluster and the remaining 49 clusters. We show the back-propagation results, since they had best performance. The columns indicate the land cluster (target) involved in the experiment. The rows represent the various history and increment combination used as input. Rows 1, 2, 3 and 4 are experiments run with variations in the history or past values and Increment set to 1. History is the set of past inputs fed as input and Increment is the sampling time for the history. The recurrent network does not need history, because its architecture innately has those properties.

From Table 3, we find that clusters C1, C2 and C4 [refer to rows 1, 2 and 3 in Table 4] increase in their correlation with the past values (as the error rate decreases) when the time window of the input or histories increases. But for a history of 24-month values [row 4 in Table 4], the correlation ceases to exist (except for C3).

No	Input	C1	C2	C3	C4
1	HISTORY 1	0.80665	0.88215	0.942715	0.916097
2	HISTORY 7	0.77135	0.84176	0.847776	0.834548
3	HISTORY 13	0.74575	0.786665	0.959724	0.825234
4	HISTORY 25	0.78221	0.81447	0.952456	0.885453
5	INCRE 12 HISTORY 1	0.77137	0.81447	0.926042	0.920832
6	*PAST 2 INPUTS	0.88485	0.90679	0.737776	0.986097
AVERAGE		0.793697	0.841051	0.894415	0.89471

Table 4 shows the summation of root mean square error rate (for all the Land Clusters 1, 2, 3 & 4) for the BackPropagation.

Also, we find that back-propagation performs better than the two recurrent neural networks, when the time window is increased (for recurrent neural network experimental results refer to Table 5).

In the next experiment, increment was set to 12 and history as 1 (row 5 in Table No.4). Increment 12 means that the sampling time for the input window is one year. There is not any significant increase in correlation between clusters with increment set to 12 in comparison with the previous experiments (rows 1 to 4 Table 4) and hence we can argue that the impact on the land clusters from other clusters is a function of immediate past.

For the final experiment [row 6 in Table 4] Increment is set to 1 and History is set to 3. But we also take the input clusters previous two years' values (t+12 and t +24, where t is the current instant) along with the inputs t, t+1, t+2 and t+3. This experiment was conducted to find the intensity of the correlation between the land clusters and the remaining clusters when the immediate past values plus previous year's values at that point in the cycle are taken. The error rate increases for all the target clusters except C3. We can conclude from the results that there is no correlation as the error rate increases (except C3) in comparison with previous experiments.

	ELMAN	JORDAN
C1	0.807273	0.80631
C2	0.88004	0.88180
C3	0.942216	0.943183
C4	0.916388	0.91656

Table. 5 shows the summation of root mean square error rate (for all the land clusters 1, 2, 3 & 4) recurrent neural network.

5.0 FUTURE WORK

The system shows considerable promise; however, there is ample opportunity for future improvements. One important future task is to transform our results into a user understandable format. We represent the accuracy of the predictive models in terms of the root mean square and based on the magnitude we conclude the extent of correlation. However, if the user wants to know the accuracy in terms of temperature values then this approach is not representative enough to evaluate the predictive models developed. The approach for solving this is to de-normalize all the results and find the variance in terms of temperature (as raw input). This value will help the user to assess the predictive models and also evaluate the predicted results.

Another future work direction is to narrow down the clusters that have high correlation with the target land clusters. Once we find the highly correlated clusters with the target, we can find whether there is a *combination* of the selected clusters that helps in predicting the target. This is a valuable piece of information to GIS scientists, as they discover different regions in the Earth that affect the climatic conditions of land simultaneously. Also, by varying the increment we can develop models with lead and lag in prediction. This means 6 or 12-month increment models can predict the climatic condition 6 or 12 months ahead of particular land region given the input of current trends. Anomalies are abnormal surface temperature points on the Earth. Instead of taking the statistical values of the raw data inside a cluster, we could first find the anomalies inside the cluster and then calculate the statistical values. Then we could apply the unsupervised learner (clustering algorithm) to get clusters of anomalous points. Next we could execute the same experimental methodology of training the back-propagation neural network to learn the patterns and correlation of the spatial time series. The shortcoming of our current approach is that the models developed learn the seasonalities of the spatial time series, which may not be the most interesting pattern for the user. Models developed using the anomalous data could learn the unique trends of the spatial time series.

6.0 CONCLUSION

The goal of this work is to develop software that can exploit the latent patterns of information in predicting spatio-temporal data. We have argued that GIS software tools can be more useful and powerful if developed to work on spatio-temporal data and not merely on spatial data. In this thesis, we presented and developed a novel approach to this problem. Our results on a data set consisting of global temperature values over the past twenty years demonstrate the potential of our system, which can serve as a precursor for further research in this area.

7.0 REFERENCES

[Georg Dorffner 96] Neural Networks for Time Series Processing - Georg Dorffner (1996)

[Jain and Dubes 88] Algorithms for clustering data. A.K. Jain and R.C. Dubes (1988).

[Elman 90] Finding Structure in Time, Cognitive Science - Elman, J. L (1990).

[Makridakis, et al 98] Forecasting Methods and Applications - Makridakis, Spyros, Steven C. Wheelwright, and Rob J. Hyndman, Third edition. John Wiley and Sons (1998).

[Box, et al 94] Time Series Analysis: Forecasting and Control -, Jenkins and Reinsel (1994).

[Micheal, et al 2001] Clustering Earth Science Data: Goals, Issues and Results - Michael Steinbach, Pang-Ning Tan, Vipin Kumar, Steven Klooster, Christopher Potter (2001).

[Mitchell, 1977] Machine Learning. Tom Mitchell. McGraw Hill (1997).

[Mitchell, 1991] The Need for Biases in Learning Generalizations. Tom Mitchell (1980)

[Rumelhart, et al 1986] Learning internal representations by error propagation, in Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1, Foundations, Cambridge, MA: MIT Press.

[Werbos 1994] The roots of back-propagation: From Ordered Derivative to Neural Network and Political Forecasting, New York, Wiley.

[Werbos 1995] Back-propagation: Basics and New Developments. Paul J.Werbos. The Handbook of Brain Theory and Neural Networks. MIT Press (1995).

[Rumelhart, et al.1988] McClelland, J., and D. Rumelhart. 1988. Explorations in Parallel Distributed Processing. Cambridge, MA: MIT Press.