Lib. Copy

1938

DOCUMENT EXPLORATION USING SELF-ORGANIZING MAPS

Dissertation submitted in partial fulfillment of the requirements for the award of the degree of

Master of Technology

in

Computer Science & Technology

by

NAGADASU PRAVEEN KUMAR

SCHOOL OF COMPUTER AND SYSTEMS SCIENCES

JAWAHARLAL NEHRU UNIVERSITY

NEW DELHI - 110067

JANUARY 2002

जवाहरलाल नेहरू विश्वविद्यालय
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI-110067 (INDIA)

SCHOOL OF COMPUTER & SYSTEMS SCIENCES

CERTIFICATE

This is to certify that the dissertation entitled **"Document Exploration Using Self-Organizing Maps"** being submitted by **Nagadasu Praveen Kumar** to the School of Computer and Systems Sciences, **Jawaharlal Nehru University**, New Delhi, in partial fulfillment of the requirements for the award of the degree of **Master of Technology in Computer Science** is a bonafide work carried by me under the guidance and supervision of Prof. **S.Balasundaram**. The matter embodied in the dissertation has not been submitted for the award of any other degree or diploma.

[illegible]
5/01/2002
Prof. S. Balasundaram
(Supervisor)
School of Computers and Systems Sciences,
Jawaharlal Nehru University,
New Delhi-110067

N. Praveen Kumar
Nagadasu Praveen Kumar

[illegible]
Prof. K.K. Bhardwaj
Dean.
School of Computers and Systems Sciences,
Jawaharlal Nehru University,
New Delhi-110067.

ACKNOWLEDGEMENTS

I express my deepest gratitude to my supervisor Prof. S.Balasundaram for his expert suggestions and invaluable guidance in course of this work. I am grateful to him for giving me time from his busy schedule even in weekends.

I would like to sincerely thank Mr. Krishnamachari for his constant moral help through out the course of the project period.

I express my gratitude to the Dean of the School, Prof. K. K. Bhardwaj for providing the computing facilities and Internet access.

I take this opportunity to thank to all my friends G. Srikanth, T. Srinivas, Panchanana, Deepak and Ramesh for their co-operation and valuable suggestions in editing this work.

Last but not the least, I thank to all of my classmates for making all these days in JNU enjoyable.

Nagadasu Praveen Kumar

CONTENTS

Chapter1

INTRODUCTION

Recently, a lot of web documents are distributed widely over World Wide Web, but it is difficult to reach desired web document for users. As the amount of web documents has grown to enormous amount, here are some demands for query facilities to find easily necessary information. An approach to overcome these problems is usually to use search engines. Search engines specialize in locating specific documents in answer to well-defined information requests. Another approach is to introduce the concept of navigation by query, which is implemented by WWW and DBMS. These approaches are useful to overcome the above problems to some extent. However, both of the approaches may often produce a vast of documents as a result of retrieval, and they do not provide any effective method for organizing the retrieval results, so we have a lot of labors to examine each document. Moreover, when we input keywords to search engines, only documents which containing all the given keyword are returned.

Large quantities of textual data available on the Internet pose a continuing challenge to application that help the user in making sense of the data. However, fulfilling a vague information need regarding an unknown domain, or obtaining an overview of a topic or a domain is very hard. Therefore, we need a function to classify documents corresponding to user's point of view and their purposes. The information need is better served by methods enabling a combination of visualization and interactive exploration.

To be able to solve the problems related to information retrieval discussed above we have implemented a Document exploration system that automatically organizes vast document collections efficiently and automatically according to textual similarities. In this scheme for document exploration, the purpose is to assist the user in familiarizing himself with a large collection of data and to facilitate interactive browsing.

In recent years, neural networks have been successfully applied to a variety of data analysis problems on complex data sets. Neural networks have shown their ability to summarize large databases in many real world applications. Application of Artificial Neural Networks may be recommended in areas that are characterized by noise, second poorly understood intrinsic structure, and third changing characteristics, each of which is present in text classification. The noise is imposed due to the fact that no completely satisfying way to represent text documents has been found so far. Second, the poorly understood intrinsic structure is due to the non-existence of a authority knowing the contents of each and every document. Finally, the changing characteristics of document collections are due to the fact that the collections are regularly enlarged to comprise additional documents.

From the proposed architectures of artificial neural networks, the unsupervised models are well suited for text classification. This is due to the fact that in a supervised environment, one would have to define proper input-output-mappings. Input-output-mapping refer to the manual assignment of documents to classes which is only possible when assuming the availability of considerable insight in the structure of the text archive. Contrary to that, in an unsupervised environment it remains the task of the artificial neural networks to uncover the structure of the document archive. Hence the unrealistic assumption of providing proper input-output-mapping is obsolete in an unsupervised environment. One of the most popular unsupervised neural network models certainly is the Self-Organizing Map. It is in general a unsupervised tool for ordering high dimensional data in such a way that alike input items are mapped close to each other. In order to use the Self-Organizing Map to cluster text documents, the various texts have to be represented as the histogram of its words. With this data, the artificial neural network performs the classification task in a completely unsupervised fashion.

This thesis describes dissertation work that has been carried out to develop an automatic method, Document Exploration System that enables easy exploration of very large collections of text documents. In this method, the Self-Organizing Map developed by T.Kohonen is applied to automatically organize large collections of text documents onto a two-dimensional display called the map.

The method places documents on regularly spaced map grid points where similar documents are generally found near each other. The resulting map can be browsed with a exploration interface. Label words positioned on the maps portray properties of the underlying map area. In addition, a search facility provides a means for describing a specific interesting topic and for finding a suitable starting point for exploration.

With technological advances, our possibilities to collect, generate, distribute and store text data have grown fast. Nowadays virtually anyone or any institution within the technologically developed world become an information provider for an unlimited audience. As a result, we are faced with a vast amount of textual data of unknown value. The former methods of managing the texts, such as libraries and hierarchies organized and catalogued by human effort, have become both inadequate and too expensive to perform and to maintain for the majority of the available data.

The use of automatic methods, algorithms and tools for dealing with large amounts of data, especially of textual data, has become necessary. Attempts to solve particular aspects of this general problem can be correctly described as efforts in text mining. Text mining can be viewed as a specific field of data mining. Data mining is the analysis of observed data sets to find relationships and to summarize the data in novel ways, which are both understandable and useful to the database owner. The data-mining field is closely related to exploratory data analysis.

Tasks that a data mining systems should help with include:

- Organizing, clustering and classifying of data.
- Creating overview and summaries.
- Identifying trends and changes across time.
- Identifying dependencies and unsuspected relationships with data.
- Providing other tools and indicators for specific decision making tasks.
- Visualizing properties of individual data items, of collections of data and of relationships between data items and collections.

The goal of a text mining system is to aid the user in fulfilling his information need. The major tasks related to various information needs could be described as searching, browsing and visualization.

Searching

In the searching approach, the user specifies an information request in terms of a query and asks the system to locate individual documents that correspond to the query. The Internet search engines are familiar example of tools that specialize in this task.

Browsing

In browsing, the user navigates in the collection of text, e.g. via links between individual documents. The browsing approach allows for the information need to be more unconscious, since no explicit description of the need is required. In both searching and browsing, the background assumption is that the information need of the user can be addressed by individual documents that the user should read. However, when the need is either very vague, or very general, providing access to even the most appropriate individual documents might not fulfill the need .In such cases, summary-like information might be more appropriate and useful.

Visualization

In visualization of information something familiar is used as a means for illustrating something yet unfamiliar. Some times, there exists information need that requires convergence similarities, differences, overlaps and other relationships between collections of documents. Using suitable visualizations intricate relationships between large collection of items can be fast and intuitively.

Shneiderman identifies the following seven major tasks that a visual and interactive information exploration system should address:

- Gains in overview of the entire collection
- Zoom in on items of interest
- Filter out uninterested items
- Select an item or group and get details when needed
- View relationships among items
- Keep a history of action to support undo, redo and progressive refinement
- Allow extractive of sub-collection and the study of their properties

The text mining tools should encompass all of the above aspects to explore large collection of text by enabling visualization, browsing and searching. The tool with above aspects can solve all the problems related vast document collection. The Document Exploration System addresses all above aspects except zooming and history maintenance.

Chapter 2
ARTIFICIAL NEURAL NETWORKS

Neural Networks are an information processing technique based on the way biological nervous systems, such as the brain, process information. The fundamental concept of neural networks is the structure of the information processing system. Composed of a large number of highly interconnected processing elements or neurons, a neural network system uses the human-like technique of learning by example to resolve problems. The neural network is configured for a specific application, such as data classification or pattern recognition, through a learning process called training. Just as in biological systems, learning involves adjustments to the synaptic connections that exist between the neurons. Neural networks can differ on the way their neurons are connected; the specific kinds of computations their neurons do; the way they transmit patterns of activity throughout the network; and the way they learn including their learning rate.

Neural networks are being applied to an increasing large number of real world problems. Their primary advantage is that they can solve problems that are too complex for conventional technologies, problems that do not have an algorithmic solution or for which an algorithmic solution is too complex to be defined. In general, neural networks are well suited to problems that people are good at solving, but for which computers generally are not. The goal of ANNs is not the recreation of the brain, but to engineer solutions to problems that have not been solved by traditional computing [11].

Neural networks, with their remarkable ability to derive meaning from complicated or imprecise data, can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. A trained neural network can be thought of as an expert in the category of information it has been given to analyze. This expert can then be used to provide answers if questioned.

Other advantages include:

Adaptive learning: An ability to learn how to do tasks based on the data given for training or initial experience.

Self-Organization: An ANN can create its own organization or representation of the information it receives during learning time.

Fault Tolerance : Partial destruction of a network leads to the corresponding degradation of performance. However, some network capabilities may be retained even with major network damage.

2.1 The Learning Process

Of all of the interesting characteristics of ANNs, the special one is their ability to learn. Learning is the determination of the weights. Every neural network possesses knowledge, which is contained in the values of the connection weights. A network is trained so that application of a set of inputs produces the desired set of outputs. During training, the network weights gradually converge to values such that each input vector produces the desired output. Learning methods are classified into two major categories: supervised and unsupervised.

2.1.1 Supervised Learning

Supervised learning requires the pairing of each input vector with a target vector representing the desired output; together these are called as training pair. Usually a network is trained over a number of such training pairs. An input vector is applied, the output of the network is calculated and compared to the corresponding target vector, and the difference (error) is fed back through the network and weights are changed according to an algorithm that tends to minimize the error. The vectors of the training set are applied sequentially and errors are calculated and weights are adjusted for each vector, until the error for the entire training set is at an acceptable low level. Supervised methods come in two sub-varieties: auto-associative and hetero-associative. In auto-associative learning, the target values are the same as the inputs, whereas in hetero-associative learning, the targets are generally different from the inputs.

2.1.2 Unsupervised Learning

Unsupervised learning requires no teacher (target output vector) for the outputs. The training set consists solely of input vectors. The training algorithm modifies network weights to produce output vectors that are consistent, i.e. both application of one of the training vectors or application of a vector that is sufficiently similar to it will produce the same patterns of outputs.

The training process, therefore, extracts the statistical properties of the training set and groups similar vectors in to classes. Applying a vector from a given class to the input will produce a specific output vector, but there is no way to determine prior to training which specific output pattern will be produced by a given input vector class. Hence, the outputs of such a network must generally be transformed into a comprehensible form subsequent to the training process. Many unsupervised methods are equivalent to auto-associative supervised methods.

2.2 Supervised training algorithms

2.2.1 Hebb net

The earliest and simplest learning rule for a neural net is generally known as the Hebb rule. Weights are initialized to 0, and weight updates (learning) are performed using the rule

$$w_i(new)=w_i(old)+x_i t$$

where w_i is the weight associated with input x_i and t is the target output. Inputs and outputs may be in binary or bipolar form, but binary does not work as well since the net never learns if the target is 0. The Hebb rule is limited in that it is not guaranteed to give correct outputs after training.

2.2.2 Perceptron

The Perceptron learning rule is a more powerful learning rule than the Hebb rule. Its iterative learning procedure can be proved to converge to the correct weights; those weights that will give the correct output for all input training patterns. One of the assumptions made in this proof is that such weights exist.

The output of the perceptron is $y=f(y_{in})$, where $y_{in}=b+x_i w_i$.

$$f(y_{in})=\begin{cases} 1 & \text{if } y_{in} > \theta \\ 0 & \text{if } -\theta \leq y_{in} \leq \theta \\ -1 & \text{if } y_{in} < -\theta \end{cases}$$

Note that here, there are actually two thresholds, θ and $-\theta$ which define two decision boundaries and three output spaces; 0,1 and -1.The "undecided" band `0' separates the region of positive response from that of negative response. The weights are adjusted only when an error occurs; that is, when the output does not match the target function.

Weights are adjusted according to the formula

$$w_i(new)=w_i(old)+\alpha\ tx_i$$

where the learning rate α ranges by $0<\alpha\leq 1$. As more training patterns produce the correct response, less learning occurs, since by definition weights are adjusted only when an error occurs.

Perceptrons can be applied to problems, which can be linearly separable. Linear seperability limits single layer networks to classification problems in which set of points corresponding to input values can be separated. Perceptrons can't solve problems, which are linearly inseparable such as XOR problem.

2.2.3 Adaline

The Adaptive Linear Neuron (Adaline) is very similar to a perceptron. The net input y_{in} is computed identically, but the activation function for an Adaline is

$$f(y_{in})=\begin{cases} 1 & \text{if } y_{in}\geq 0 \\ -1 & \text{if } y_{in}<0 \end{cases}$$

Weights are updated with the formula

$$w_i(new)=w_i(old)+\alpha\ t(t-y_{in})x_i$$

The differences between the perceptron and an Adaline are that the threshold function is slightly different, and more importantly, that the weight update Δw is proportional to the difference between the target output and actual output, rather than the target output itself. The Δ function attempts to minimize the output error over all training patterns, and like the perceptron, will converge if appropriate weights exist.

A Madaline simply extends the Adaline algorithm to multiple layers. There are various algorithms, which differ slightly in their implementation of weight updates. In either case, training continues over the given set typically until a specified error tolerance has been met or until a preset number of training epochs has been reached.

2.3 Unsupervised training algorithms

2.3.1 Hebbian learning

Hebb proposed a model for unsupervised learning in which the synaptic strength is increased if both the source and destination neurons are activated. In this way, often-used paths in the network are strengthened. Unfortunately, this learning continually strengthens its weights without bound. It increases its network weights according to the product of the excitation levels of the source and destination neurons. The weights are strengthened by

$$w_{ij}(n+1)=w_{ij}(n)+\alpha o_i o_j$$

where o_i and o_j are outputs of neuron i and j respectively.

2.3.2 Kohonen's Self-Organizing Maps

The most novel form of unsupervised learning in the NN literature is Kohonen's self-organizing [8]. SOMs combine competitive learning with dimensionality reduction by smoothing the clusters with respect to a priori grid. The SOM algorithm involves a trade-off between the accuracy of the quantization and the smoothness of the topological mapping. Kohonen networks are similar to a variety of competitive networks. The defining characteristic of these nets is that they choose one or more output neurons that will respond to any given input pattern, instead of providing an output pattern using all output neurons. SOMs are intended not only for pattern recognition but also for clustering, visualization, and abstraction.

2.3.3 Adaptive Resonance Theory

Adaptive Resonance Theory (ART) clusters input vectors using unsupervised learning. There are three layers of neurons; input, interface and output cluster. Between each interface and cluster neuron, there is both a forward and a backward connection. The forward connection weights to the output cluster layer determine the cluster with the largest net input upon presentation of an input pattern; this cluster is chosen as the winner. The backward connection from the cluster layer to the interface layer determines whether the input pattern is similar to that cluster's exemplar vector, if so, only that cluster is allowed to update its weights; if not, the cluster is rejected and a new winner is chosen by repeating the above algorithm.

Thus, the net resonates as learning occurs. Different clusters may be chosen for the same input pattern depending on when it is presented. A stable net will not oscillate among different cluster units during training. A plastic net is able to respond to a new pattern equally well at any stage of learning. This is sometimes difficult to achieve since the learning rate typically changes during training.

2.4 Applications

ANNs are used in many applications such as in pattern recognition, pattern matching, pattern classification, image data compression, data mining, etc. NNs are used in combination with fuzzy systems to build powerful neurofuzzy systems, which can be used in missile tracking and invarious defense applications. NNs are increasingly popular tools for modeling of complex dynamics, noisy signal processing problems. Neural nets learn statistical relations from observations rather than relying on algorithmic solutions. Most standard neural network architectures posses the property of being universal learners.

Neural networks have shown their ability to summarize large databases in many real world applications. A new area is the organization of very large document collections. The best algorithm suitable for solving the problem in organizing very large data sets is SOM, because of its clustering properties. The ability of the SOM to perform complex statistical pattern recognition even in very noisy environments has led to a wide range of applications from analyzing semantic relationships in sentences to robotics, telecommunications and process monitoring and control. The self-organizing map has been used most effectively in the area of speech processing. Many industrial projects use the SOM as a tool for solving hard real-world problems.

Many fields of science have adopted the SOM as a standard analytical tool: statistics, signal processing, control theory, financial analysis, experimental physics, chemistry and medicine. The SOM solves difficult high-dimensional and nonlinear problems such as feature extraction and classification of images and acoustic patterns, adaptive control of robots, and equalization, demodulation, and error-tolerant transmission of signals in telecommunications.

Chapter 3
DOCUMENT REPRESENTATION MODELS

These methods concentrate on models that can be estimated automatically and efficiently based on very large quantities of data at high speeds. They are designed to provide computationally feasible engineering solutions for tasks in which utilizing human labor would be expressive, too slow, or even impossible due to large amounts of data. Typical tasks that the methods are used for in text mining are, accessing a piece of text based on partial or noisy information, ordering items based on similarity, summarizing the content of documents or collections and extracting properties of individual textual items or collecting of them.

3.1 Vector Space Model

Documents are represented as points (or vectors) in an n – dimensional Euclidean space, where each dimension corresponds to word of the vocabulary. The i^{th} component d_i of the document vector expresses the number of times in the word with index i occurs in the document. Furthermore, each word may have an associated weight to describe its significance. The similarity between two documents is defined either as the distance between the points.

The Vector Space Model is most common way to represent text documents in mining text document collection [14]. Vector operations can be performed very fast and efficient. Standard algorithm exists for performing dimension reduction and visualization in vector spaces. An obvious problem with the Vector Space Model is the high dimensionality: the number of different words in a document collection easily rises to hundreds of thousands. The size can be reduced by automatically or manually selecting a subset containing the most important words that still represents the essential characteristics of the documents.

3.2 Latent Semantic Indexing

Relationships between words can be deduced from their occurrence patterns across documents. This notion is utilized in a method called Latent Semantic Indexing, which applies Singular Value Decomposition (SVD) to the document by word matrix to obtain a projection of both documents and words into a space referred to as the Latent space. Dimensionality reduction is achieved by retaining only the latent variables (projection dimensions) with largest variance (largest eigen values). Subsequent distance calculations between documents or terms are then performed in the reduced-dimensional latent space.

The original LSI algorithms had a high computational complexity $O(N^3)$, which is problematic for use with large data sets. The computational complexities of the LSI is known to be $O(Nld)$, where N is the number of documents, l is the average number of different words in each document and d is the resulting dimensionality.

3.3 Random Projection

For many applications and methods, the central aspect is document representation is the distance between documents. It has turned out that an initially high a dimensional data space can be projected on to a randomly selected, much longer dimensional space so that the original distances are nearly preserved. In effect, the exactly orthogonal basis values of the original space is replaced by vectors that are with high probability nearly orthogonal, even with randomly chosen directions if the fixed dimensionality is sufficiently high. A reason for this is the very high-dimensional spaces the number of nearly orthogonal vectors is much larger than the dimensionality of the space.

The advantage of random projection is that it is extremely fast; is an efficient implementation of random projection by pointers the computational complexity is only $O(Nl) + O(n)$, where N is the number of documents, l is the average number of original documents in each document, n is the original dimensionality of the input space. Furthermore it can be applied to any high-dimensional vector representation, and any algorithm that relies on vector distances, can be applied after the random projection.

3.4 Independent Component Analysis

In the ICA model, the data is assumed to be generated as some linear mixture of a set of independent random variables, also called sources, $x=As$, where x is known data, A is the unknown mixing matrix and s is the unknown sources. Various ICA algorithms attempt to estimate the sources as well as the mixing matrix by maximizing a measure of independence of the sources. In ICA, a document is assumed to be generated by a set of independent topics. A single document is represented as a linear combination of the active topics, several of which could be active for a single document. The topics are assumed to differ in their probability density distribution for words.

3.5 Word Clusters

Clustering methods can be used for reducing the number of data by grouping together similar items. In document representation, clustering methods can be applied to group similar words, and then represent documents in terms of word clusters rather than individual words. A clustering that is well suited for document representation should reduce the variation of forms while losing as little information as possible regarding semantic content, especially the topics discussed in a document. In languages with strict restrictions on word orders, such as English, the distribution of words in the immediate context of word contains considerable amounts of information regarding the Syntactic category of the word and mostly with in the syntactic categories information about the semantic category as well.

3.6 Term Weighting

In considering the meaning of a piece of text, some words carry more meaning than others. In addition to a basic division to function words (*e.g.* which, of, and and) and context words (*e.g.* label, sun), some content words seem to target the theme of discussion much more precisely than others. Regardless of which methods is used for dimension reduction or for deducing latent, it is possible to assign weights to the words which attempt to describe how important the word is for the document representation.

A commonly used weighting scheme is the *tf* * *idf,* where *tf* stands for term frequency within the document and *idf* stands for the inverse of the number of documents in which the term appears. The scheme is based on the notion that words that occur frequently in documents are often less significant for meaning and rare words probably carry more meaning. Many variations of the general scheme exist. For example, the weight W_{ij} of a word W_i occurring in document d_j can be calculated as follows:

$$W_{ij} = (1+\log(tf_{i,j}))\ \log(N/df_i)$$

where $tf_{i,j}$ is the frequency of the term i in document j and df_i is the document frequency, *i.e.* the number of documents in which the term i appears. The weighting assigns maximum weight to words that appears only in a single document. Since in the

Vector Space Model term weights directly effect the distance between documents, the results greatly depend on the weighting of terms.

The above global weighting schemes attempt to describe the importance of a word irrespective of its particular context, such a near by words or the location of the word in the document structure. Prior information about the structure of documents may be utilized as well, for example to emphasize title words or words that appear in the beginning of the document.

3.7 Probabilistic Modeling

Probabilistic modeling allows answering questions in terms of probabilities, *e.g.* "how probable is this document in this model", or if the model or if the model has been constructed for document classification, "what is the most probable class for this document". Furthermore, if appropriate prior information about the task or the data exists it can be encoded explicitly using a rigorous mathematical framework.

The binary encoding of documents that disregards word occurrence counts in a document is captured by the Multivariate Bernoulli Independence Model. In other words, a document is assumed to be generated by a collection of discrete, independent random variables, one for each word in the vocabulary. If a certain word appears in the document, the value of the respective random variable is 1, otherwise 0, thus disregarding the frequency of occurrence. The bag-of-words representation is captured by the Multinomial Model, in which each word in a document is assumed to the drawn from a multinomial distribution of words with as many independent trails as the length of the document counted in words.

Chapter 4
DOCUMENT MAPS

Document maps are constructed automatically using SOM. The SOM is used for ›rojecting documents from very high-dimensional spaces on to a two-dimensional map-;rid. The map can be used for visually conveying information about document collection .nd for performing searches for documents [17].

l.1 Self-Organizing Map

Self-Organizing Map (SOM) is the most popular artificial neural network lgorithm with unsupervised learning rule. Amongst all the algorithms and architectures roposed for artificial neural networks the SOM has the special property of being able to xtract invariant features from complex high-dimensional input spaces and create ordered epresentations of them in a lower dimensional image space[13]. Therefore, the main pplication of this algorithm is clustering of data, obtaining a two-dimensional display of ıe input space that is easy to visualize.

SOMs consist of two layers of units: A one-dimensional input layer and a two imensional competitive layer, organized as a 2D grid of processing units called neurons. ι model of some multidimensional observation, eventually a vector consisting of eatures, is associated with each unit. The map attempts to represent all the available bservations with optimal accuracy using a restricted set of models. The models are roduced by a competitive learning process that automatically orders them on the grid long with their mutual similarity. The models become ordered on the grid so that milar models are close to each other and dissimilar models far from each other.

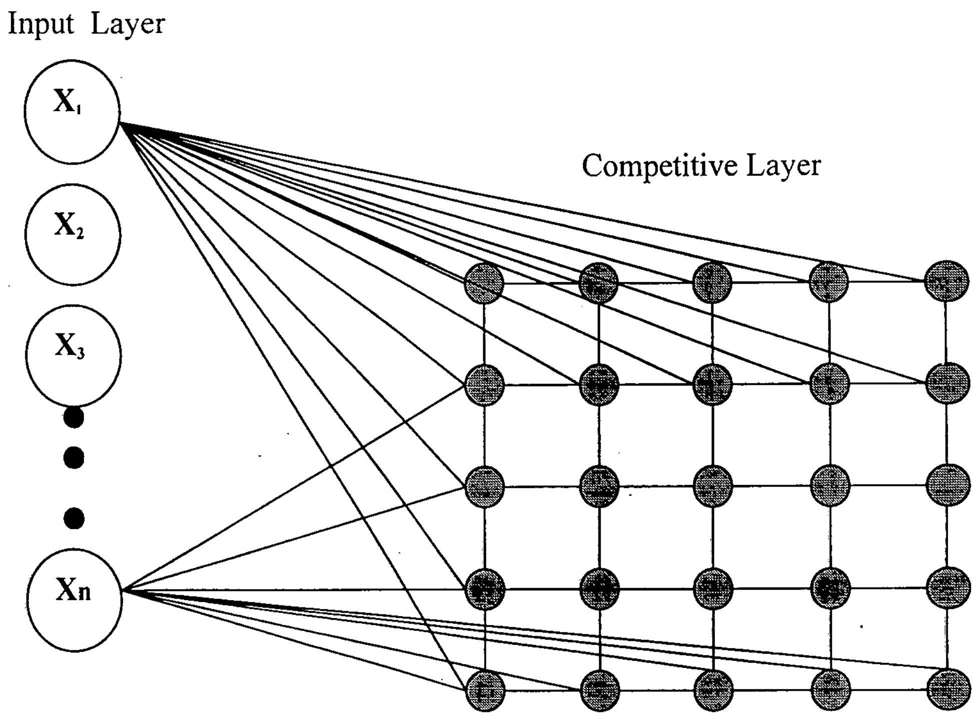

Figure: 4.1: The SOM network

4.2 The Self-Organizing Map Algorithm

Competitive learning is an adaptive process in which the neurons in a neural network gradually become sensitive to different input categories, sets of samples in a specific domain of the input space. A kind of a division of labor emerges in the network when different neurons specialize to represent different types of inputs. The specialization is enforced by competition among the neurons, when input x arrives, the neuron that is best able to represent it wins the competition and is allowed to learn it even better.

The neurons are located on a discrete lattice. The self-organizing map, the competitive learning algorithm can be generalized: if not only the winning neuron but also its neighbors on the lattice are allowed to learn, neighboring neurons will gradually specialize to represent similar inputs, and the representations will become ordered on the map lattice. This is the essence of the SOM algorithm.

The neurons represent the inputs with reference vectors m_i, the components of which correspond to synaptic weights. One reference vector is associated with each neuron called unit. The unit, indexed with c, whose reference vector is nearest to the input x is the winner of the competition. For each sample x, the winner c (best match) is identified by the condition

$$c(x) = \arg\min_{i} \{\|x - m_i\|\} \tag{1}$$

The comparison metric is usually selected as Euclidean. The winning unit and its neighbors represent the input even better by modifying their reference vectors towards the current input. The amount the units learn will be governed by a neighborhood kernel h, which is a decreasing function of the distance of the units from the winning unit on the map lattice.

Th-9474

The neighborhood function is often taken as the Gaussian

$$h_{i,j}(t) = \alpha(t)\exp\left(-\frac{\|r_i - r_j\|^2}{2\sigma^2(t)}\right) \tag{2}$$

where $0 < \alpha(t) < 1$ is the learning-rate factor which decreases with regression steps, r_i and r_j are the vectorial locations on the display grid, and $\sigma(t)$ corresponds to the width of the neighborhood function, which is also decreasing with the regression steps. For computational reasons, $h_{i,j}(t)$ is truncated when $\| r_i - r_j \|$ exceeds certain limit.

During the learning process at time t the reference vectors are changed iteratively according to the following adaptation rule,

$$m_i(t+1) = m_i(t) + h_{c(x),i}(t)\left[x(t) - m_i(t)\right] \qquad (3)$$

where $x(t)$ is the input at time t and $c=c(x(t))$ is the index of the winning unit:

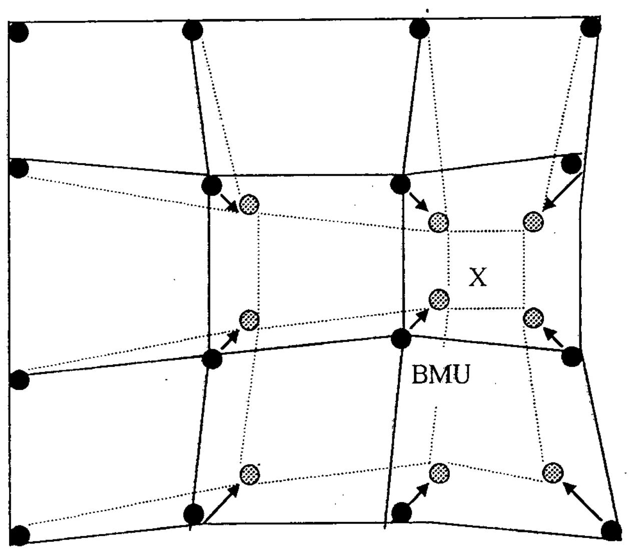

Figure 4.2: Updating the Best Matching Unit (BMU) and its neighbors toward the input sample marked with x. The Solid and dashed lines correspond to the situation before and after updating, respectively.

In practice the neighborhood kernel is chosen to be wide in the beginning of the learning process to guarantee global ordering of the map, and both its width and height decrease slowly during learning. This regression is usually reiterated over the available samples. The optimal solutions m_i are usually obtained in a few iteration cycles, after the discrete valued indexes *c(x)* have settled down and are no longer changed in further iterations.

4.3 Properties Useful in Exploring Data

4.3.1 Ordered display

The ordered nature of the regression justifies the use of the map as a display for data items. When the items are mapped to those units on the map that have the closest reference vectors, nearby units will have similar data items mapped onto them. Such an ordered display of the data items facilitates understanding of the structures in the data set. Kohonen was the first to propose using such displays to illustrate a data set. The same display can be used for displaying several other kinds of information. One clear advantage of always using the same display is that as the analysts grow more familiar with the map, they can interpret new information displayed on it faster and more easily.

For example, the map display can be used as an ordered groundwork on which the original data variables, components of the data vectors, can be displayed in their natural order. The variables become smoothed locally on the display, which helps in gaining insight in the distributions of their values in the data set. Such displays are much more illustrative than linearly organized statistical tables. It might also be useful to display the residuals, average differences of the variables from their smoothed values.

4.3.2 Visualization of clusters

The same ordered display can be used for illustrating the clustering density in different regions of the data space. The density of the reference vectors of an organized map will reflect the density of the input samples. In clustered areas the reference vectors will be close to each other, and in the empty space between the clusters they will be more sparse. Thus, the cluster structure in the data set can be brought visible by displaying the distances between reference vectors of neighboring units.

The cluster display may be constructed as follows. The distance between each pair of reference vectors is computed and scaled so that the distances fit between a given minimum and maximum value, after optionally removing outliers. On the map display each scaled distance value determines the gray level or color of the point that is in the middle of the corresponding map units. The gray level values in the points corresponding to the map units themselves are set to the average of some of the nearest distance values (on a hexagonal grid, e.g., to the average of three of the six distances toward the lower-right corner). After these values have been set up, they can be visualized as such on the display, or smoothed spatially. The resulting cluster diagram is very general in the sense that nothing needs to be assumed about the shapes of the clusters. Most of the clustering algorithms prefer clusters of certain shapes.

4.3.3 Missing data

A frequently occurring problem in applying methods of statistics is that of missing data. Some of the components of the data vectors are not available for all data items, or may not even be applicable or defined. Several simple and more complex approaches have been proposed for tackling this problem, from which all of the clustering and projection methods suffer likewise.

In the case of the SOM the problem of missing data can be treated as follows: when choosing the winning unit by Equation 1, the input vector can be compared with the reference vectors using only those components that are available in. Note that none of the reference vector components is missing. If only a small proportion of the components of the data vector is missing, the result of the comparison will be statistically fairly accurate. When the reference vectors are then adapted using Equation 3, only the components that are available in will be modified.

Better results can be obtained with the approach described above than by discarding the data items from which components are missing. However, for data items from which the majority of the indicators are missing it is not justifiable to assume that the winner selection is accurate. A reasonable compromise used is to discard data items with too many (exceeding a chosen proportion) missing values from the learning process. Even the discarded samples can, however, be tentatively displayed on the map after it has been organized. Although the SOM as such can be used to explore incomplete data sets, some preprocessing methods may have problems with missing components of the input data items. For example, normalization of the data vectors cannot be done.

4.3.4 Outliers

In measurement data there may exist outliers, data items lying very far from the main body of the data. The outliers may result, for instance, from measurement errors or typing errors made while inserting the statistics into a database. In such cases it would be desirable that the outliers would not affect the result of the analysis. This is indeed the case for map displays generated by the SOM algorithm: each outlier affects only one map unit and its neighborhood, while the rest of the display may still be used for inspecting the rest of the data.

Furthermore, the outliers can be easily detected based on the clustering display: the input space is, by definition, very sparsely populated near the outliers. If desired, the outliers can then be discarded and the analysis can be continued with the rest of the data set. It is also possible that the outliers are not erroneous but that some data items really are strikingly different from the rest. In any case the map display reveals the outliers, whereby they can either be discarded or paid special attention to.

4.4 Some variants of SOM

A rich variety of versions of the basic SOM algorithm have been proposed. Some of the variants aim at improving the preservation of topology by using more flexible map structures instead of the fixed grid. But, they cannot be used for visualization, at least not as easily as the regular grid. Some other variants aim at reducing the computational complexity of the SOM. The speed of computation is an extremely important aspect in data mining when vast databases are analyzed.

The search for the best-matching unit can be speeded up by constructing a tree-structured SOM [5], where each level of the tree consists of a separate, progressively larger SOM. The search for the best match then proceeds level by level, at each time restricting the search to a subset of units that is governed by the location of the best match in the previous, smaller level. The map is taught one level at a time, starting from the smallest level. During teaching the best match search can be done even more quickly if the data set is relatively small: the location of the best match in the previous level can be tabulated for each input sample. The subset of units in which the search for a winner needs to be performed can then be found by a single table lookup.

Yet another fast approach is to construct a hierarchical tree of SOMs, where the SOMs in the bottom layer treat different subsets of the components of the input variables. The outputs of the SOMs in the bottom layer are then combined in a hierarchical manner towards the final SOM that takes into account the whole input vector. Since each of the small SOMs receives only very small-dimensional input vectors, the winning unit can potentially be sought with a single, fast table lookup.

4.5 Statistical Accuracy

The question of how large the SOM grid should be has a simple answer if there is an unlimited number of learning samples available is as large as the available computational resources allow .If the data set consists of a finite sample from a larger data set, then the question of the statistical accuracy of the map must be considered, i.e., whether the correct positions of the reference vectors can be estimated accurately based on the available data. The question is especially important if the map is to be used later for displaying new data items. If the new items can be assumed to follow the same distribution as the items that were present while learning, and if the map is statistically accurate, then the new items can be displayed as accurately as the old ones. In high-dimensional spaces it is in general very difficult to achieve sufficient statistical accuracy since the samples are necessarily sparse. By fitting lower-dimensional structures like the SOM grid to the data set in an unsupervised manner, however, there is some hope of a generalizability of the results.

If the goal of data analysis is merely to visualize a given data set for exploratory purposes, and if no new samples need to be mapped later, then the map can be taught to represent the given input data set with a sufficient statistical accuracy.. Then, of course, there is no guarantee that the map will generalize well to new data. The SOM is useful, however, even for maps having more units than there are data items. Even if the map passed through all data items, it would do so in an ordered manner and the distances between close-by data item pairs could be visualized on the map displays .

4.6 Applications

The Self-Organizing Maps have been used in applications such as:

- Automatic speech recognition
- Clinical voice analysis
- Cloud classification from satellite images
- Analysis of electrical signals from the brain
- Organization of and retrieval from large document collections
- Analysis and visualization of large collections of statistical data

Chapter5

IMPLEMENTATION DETAILS

In this dissertation work, a Document Exploration System (DES) has been implemented using Self-Organizing Maps. DES is a full-text information retrieval and exploration method for large document collections. It provides a novel way of navigating through a large collection of text documents.

The functions of DES

- Organizes vast document collections automatically and efficiently according to textual similarities.
- Provides an overview of an unknown document collection to the user.
- Facilitates interactive browsing

In the development of DES, the unsupervised learning algorithm, Self-Organizing Maps (SOM) have been used. SOM is used to create document map. This document landscape provides a good basis for search and exploration. DES includes a navigation interface, which aids the user in making informed decisions about future actions. It consists of the image of the whole document map. Documents can be retrieved by simply clicking with the mouse at the desired location on the map image. A search facility has also been implemented by which suitable starting points for exploration can be located. Two types of searching are provided: content addressable search and keyword search.. This DES can be used as a tool for data mining in textual databases. The overview of construction of DES is show in Figure 5.1

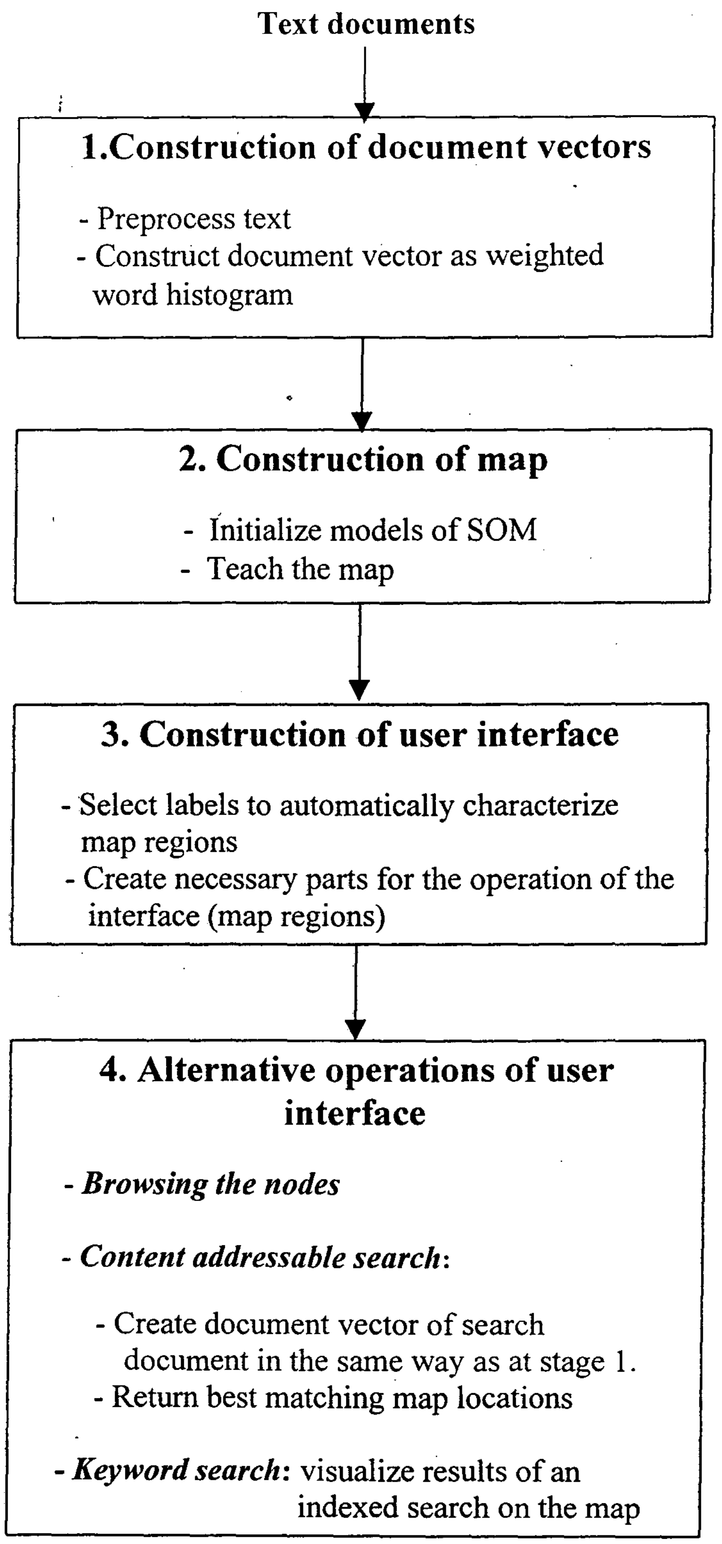
Text documents
1.Construction of document vectors
- Preprocess text
- Construct document vector as weighted word histogram
2. Construction of map
- Initialize models of SOM
- Teach the map
3. Construction of user interface
- Select labels to automatically characterize map regions
- Create necessary parts for the operation of the interface (map regions)
4. Alternative operations of user interface
- Browsing the nodes
- Content addressable search:
- Create document vector of search document in the same way as at stage 1.
- Return best matching map locations
- Keyword search: visualize results of an indexed search on the map

Figure:5.1 Overview of the construction of Document Exploration System

The system consists of three main modules: vector generation module, SOM training module and user interface module. The vector generation module generates document vectors by using vector space model. The resulting vectors constitute input data set for training by SOM. These vectors are passed on to SOM training module. The SOM network is trained over these vectors until it converges. The resulting document map is given to the user interface. User interface module performs the function of retrieving required documents by connecting to the document database on the basis of user's query.

The interaction between the modules is shown below.

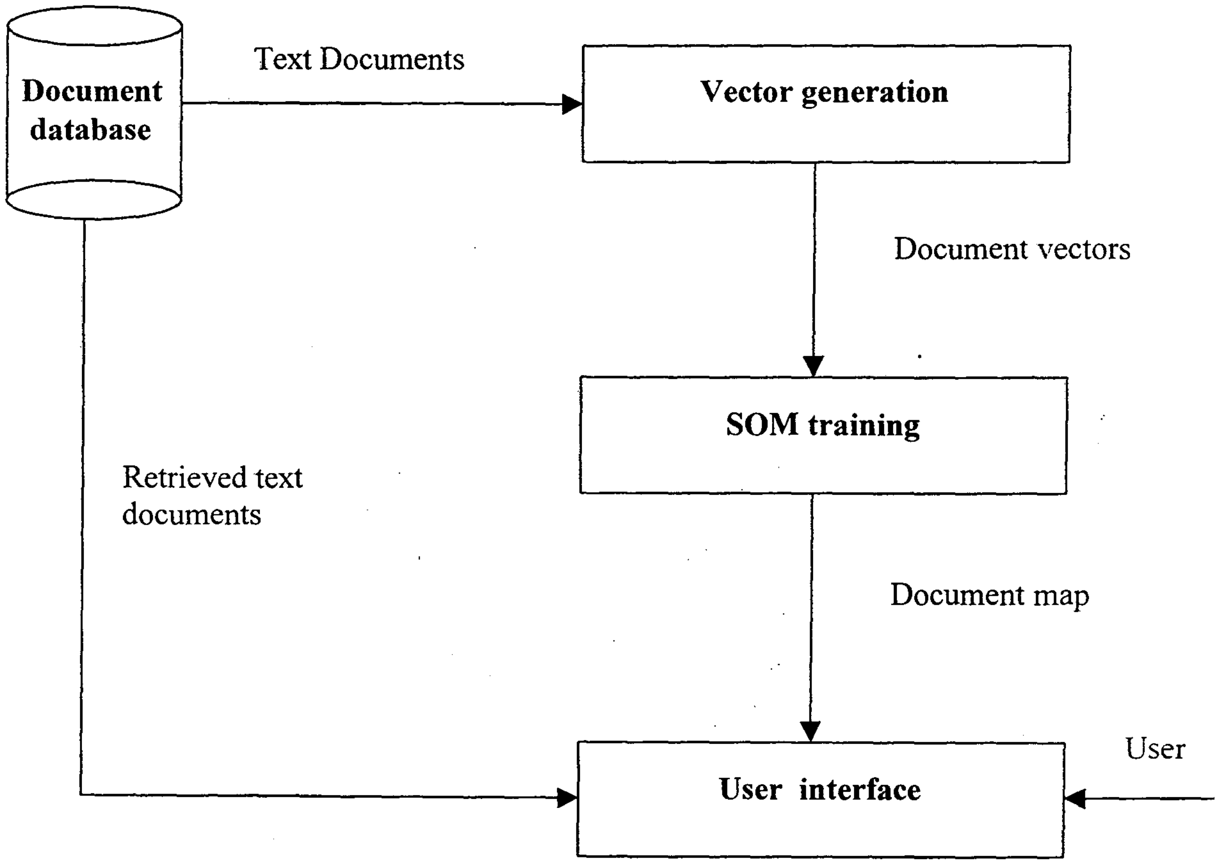

Figure: 5.2 Implementation overview of Document Explorartion System

5.1 Vector Generation

In order to reduce the complexity of the documents and make them easier to handle, the documents have to be transformed from the full text version to a document vector, which describes the contents of the document. Document vectors are content descriptive features for documents. Using the frequency of occurrence of words is a promising technique to find word associations in documents. The vector space model has been adopted to represent documents.

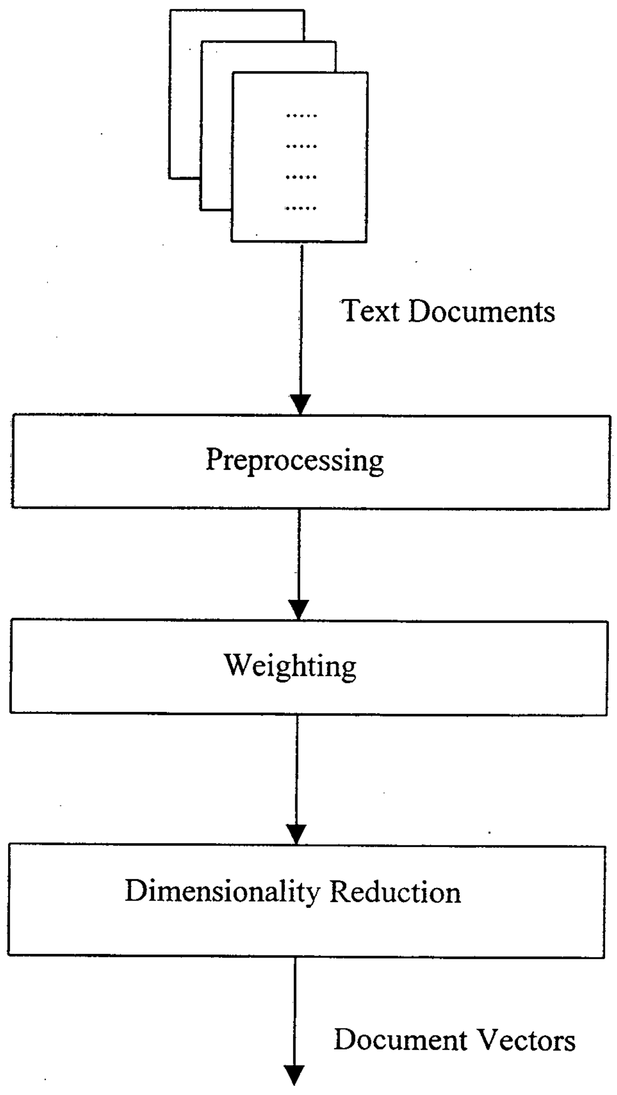

Figure: 5.3 Vector Generation Module

5.1.1 Preprocessing

The non-textual information from the documents is removed and the mathematical symbols, numbers are replaced with dummy tokens. Common words that are listed as stop words are discarded. In general, 40%-50% of the total number of words in a document are removed with the help of a stop list. Also the rarest words are also removed.

5.1.2 Weighting

A common weighting scheme for terms within a document is to use the frequency of occurrence. The term frequency is content descriptive for the documents and is generally used as the basis of a weighted document vector. This weighting scheme is used to discriminate one document from the other.

5.1.3 Dimensionality Reduction

Although preprocessing stage reduces the initial vocabulary, the number of remaining words is still very high. The vector space model is used to reduce the dimensionality. It is obvious that many words in a document will not describe the content. Non significant words are removed, so that document will only be represented by content bearing words. After extracting keywords, the words are assigned weights. These weights represent the occurrence of particular word in the document.

The document vectors of all the text documents are constructed and stored in a database, which is required at the time of training. The keywords are also stored in separate file, which is useful for keyword search.

5.2 Training

The main tasks of this module are clustering of data, obtaining a two-dimensional display of the input space that is easy to visualize. These tasks have been achieved by using Kohonen's Self-Organizing Map training algorithm. The document vectors generated by vector generation module have been used to train the SOM network.

SOM training can be performed with the learning function. Five parameters have to be passed to this learning function:

Learning rate factor:	The initial learning rate factor can vary between 0 and 1. It determines the overall adaptation strength.
Neighborhood size :	The initial neighborhood size is the radius of the neighborhood of the winning unit. All units within this radius are adapted. Values should range between 1 and the size of the map.
Decreasing Factors:	The learning rate factor, neighborhood size decreases monotonically after every iteration. This decreasing is controlled by the decrease factors .
Horizontal size :	Since the internal representation of a network doesn't allow to determine the 2-dimensional layout of the grid, the horizontal size in units must be provided for the learning function. It is the same value as used for the creation of the network..

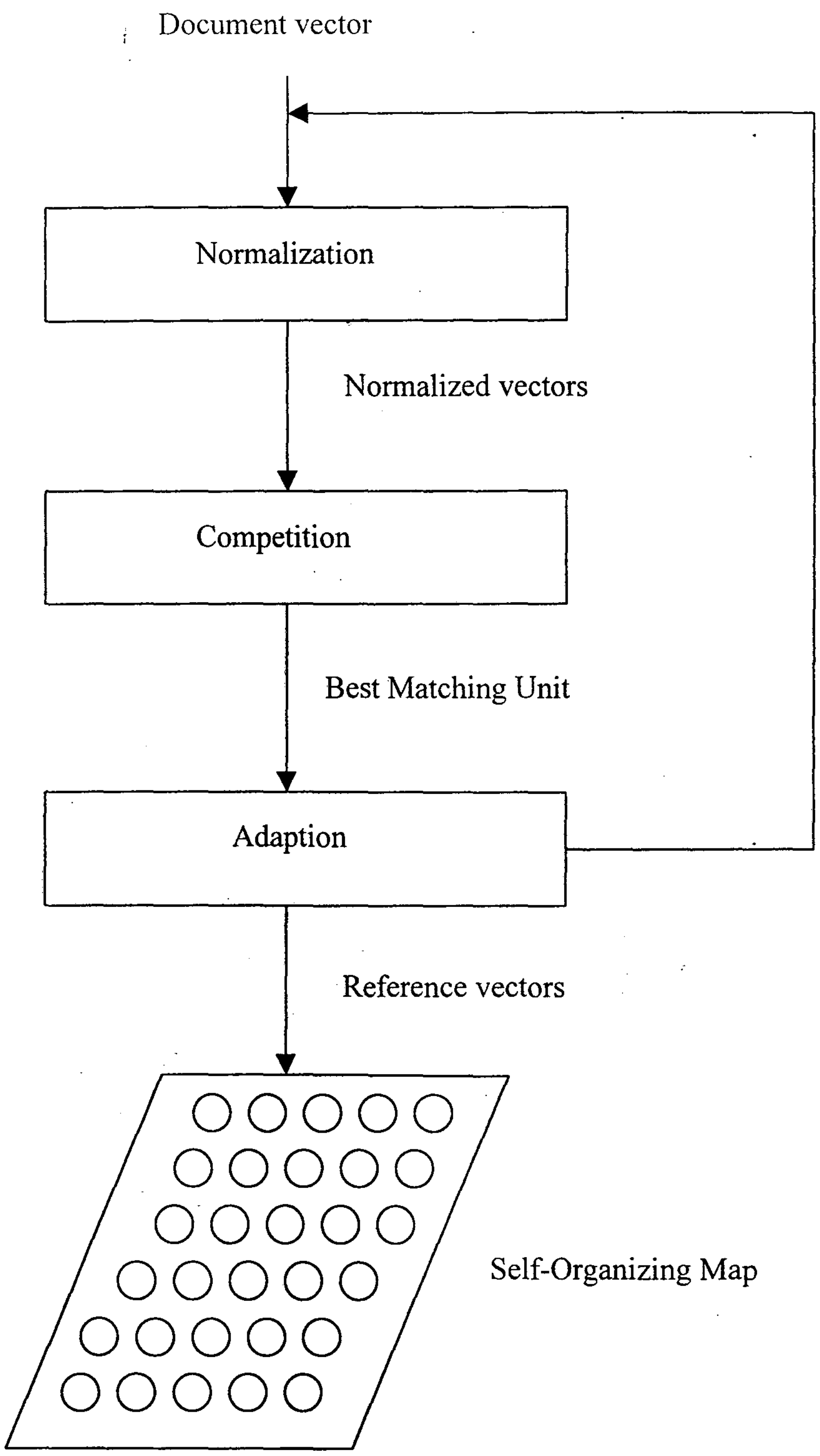

Figure: 5.4 Training module

Steps involved in training

1. Initialization :

Before starting the learning process, it is important to initialize the competitive layer with normalized vectors. The parameters learning rate factor, neighborhood size, horizontal size of map and decreasing factors are also initialized. The topology of the map selected is rectangular.

2. Competition:

The components of a document vector are presented to all competitive units in parallel and the best matching (nearest) unit is chosen as the winner. Since the vectors are normalized, the similarity between the normalized input vector x and the reference units can be calculated using Euclidean distance measure.. The vector most similar to x is the one with the minimum distance with x.

3.Weight Adjustment:

The topological ordering is achieved by using a spatial neighborhood relation between the competitive units during learning, i.e. not only the best-matching vector, but also its neighborhood is adapted. Once the winning unit is known, all the weights of neighboring units in the map are adapted in such a way that units which are geographically closer to the winning unit have larger weight changes than those of units farther away.

4. Step 2 and 3 are repeated for each document vector.

5. Application of all document vectors in the input data set is treated as one iteration. The SOM network is trained over number of iterations until the reference vectors stabilize.

After each iteration, adaption height and radius are decreased to enforce the clustering process.

Once trained, the SOM is ready for loading of the entire text collection. Each node of the map will have a list of documents that are associated to it. Now, the map can be used for browsing.

5.3 User Interface

The purpose of developing the user interface was to explore document maps conveniently. The visualization was to offer a view of the collection that would help in forming a general understanding of the domain as well as to guide exploration towards interesting area.

User interface consists of an image document map. Documents can be retrieved by simply clicking with the mouse at the desired location on the map image. By clicking on a dot, the list of documents associated with the map unit is displayed. The map images were created in advance and stored as static files to minimize calculations while using interface. In the current implementation, both the documents and the contents of the map units are stored in databases.

5.3.1 Document Map

Document map has been visualized using smoothened document density diagram. In this diagram, light color denotes a large number of similar documents and dark color denotes an emptier area. This diagram visualizes the cluster structure. The document map describes document density at each area. It has been labeled with descriptive words that characterize regions of the map. The labels describe the underlying area.

5.3.2 Search Facility

When browsing large document maps, it may be difficult to decide where to start browsing the map. A search facility has been implemented to provide suitable starting points for exploration. Two types of searching have been implemented.

Content addressable search:

The interface to the map has been provided with a form field into which the user can type a query, or a description of interest, in the form of a short document. This query is preprocessed and a document vector is formed in exactly the same manner as for the stored documents prior to the construction of the map. The resulting vector is then compared with the model vectors of all map units, and the list of documents of best matching units is displayed.

Keyword Search:

A more conventional keyword search has also been provided for finding good starting points for browsing. After building the map, for each word, we have indexed the map units that contain the word. User can type a key word, the matching units found from the index and the best matches are returned.

CONCLUSION

The Self-Organizing Maps are suitable for interactive data mining or exploration tasks in which the user either does not know the domain very well or has only a general idea of the contents of the full-text database being examined. Unsupervised models are well suited for text classification. The DES is able to uncover a kind of topological organization of the text material in an unsupervised manner. Its performance in the search task has been evaluated using a small, standard document collection, and found to be comparable with the performance of other search approaches. The visual metaphor of maps seems to offer framework for performing the major tasks studied in the field of text mining, namely, searching, exploration and visualization. Directions for feature work include the development of various text mining tools as well text visualization models.

BIBLIOGRAPHY

[1] Bishop CM, *Neural Networks for Pattern Recognition* Clarendon P, Landon,1995.

[2] D.Merkl, "Text classification with self-organizing maps:Some lessons learned, " *Neurocomputing,*vol .21,pp.61-77,1998.

[3] D.Roussinov and H.Chen, "A scalable self-organizing map algorithm for textual classification neural network approach to thesaurus generation, " *C-AI-Commun., Cogn.Artif.Intell.,*vol.15,pp.81-111,1998.

[4] Elaine Rich, Kevin knight, *Artificial Intelligence* second edition, Tata McGraw-Hill Publishing Company Limited, New Delhi, 1995.

[5] Fausett L, *Fundamentals of Neural networks: Architectures, Algorithms and applications,* Prentice Hall, Englewood Clifs, 1994.

[6] H.Chen, C.Schuffels, and R.Orwig, "Internet categorization and search: self-organizing approach, " *J. Vis.Commun.Image Represent.*,vol. 7,no.1, pp.88-102,1996.

[7] J.C.Scholtes, "Unsupervised learning and the information retrieval problem, "in *Proc. IJCNN '91,Int.Joint.Conf.Neural Networks*, vol.1 Singapore,1991, pp.95-100.

[8] Juha vesanto and Esa Alhoneiemi,"Clustering of the self -organizing map," *IEEE transactions on neural networks* ,vol.11,No.3,May 2000.

[9] K.Lagus and S.Kaski, "Keyword selection method for characterizing text document maps, " in *Proc.ICANN99,Ninth Int. Conf.Artificial Neural Networks*,vol. 1,1999, pp.371-376.

[10] Olli Simula juha Vesanto,"The self organizing map in industry analysis," *Industrial applications of neural networks,* edited by L.C.Jain and V.Vemuri.

[11] Sankar A and Mammone R J (1991) *Neural Networks: Theory and applications* Ed. by Mammone R and Zeevi Y. Academic Press: SanDiego pp 281-302.

[12] Stamatios V. Kartalopoulos *Understanding neural networks and fuzzy logic* Basic concepts and application, *IEEE, Inc., New York.*

[13] S.Kaski, T.Honkela, K.Lagus, and T.Kohonen, "Creating an order in digital libraries with self-organizing maps", in *Proc. WCNN '96 World Congr.Neural Network*, San Diego, CA,Sept.15-18,1996,pp.814-817.

[14] Teuvo Kohonen, Samuel Kaski, Krista Lagus, Jarkko Salojarvi,Jukka Honkela,Vesa Paatero, and Antti Saarela,"Self -Organization of massive document collection," *IEEE trans.Neural Networks,* vol.11,NO.3,May 2000.

[16] T.Kohonen, "Comparison of SOM point densities based on different criteria, " *Neural Comput.*,vol.11,no.8,pp.2171-2185,1999.

[17] T.Kohonen ," Exploration of very large databases by self -organizing maps," in *Proc.ICNN '97, Int.conf.Neural Networks*,Houston,TX 1997,pp. PL1-PL6.

[18] T.Kohonen, "Things you haven't heard about the Self-Organizing Map, " in *Proc.ICNN'93, Int .Conf. Neural Networks*,1993,pp.1147-1156.