SELF-ORGANIZING MAPS

Abstract

One of the functions of competitive neural networks is to graphically visualize data. Tuevo Kohonen developed Self Organizing Feature Maps to do just that. The algorithm behind SOM is to determine the winning neuron than apply Euclidean distance measurements to calculate and assemble the neurons in a high-dimensional space. The classic algorithm has been improved constantly in order to facilitate and rapidly form the map. Some approaches to improve the algorithm have been by changing the learning process by adding kernel functions, allowing for the use of matrices instead of vector data, and also layering the SOM itself. Some of these new algorithms include Bayesian learning, EM algorithms, Fuzzy SOMs and Growing Hierarchical SOMs. Bayesian learning has proved to be faster and more efficient than EM. Some of these algorithms have problems such as not being able to converge fast enough and having trouble normalizing the input vectors. For now it seems that the GHSOM algorithm is the best SOM algorithm developed.

Introduction

Kohonen's Self-Organizing Maps is a neural network that is similar to the structure of the brain. Their application is used to demonstrate patterns but not the way Adaptive Resonance Theory is used. SOM is used to observe patterns of topology. The architecture of SOM is related to the brain and the way the neurons in the brain are arranged. Self-organizing maps come in two different arrangements, hexagonal and rectangular [1]. Each shape contains forty-nine units. Each unit in the rectangle has eight neighbors but six in the hexagon. Neighborhoods do not wrap around from one side of the grid to the other. The cluster units are linear. There are m cluster units, which are arranged in a one or two-dimensional array, the input signals are n-tuples. The weight vector for a cluster unit serves as input patterns associated with that cluster. During the self-organization process, the cluster unit whose weight vector matches the input pattern most closely is chosen as the winner. The winning units and its neighboring units update their weights accordingly. The weight vectors of neighboring units are not close to the input pattern.

The classic algorithm of SOM is similar to other competitive networks. As usual the algorithm is based on weight changes. The weights are initialized, as are the learning rates and neighborhood parameters [2]. The square of the minimum Euclidean distance is computed then all the units within a specified neighborhood the new winning unit is the sum of the old unit and the product of the learning rate multiplied by the distance between the old and new winning unit. The learning rate is then updated. The radius of the neighborhood is then shortened at certain times. The stopping condition is tested.

The algorithm of SOM determines how the map will appear. Besides the classical algorithm there are Bayesian, Expectation Maximization, and Growing Hierarchical Self

Organizing Map Algorithms. The latter is an extension of the classic SOM algorithm. The soft topographic vector quantization (STVQ) and its extensions are the kernel based soft topographic mapping (STMK) and soft topographic mapping for proximity data (STMP) are based on Euclidean distances as is the classic algorithm [3]. However, these algorithms use fixed neighborhood functions to encode desired relations between neighboring units [4]. All of these algorithms are used to design Self-Organizing Maps using different techniques and mathematical functions. These algorithms are used for different applications and some are generally better than others.

Methods

To begin the classic algorithm, Kohonen's SOM, has a learning rate that slowly decreases over time, as does the radius of the neighborhood around a cluster unit [5]

The map then forms. Kohonen's SOM has been used for certain character recognition problems. Mainly though its been used to see a certain path that would use the minimum number of units. This has also been known as the traveling salesman problem [6].

The Bayesian SOM uses a different approach to learn the data. The BSOM is used to improve the classic algorithms classification ability [7]. The algorithm begins not by setting parameters but by having a certain number (k) of units in the input space. Now each unit has specific parameters that designate its weight. The parameters consist of a mean vector, covariance matrix, and prior. At each time a sample is taken from the input data. The winning unit is chosen from the estimated posterior probability. Then the weights of the neighboring units are update according to the following mathematical equations. The winner and neurons inside the neighborhood of the winner adapt to this input by the corresponding scales apart from the normal learning rate. Each neuron in the BSOM learns a model-based pattern distribution. The BSOM naturally estimates the mixture of modeled probability density functions in an unsupervised manner [8]. The mathematical equation for determining the winning unit is as follows:

$$\hat{P}[\omega_k|\mathbf{x}(n),\hat{\theta}_k] = \frac{p[\mathbf{x}(n)|\omega_k,\hat{\theta}_k]\hat{P}(\omega_k)}{\sum_{j=1}^{K} p[\mathbf{x}(n)|\omega_j,\hat{\theta}_j]\hat{P}(\omega_j)}$$

Then the weights are updates in a fix-sized neighborhood of the winner, according to the following self -organizing rules[9]:

$$\Delta\hat{\mathbf{m}}_k(n+1) = \alpha(n)\hat{P}[\omega_k|\mathbf{x}(n),\hat{\theta}_k][\mathbf{x}(n) - \hat{\mathbf{m}}_k(n)]$$

$$\hat{\Sigma}_k(n+1) = \hat{\Sigma}_k(n) + \alpha(n)\hat{P}[\omega_k|\mathbf{x}(n),\hat{\theta}_k]\left\{[\mathbf{x}(n) - \hat{\mathbf{m}}_k(n)][\mathbf{x}(n) - \hat{\mathbf{m}}_k(n)]^T - \hat{\Sigma}_k(n)\right\}$$

$$\Delta\hat{P}(\omega_k|n+1) = \alpha(n)\{\hat{P}[\omega_k|\mathbf{x}(n),\hat{\theta}_k] - \hat{P}(\omega_k|n)\}$$

The reason that the updating can be limited to a small (and fixed size) neighborhood of the winner is that the topological ordering property, which can always be preserved when the mapping is in the same dimension, and the locality of the Gaussians. So,

$$p(\mathbf{x}|\hat{\Theta}) \approx \sum_{k \in \eta_\nu} p(\mathbf{x}|\omega_k, \hat{\theta}_k)\hat{P}(\omega_k)$$

$p_i(\boldsymbol{x} \mid \theta_i)$	ith component conditional density;
θ_i	parameter vector for the ith conditional density, $i=1, 2, ..., K$;
Θ	$(\theta_1, \theta_2, ..., \theta_K)$;
P_i	prior probability of ith component

The BSOM algorithm has been compared to the Expectation Maximization algorithm because of its similarity. [10]

The Expectation Maximization Algorithm can function in two ways, with missing input values and without missing input values [11]. The EM algorithm begins by expectation; this step minimizes the value of free energy functional, F (P, W) with respect to probability assignments P given weight parameters W. The maximization step minimizes the value of F (P, W) with respect to parameters W given assignments P [12]. However when there are missing values another approach is taken. The expectation step takes on a Gaussian probability distribution over the missing inputs given the average weight. The maximization step is based on the minimization of free energy with respect to parameters W with fixed P. The only difference between the EM algorithm with missing values and without missing values is the parameter used for filling in the unknown input is the average weight not the original weight [13].

Another SOM algorithm is called the Growing Hierarchical SOM (GHSOM). This is an extension of the classic Kohonen SOM. This algorithm aims to add layers to the feature map. Each layer has its own SOM. Layer 0 consists of a single-unit SOM. Layer 1 consists of a 2x2 SOM and for each unit in this layer's SOM, additional SOM may be added. The same structure applies to lower level SOMs [14]. The GHSOM will grow in 2 dimensions, in width (by increasing the size of each SOM) and in depth (by increasing the levels of SOM). The algorithm begins the same way the classic algorithm begins by initializing the weights of each unit with random values. Compute the minimum square of the Euclidean distance of each unit. The unit with the largest deviation between its weight and input vectors is chosen as the winner. Then the hierarchy comes in. Insert a row or column between the winner and the most dissimilar neighbor unit in terms of input space. These steps are repeated until the quantization error, the sum of the distances between the weight vector of a unit i and the input vectors mapped onto this unit, is t1 fraction of the qi, average quantization error, of the unit i in the preceding layer of the hierarchy. The general idea is to keep checking whether the lowest level SOMs has achieved sufficient coverage for the underlying input data. Check each unit's qi to ensure it is above a certain threshold t2. Assign a SOM layer to each unit with qi greater than t2 and train SOM with input vectors mapped to this unit [15].

Another extension of the classic SOM is the Fuzzy SOM. The Fuzzy SOM [16] unlike the classic SOM uses if-then rules on the input-output data. The FSOM learns both the centers of clusters and deviations around the centers. The algorithm begins by initializing the parameters. Set the value of Qki to zero for each output node. For each k and each ai and Ci=Ck update the parameters. for T times. During this process add the value Fki (ai) to Qki in each output node of the FSOM layer. Determine the degrees of confidence in the rules by wki=Qki/sum of m to i=d Qkd. A total of L FSOMS are

initialized, the fuzzy weight between the jth input node and the ith output node of the kth FSOM correspond to the jth fuzzy membership function of the ith rule with the kth output value in its then part. Then a lower layer with only one node is added to each FSOM and makes the weight between the ith node and the above output node in the kth FSOM correspond to the degree of confidence for the ith rule with the kth output value. This entire quantity is known as Fuzzy Inference Network (FIN). The output value of each FIN is defined as the weighted sum of the outputs from the FSOM layer and the weights between the FSOM layer and the output node [17]. Then the output values of the entire FIN are normalized. The degrees of confidence of the rules are determined by normalizing the total degree of firing in each output node in the FSOM.

The next three algorithms are similar in their architecture and overall application. The three algorithms are the soft-topographic vector quantization algorithm (STVQ); the kernel based soft topographic mapping algorithm (STMK) and the soft topographic mapping for proximity data algorithm (STMP).

STVQ uses the Euclidean distance measure where as the STMK does not. STMK and STMP are branches of STVQ [18]. STVQ begins by assuming a cost function. This cost function takes its minimum with respect to the parameters to be determined when the desired state of the mapping is reached. The minimization of the cost function will automatically yield a set of parameters fit for the data. The cost function relies on the given data vectors x , model vectors, the neighborhood function, and binary assignment variables that are set to one if the data vector x is assigned to node r and the rest are set to zero. The cost function is the squared Euclidean distance from the data vector to the corresponding model vectors weighted by the neighborhood function. The cost is lowest when the model vectors are as close as possible to their assigned data vectors and also when their neighbors in the map are assigned to similar data vectors. Therefore neighboring nodes represent neighboring volumes of data space. The optimization of the cost function depends on both binary variables and continuous variables. This poses a problem during the optimization technique. Therefore an important part of the algorithm known as deterministic annealing allows the probability distribution over the parameter space to be calculated directly and not estimated by a sampling process. The deterministic annealing process is performed by cost function parameterized by beta. Beta determines the amount of smoothing that is done on the original cost function. [19]. At low values of beta the cost function is smoothed to a degree that only one global minimum remains; this is determined by an EM algorithm. When beta is large the cost function is reflected in the free energy. Deterministic annealing begins by determining the minimum of the free energy at low values of beta and attempts to track the minimum through higher values of beta, it continues to do so until the global minimum of the free energy at beta approaching infinity coincides with the global minimum of the original cost function. The minimization of the free energy is achieved in two nested loops. In the inner loop the equations can be solved by fixed-point iteration for a given value of beta. The assignment probabilities P of x are calculated on the previous estimate of the model vectors [20]. This loop ends when the absolute value of the change in the position of the model vector is less than 10^{-5} for data normalized to the unit. This procedure constitutes and EM algorithm. In order to find the global minimum of the cost function, beta is varied in the outer loop according to an annealing process. It starts at a small value of beta and increases stepwise until a good minimum of the original cost function is found at high

values of beta. This minimum should correspond to a topologically ordered state of the map representing the data. An exponential annealing technique is used to accomplish this. The STVQ algorithm uses many techniques seen in other algorithms. It uses fuzzy techniques, EM algorithms, and a new technique known as soft SOM. Soft SOM allows quick computations of large sets of model vectors [21]. During the annealing process the model vectors remain at the center of mass of the data vectors up to a certain value of beta. At that point the representation undergoes a transition and the model vectors split up in a data space. The initial value of beta should be just slightly higher than the value of beta where the model vectors remain. That value is calculated by the eigenvalue of the covariance matrix of the data along the principal axis [22]. The annealing process of the STVQ has no trouble in finding the globally optimal solution. The deterministic annealing technique allows the relations between the nodes to be encoded and this does not interfere with the optimization process. Therefore the entire algorithm is as follows initialize the model vectors randomly, calculate lookup takes for h, choose beta start, beta final, annealing facto and convergence criterion, while beta is less than beta final calculate EM algorithm, E step, M step, until beta approaches beta multiplied by annealing factor.

The next algorithm called Kernel Based Soft Topographic mapping (STMK) uses new distance measures in data space based on kernel functions. This is similar to the STVQ but the extension allows the algorithm to perform in a high-dimensional space. Therefore is does not use the usual Euclidean calculations. STMK also uses deterministic annealing. The whole idea between STMK is that its mapping techniques allow the data to be seen in a way that is not possible using Euclidean representation. The model vector in this space not only represents the mean of its assigned data vectors but also the mean of the pair wise correlations of their components [23]. This changes the distances between the data vectors and leads to different assignments of data vectors to nodes. The quantization is performed in feature space not data space. The cost function for kernel-based topographic mapping relies on the idea that the model vectors are expressed as linear combinations of the data vectors according to the an equation where the coefficients are replace the model vectors as map parameters. The kernel function it self is calculated in data space instead of feature space. There are three kernel functions used the polynomial kernel, which corresponds to a mapping into product spaces of an assigned degree. The sigmoid kernel is chosen in analogy to the sigmoid transfer function. The last kernel is the RBF kernel, which is isotropic and changes the expected range of the nodes in data space. Generally a mapping to a space where the kernel function acts as a dot product exists if the kernel function is a continuous kernel of a positive integral operator. This condition shows that the quantization can be performed in a suitable space whenever the corresponding kernel function is known [24]. The kernel function works better than the dot product because it allows the algorithm to work in a high-dimensional product space. The neighborhood function is free to encode desired neighborhood relations between the nodes. The annealing parameter and the convergence parameter use similar values as in STVQ. The parameters of the kernel functions should be chosen by trial and error, but should be chosen in accordance with the range and dimensionality of the data. Assuming normalized data of dimensionality the polynomial kernel takes values of degree from one to six. The sigmoid kernel values should be no higher than 1.5. The width of the RBF kernel function should be in the range of 0.1 to 1.

Therefore the entire algorithm goes as follows. Initialize parameters randomly, calculate the lookup table, chose a kernel function, calculate lookup table for that kernel function at time t from the data, choose beta start value and beta final value and the annealing factor and convergence criterion. Then while beta start is less than beta final repeat the EM algorithm, calculate the E step and M step until beta approaches the annealing factor multiplied by beta [25]. The difference between STMK and STVQ is simply the kernel function employed in the initialization. The annealing process is calculated by an EM algorithm in both algorithms.

The last algorithm, the Soft Topographic for Proximity data (STMP), uses a matrix instead of vectors. The matrix consists of mutual proximities or dissimilarities. Some examples are asking subjects to determine the pair wise dissimilarities of a set of colors or speech recognition [26]. Each color is a data item, which is characterized only by its dissimilarities to other colors. A cost function is used to topographically map the data. The elements of the dissimilarity matrix are added to the cost, whenever both data items are associated with the same nodes. The neighborhood function again makes sure that data items, which are similar, are mapped to nodes, which are close to one another on the grid. The overall cost does not increase with the number of items assigned to a node. This is necessary in order to obtain a coherent representation of the data. Soft assignments of data items to nodes where the partial assignments costs need to be determined in terms of the neighborhood function and the dissimilarity data [27]. Then a mean-field is used to determine the assignment costs and make the simplifying assumptions of zero self-dissimilarity and symmetry of the dissimilarities. The former assumption is justified by the fact that the assignment costs is defined only up to an additive constant, which leaves the cost function unchanged. The cost function is invariant under certain conditions. The equations used for the cost function is similar to the kernel functions. The soft assignments and the weighting coefficients of the two algorithms have the same form while the equations for the partial assignment costs and the mean fields are the same. The algorithm is as follows. Initialize the parameters randomly, calculate the lookup table, prepare dissimilarity matrix from data, and choose beta start and beta final, annealing factor and convergence criterion, while beta start is less than beta final, calculate EM algorithm, set beta to beta start multiplied by annealing. The parameters annealing factor and convergence criterion can be chosen in the same range as for STVQ [28]. As in STVQ, the neighborhood function is kept constant during the optimization process and can be chosen freely to encode desired relations between the nodes. STMP is able to perform a mapping from data items that are characterized by general mutual dissimilarities to nodes that are characterized by general neighborhood couplings.

All of the algorithms described above have their own applications.

Data

The classic algorithm has been used to look at patterns in music, and in letters. Particularly though the classic algorithm has been used to generate a spanning tree. The tree makes it easier to identify the results. The results do not resemble a tree but more a wave of lines encasing the patterns. Another application of the classic Kohonen SOM is used for finding the shortest trail between certain elements. It also shows useful that

changing the weights within a small region of the input space does not change the results [29]. Other uses for the classic SOM is clinical voice analysis, monitoring the condition of industrial plants and processes, cloud classification from satellite images, analysis of electric signals from the brain, organization of and retrieval from large document collections, analysis and visualization of large collections of statistical data. In a biological aspect the emerging field of functional genomics has needed a technique to organize the data from gene expression analysis into a meaningful classes and correlate these with other biological datasets. Self-Organizing maps are currently being used to mine gene expression data [30].

Bayesian SOM has been used to judge the uncertainty of predictions, to choose the appropriate network architecture, and how to adapt to the characteristic of the data [31]. The Bayesian SOM converges very fast [32] therefore it proves useful for the data mining of gene expression as discussed before. Bayesian techniques have been used by biologists to classify tissue samples and attempt to find structure in the genes themselves.

The EM algorithm has been seen in other SOM techniques as well. The EM algorithm designed for SOM is widely used when data is missing. The algorithm In the EM method, the E step takes an average of the log likelihood function over the missing variables, while the M step maximizes this likelihood with respect to each parameter [33]. The EM algorithm has been used as a statistical analysis approach to biological data. For example scientists have taken samples of salamanders and tried to analyze the mating success between different species [34]. The effect of each salamander is assumed random while the effect of the species cross is assumed fix. The algorithm was run for one hour during which it performed forty-six EM iterations. EM algorithms have proved useful in climate data analysis. The EM algorithm allows accurate estimates of the missing data therefore giving researchers results to something without any. Schneider uses ridge regression and inverse matrices to compute the EM algorithm. The EM algorithm cannot only be applied to incomplete datasets containing values of a surface temperature at various grid points but it can also be used to construct historic surface temperature from proxy data [35].

Fuzzy SOMS are widely used in genetic algorithms particularly dealing with crossover events in chromosomes. Crossover occurs in genes that are on the same chromosome and not too far apart from one another. Crossover allows different gene arrangements to come into a population [36]. The chromosomes are represented by bit strings, then from these bit strings the fitness of the individual is calculated. The chromosomes consist of values of the center and width of a fuzzy membership function. One fuzzy rule corresponds to one numerical chromosome using an array of numerical parameters from the Gaussian membership functions [37]. The FSOM determines the phenotype of the individual. The phenotype is the way the genes represent themselves in the individual. Basically the individual learns the input-output data using fuzzy competitive learning by representing the chromosome as a starting point of learning, and then the fitness of the individual is evaluated based on the learning result [38]. The approach is to initialize the arrays randomly and transfer the parameters of the rules in the chromosomes into weights, then calculate the fitness based on the parameters of the rules in the FSOMs, then calculations are done to figure in crossover and mutation, then the weights are modified based on fuzzy competitive learning, then these steps are repeated a certain number of times and the degrees of confidence in the rules by normalizing the

final fitness values among the chromosomes in the population are calculated. Fuzzy SOMS are used in this way to see which population will adapt to a certain environment faster than another and which population will be better suited for a certain environment.

Growing Hierarchical Self Organizing Maps have been used quite efficiently in looking through text documents. Much like a search engine is used to surf the Internet, GHSOM is used to sort through large documents. The words of the document are represented numerically. The words are listed in a template vector; some modifications to the words are made in order to reduce their content. The template vector as one can imagine is quite large. Therefore in order to go around this common everyday words like the and is, are removed from the vector. Sometimes the program is created to remove words that are going to be used in the articles many times. For instance if the article is about neural networking, the word computer might be removed. The second part of this GHSOM algorithm is that a vector description is created for each document. Every document is described by the words in it. Therefore to reduce this to numerical representation if the word is in the document the weight vector is a one and zero otherwise. Another approach is to have a counter that would keep track of how many times a word is present in the document. A word is considered more important if it occurs very often in the document, these words receive a low weight. The next step is to normalize the unit length to make up for different document lengths. The GHSOM part comes in when the SOM is trained. The parameters are set and a hierarchical structure begins to form. One particular GHSOM algorithm being used for document archive clustered the archive into seven layers [39]. There are a few branches, each branch relates to a certain topic. Since the archive is very large some larger clusters are represented by two neighboring units that are in the first layer. Therefore items that overlap do not have to be in two different layers. GHSOM has proved very useful in organizing large amount of documents. One most interesting use for GHSOM is its attempt to detect and classify abnormalities of artificial hearts [40]. The GHSOM looks specifically at the aortic pressure signal measured from an artificial heart. The network is made up of two different SOMs. The first clusters the aortic pressure beat patterns then the input vector of the second SOM is associate with a class vector, which is used to output the weights. GHSOM has proved to be a very efficient algorithm for clustering and classification.

The Soft Topographic Vector Quantization algorithm is not used in a specific application but more so for analyzing phase transitions. Pattern recognition and signal processing tasks often involve high-dimensional data, which are hard to visualize, and cannot be processed directly. It is very important also to fins some mapping of the high-dimensional input space to a lower dimensional space in a way that shows the spatial relations of the data [41]. The phase transitions occur during the annealing process. The phase transitions occur in the parameter beta, when the cluster centers are located at the center of mass of the data, beta takes on a value that leaves the representation unstable. Phase transitions also occur when the dimensionality of the array is less than the dimensionality of the data. This then leads to the automatic selection of feature dimensions. Also studied using STVQ is the affects of reverse annealing. During this time the map gradually unfolds until an ordered state is reached.

Kernel-based Soft Topographic Mapping (STMK) is used in another fashion. STMK is used primarily to represent clustering in a very high dimension which in a nonlinear map. STMK has been used to analyze handwritten digits. Four maps were

generated. One map uses polynomial kernel functions while the other three use radial basis kernel functions. The maps show that if the digit does not vary when written then the digit is coded in a high density [42]. The reason STMK is used over STVQ in this application is because the kernel functions allow for simple computations. There are other applications to STMK such as document clustering, however it has been shown that GHSOM is more accurate and faster for this application.

Soft Topographic Mapping for Proximity Data is used for data items that are not in a Euclidean data space. This application is most useful for data such as psychology, linguistics and economics. Studies have been used STMP to map the cerebral cortex of felines. The map shows the connections of the visual region.

Discussion

One important aspect is to see if one algorithm is sufficient to perform all Self Organizing applications. The Growing Hierarchical SOM algorithm is faster than the Soft Topographic Mapping Algorithms and the Bayesian Algorithm is faster than the EM algorithms. Therefore perhaps an algorithm that incorporates the convergence of the Bayesian algorithm with the dimensionality of the GHSOM algorithm would be able to perform any task involving clustering. The algorithm would still be hierarchical but each individual SOM layer would be performed under Bayesian learning. It has already been shown that both the GHSOM and Bayesian SOM algorithms are used in the classification of gene expression. Just as the EM algorithm is used in Soft Topographic Mapping both kernel based and vector quantization, the Bayesian algorithm could be applied to the GHSOM algorithm.

The Bayesian algorithm is more efficient than the EM algorithm because it converges faster. The reason behind that is the convergent variance index of the EM algorithm is always higher than that of the Bayesian. Another reason the Bayesian algorithm is more efficient is that the Bayesian algorithm can escape from local minima whereas the EM algorithm has trouble doing so [43].

The difference between the algorithms is basically how the learning is processed. The classic Kohonen SOM uses Euclidean distance as the measurement. Then the minimum distance is chosen as the winner. The classic SOM is shown using a hexagonal or rectangular grid. Then when the SOM is trained it can result in a spanning tree of data. This is shown by the letters recognition example. The other application is the traveling salesman problem. The classic SOM is used to find the smallest trail to a certain point from a certain unit. Algorithms that do not use Euclidean distance as a tool for measurement such as the STMP are more useful for analyzing pair wise dissimilarities by using matrices. The other Soft Topographic SOMS such as STVQ and STMK use EM algorithms to perform the annealing process. The only difference between the two is that the kernel based functions make simpler computations.

Fuzzy SOMS are different than most clustering algorithms. Fuzzy SOMS use if then statements to learn both the centers of clusters and deviations around the centers. Fuzzy SOMS have been used in collaboration with genetic algorithms to calculate fitness values. One problem with Fuzzy SOM much like EM algorithms is local minimum and correctness rate of the rules change very quickly. All of the algorithms discussed above

have their own uses however some are more efficient than others in performing these applications.

Conclusion

All of the algorithms described above Bayesian, EM, Fuzzy, Growing Hierarchical, Soft Topographic Vector Quantization, Kernel-Based Soft Topographic Mapping, and Soft Topographic Mapping for Proximity Data are based on self-organization and are used to visualize data. Some applications to the algorithms include but are not limited to classifying genes and tissue samples, organizing large amounts of document based data, visual representation of the cerebral cortex, and examining defects of artificial hearts. Each algorithm differs in its mathematical computation of the vectors. The classic algorithm uses Euclidean distance whereas the STMP algorithm cannot because the data is arranged in a matrix. The learning process in the algorithms of STVQ and STMK use an EM algorithm. Both the expectation and maximization steps are identical. However, the initialization of the STMK uses appropriate kernel functions. The GHSOM has performed the task of layering the SOM, which demonstrates how the clustering and overall output of the SOM can be improved. Self Organization Feature Maps prove to be a very important and useful neural network tool.

References

1. Laurene Fausett- Fundamentals of neural networks:architectures, algorithms and applications- Prentice Hall (1994) p.169

2. Laurene Fausett- Fundamentals of neural networks:architectures, algorithms and applications- Prentice Hall (1994) p.170

3. Thore Graepel, Klaus Obermayer, Matthias Burger- Self Organizing Maps: Generalizations and New Techniques Neurocomputing (1998) p.1

4. Thore Graepel, Klaus Obermayer, Matthias Burger- Self Organizing Maps: Generalizations and New Techniques Neurocomputing (1998) p.1

5. Laurene Fausett- Fundamentals of neural networks:architectures, algorithms and applications- Prentice Hall (1994) p.172

6. Laurene Fausett- Fundamentals of neural networks:architectures, algorithms and applications- Prentice Hall (1994) p.182

7. Hujun Yin, Nigel M. Allinson- Comparison of a Bayesian SOM with the EM Algorithm for Gaussian Mixtures
<http://citeseer.nj.nec.com/cache/papers/cs/544/http:zSzzSzsoft.ee.umist.ac.ukzSzhujunzSzmypublicationszSzp_wsom97.pdf/comparison-of-a-bayesian.pdf>

8. BSOM and EM Algorithm
<http://images.ee.umist.ac.uk.hujun/Currentwork2/BSOMandEM.html>

9. BSOM and EM Algorithm
<http://images.ee.umist.ac.uk.hujun/Currentwork2/BSOMandEM.html>

10. BSOM and EM Algorithm
<http://images.ee.umist.ac.uk.hujun/Currentwork2/BSOMandEM.html>

11. Tom Heskes, Jan-Joost Spangers,Wim Wiegenick- EM Algorithms for Self-Organizing Maps
<http://citeseer.nj.nec.com/cache/papers/cs/12242/ftp:zSzzSzftp.mbfys.kun.nlzSzsnnzSzpubzSzreportszSzHeskes.emsom.pdf/em-algorithms-for-self.pdf > p.1

12. Tom Heskes, Jan-Joost Spangers,Wim Wiegenick- EM Algorithms for Self-Organizing Maps
<http://citeseer.nj.nec.com/cache/papers/cs/12242/ftp:zSzzSzftp.mbfys.kun.nlzSzsnnzSzpubzSzreportszSzHeskes.emsom.pdf/em-algorithms-for-self.pdf > p.1

13. Tom Heskes, Jan-Joost Spangers,Wim Wiegenick- EM Algorithms for Self-Organizing Maps

<http://citeseer.nj.nec.com/cache/papers/cs/12242/ftp:zSzzSzftp.mbfys.kun.nlzSzsnnzSzpubzSzreportszSzHeskes.emsom.pdf/em-algorithms-for-self.pdf > p.4

14. The Growing Hierarchical Self Organizing Map
<http://www.comp.nus.edu.sg/~cs5242/bestreports/tiantaipeng/report.htm>

15. The Growing Hierarchical Self Organizing Map
<http://www.comp.nus.edu.sg/~cs5242/bestreports/tiantaipeng/report.htm>

16. Tatsuya Nomura, Tsutomu Miyoshi- Fuzzy SOMS- An adaptive fuzzy rule extraction using hybrid model of the fuzzy self-organizing map and the genetic algorithm with numerical chromosomes – Journal of Intelligent and Fuzzy Systems Vol.6 (1998) p.1

17. Tatsuya Nomura, Tsutomu Miyoshi- Fuzzy SOMS- An adaptive fuzzy rule extraction using hybrid model of the fuzzy self-organizing map and the genetic algorithm with numerical chromosomes – Journal of Intelligent and Fuzzy Systems Vol.6 (1998) p.3

18. Thore Graepel, Klaus Obermayer, Matthias Burger- Self Organizing Maps: Generalizations and New Techniques Neurocomputing (1998) p.3

19. Thore Graepel, Klaus Obermayer, Matthias Burger- Self Organizing Maps: Generalizations and New Techniques Neurocomputing (1998) p.4

20. Thore Graepel, Klaus Obermayer, Matthias Burger- Self Organizing Maps: Generalizations and New Techniques Neurocomputing (1998) p.6

21. Thore Graepel, Klaus Obermayer, Matthias Burger- Self Organizing Maps: Generalizations and New Techniques Neurocomputing (1998) p.7

22. Thore Graepel, Klaus Obermayer, Matthias Burger- Self Organizing Maps: Generalizations and New Techniques Neurocomputing (1998) p.8

23. Thore Graepel, Klaus Obermayer, Matthias Burger- Self Organizing Maps: Generalizations and New Techniques Neurocomputing (1998) p.9

24. Thore Graepel, Klaus Obermayer, Matthias Burger- Self Organizing Maps: Generalizations and New Techniques Neurocomputing (1998) p.10

25. Thore Graepel, Klaus Obermayer, Matthias Burger- Self Organizing Maps: Generalizations and New Techniques Neurocomputing (1998) p.11

26. Thore Graepel, Klaus Obermayer, Matthias Burger- Self Organizing Maps: Generalizations and New Techniques Neurocomputing (1998) p.12

27. Thore Graepel, Klaus Obermayer, Matthias Burger- Self Organizing Maps: Generalizations and New Techniques Neurocomputing (1998) p.13

28. Thore Graepel, Klaus Obermayer, Matthias Burger- Self Organizing Maps: Generalizations and New Techniques Neurocomputing (1998) p.14

29. Laurene Fausett- Fundamentals of neural networks:architectures, algorithms and applications- Prentice Hall (1994) p.185

30. HUT-CIS-Research-SOM <http://www.cis.hut.fi/research/som-research/>

31. What is Bayesian Learning
<http://www.faqs.org/faqs/ai-faq/neural-nets/part3/section7.html>

32. BSOM and EM Algorithm
<http://images.ee.umist.ac.uk.hujun/Currentwork2/BSOMandEM.html>

33. BSOM and EM Algorithm
<http://images.ee.umist.ac.uk.hujun/Currentwork2/BSOMandEM.html>

34. Richard A Levine, George Casella- Implementation of the Monte Carlo EM Algorithm- Journal of Computational and Graphical Statistics Vol.10 No.3 (2001) p.12

35. Tapio Schneider - Analysis of Incomplete Climate Data:Estimation of Mean Values and Covariance Matrices and Imputation of Missing Values - Journal of Climate Vol.14 (March 1 2001) p.869

36. Tatsuya Nomura, Tsutomu Miyoshi- Fuzzy SOMS- An adaptive fuzzy rule extraction using hybrid model of the fuzzy self-organizing map and the genetic algorithm with numerical chromosomes – Journal of Intelligent and Fuzzy Systems Vol.6 (1998) p.3

37. Tatsuya Nomura, Tsutomu Miyoshi- Fuzzy SOMS- An adaptive fuzzy rule extraction using hybrid model of the fuzzy self-organizing map and the genetic algorithm with numerical chromosomes – Journal of Intelligent and Fuzzy Systems Vol.6 (1998) p.6

38. Tatsuya Nomura, Tsutomu Miyoshi- Fuzzy SOMS- An adaptive fuzzy rule extraction using hybrid model of the fuzzy self-organizing map and the genetic algorithm with numerical chromosomes – Journal of Intelligent and Fuzzy Systems Vol.6 (1998) p.8

39. The Growing Hierarchical Self Organizing Map(GHSOM)
<http://www.ifs.tuwien.ac.at/~mbach/ghsom>

40. Xian Zheng Wang, Makoto Yoshizawa, Akira Tanaka, Ken-ichi Abe, Tomoyuki Tambe, Shin-ichi Nitta- Automatic detection and classification of abnormalities for Artificial Hearts using a Hierarchical Self-Organizing Map -Thoughts and Progress Artif Organs Vol.25 No.2 (2001) p.1

41. Matthias Burger, Thore Graepel, Klaus Obermayer- Phase Transitions in Soft Topographic Vector Quantization -Physical Review E 56[4] (1997) p.1

42. Thore Graepel- Statistical Physics of Clustering Algorithms (April 1998) <http://citeseer.nj.nec.com/graepel98statistical.html> p.58

43. BSOM and EM Algorithm <http://images.ee.umist.ac.uk.hujun/Currentwork2/BSOMandEM.html>