FSKD'02

Proceedings of the

1st International Conference on Fuzzy Systems and Knowledge Discovery

HUBEY

VOLUME 1

Lipo Wang,
Saman Halgamuge,
and Xin Yao

November 18 - 22, 2002

FSKD'02

Proceedings of the

1st International Conference on Fuzzy Systems and Knowledge Discovery

Computational Intelligence for the E-Age

Volume 1

lil alan
tail yel alay
wind land

Lipo Wang, Saman Halgamuge, and Xin Yao (Editors)

November 18 - 22, 2002

Orchid Country Club, Singapore

Organized by

Nanyang Technological University

School of Electrical & Electronic Engineering
Nanyang Technological University, Singapore

Sponsored by

Asia-Pacific Neural Network Assembly
Singapore Neuroscience Association
SEAL & FSKD Conference Steering Committees

In-Cooperation with

IEEE Neural Networks Society
International Neural Network Society
European Neural Network Society
SPIE

Supported by

Lee Foundation

SINGAPORE EXHIBITION & CONVENTION BUREAU

U.S. Asian Office of Aerospace Research & Development (AOARD)

U.S. Army Research Office-Far East (ARO-FE)

NOVARTIS
Novartis Pharmaceuticals

SALFORD SYSTEMS

NEURODESIGNER™
Comprehensive Neural Network Software

Proceedings of the 1st International Conference on Fuzzy Systems & Knowledge Discovery (FSKD'02) vol. 1
Lipo Wang, Saman Halgamuge, and Xin Yao (Editors)

ARITHMETIC AS FUZZY LOGIC, DATAMINING AND SVMs

H.M. Hubey
Professor, Department of Computer Science
Montclair State University
Upper Montclair, NJ 07043
hubeyh@mail.montclair.edu

ABSTRACT

Multiple layered neural networks can be trained to solve specific problems but the weights are difficult to interpret. The problem of choosing an architecture for a neural network has been replaced today by the problem of choosing a suitable kernel for an SVM. To identify the smallest set of features that still conveys the essential information contained in the original attributes, e.g. dimension reduction is still needed [1]. Aside from being useful for finding the nonlinear mapping into feature space, the method outlined here (i) computes the nonlinear transformation from the data into a target space, and (ii) generates dimension reduction, (iii) performs classification/categorization, (iv) performs function estimation, (iv) is comprehensible in terms of fuzzy logic, (v) performs unsupervised clustering, (vi) automatically produces association rules, and (vi) provides for a distance-based hashing access for efficient data retrieval for k-nearest neighbors type data processing. The dimension reduction is analogous to that of dimensional analysis of physics, especially fluid dynamics [18], [27]; an example of application to speech can be found in [9]. Results of dimensional analysis in biology can be found in [20]. As a side product of this method, one can unify datawarehousing, datamining and database, perform streaming datamining, and execute fast nonlinear regression. The core of the algorithm/method consists of Boolean minimization coupled with an approximation algorithm and special kind of fuzzy logic that can be found in [4]. The method is patent pending 60/294314 [28].

1. INTRODUCTION

There are a set of related problems in the fields of datamining, knowledge discovery, and pattern recognition. We don't know how many neurons should be in the hidden layer or the output layer. Thus if we attempt to use ANNs for clustering as a preliminary method to finding patterns we must use heuristic methods to determine how many clusters the ANN should recognize. This is just another view of the problem in datamining of knowing how many patterns there are in the data and how we would go about discerning these patterns. There is a related problem in k-nearest-neighbors clustering in which we need an appropriate data structure to be able to efficiently find the neighbors of a given input vector. Indeed, before the k-neighbors method can be used to classify an input vector we need to be able to cluster the input vectors and an ANN might have been used for this process. The problem of knowing how many patterns (categories or classes/clusters) there are is an overriding concern in datamining, and in unsupervised artificial neural network training.

Datamining is based on what was called pattern recognition [26]. One way of classifying the types of pattern recognition is via (i) classification and (ii) estimation [2]. Typically classification is used to create a set of discrete, finite classes, whereas estimation is taken to be an approximation of some desired numerical value based on an observation. The boundaries are not very crisp since estimation consisting of a large number of integer values may just as easily be thought of as categorization or classification. This is especially true if the measured quantities (input data) do not consist of interval or ratio-scaled values [2], [15],[16].

Clustering is the process of grouping data into classes, heaps or categories so that objects within a cluster have high similarity in comparison with one another, but are very dissimilar to objects in other clusters. *Dissimilarities* are assessed based on the attribute values describing the objects. Often *distance measures* are used. Clustering is an unsupervised activity, or should be. Clustering can be thought of as the preprocessing stage for much of datamining. An automated clustering algorithm may be said to be the goal of datamining since *classification* and *prediction* algorithms can work on the clusters. Similarly *association rules* may be derived from the clusters.

2. FUZZY LOGIC

For example, it is easily seen that the implication $A \Rightarrow B$ really implements the relationship $t(A) \le t(B)$; that is, the conditional is false if the truth value assigned to A is not greater than or equal to that of B. It's common in discrete multivalent logics and infinite valued logics to define the truth-value of a variable as that of itself; that is $t(A)=A$ and $c(t(A))=1-A$ (where c(.) is the complement function). To produce crisp logic from these valuations then, or even to produce a fuzzy logic, the simplest function to use is a threshold type of function, say the Heaviside Unit Step Function, H(A).

Instead of simply producing complementation via a threshold we've defined it via its truth-valuation. It is desirable that the complement function c(.) satisfy these properties [14]:

2c.1) *Boundary Conditions* $c(0)=1$ and $c(1)=0$
2c.2) *Monotonicity* $\forall a,b \in [0,1]$ [if $A<B$ then $c(A) \ge c(B)$]
2c.3) $c(A)$ *is continuous*
2c.4) *Involutive* $c(c(A))=A$ for all $A \in [0,1]$

Some desirable properties that the OR (union, from now on represented as $u(x,y)$) should possess are:

2u.1) $u(A,B)$ should satisfy the B.C.
2u.2) *Commutative:* $u(A,B) = u(B,A)$

$B(1+0)$ ↑ different & same

eq.

$I(\wedge(1,1))=1$ ← significance

$I(\vee(0,0))=1$

Here

$I(\wedge(\text{big}\wedge\text{big} \vee \text{small}\wedge\text{small}))=1$

$I(\vee(\text{small}\vee\text{small}))$ etc

2u.3) *Monotonic* $(B \le B') \Rightarrow u(A,B) \le u(A,B')$
2u.4) *Associative* $u(u(A,B),C) = u(A,u(B,C))$
2u.5) $u(A,B)$ *is continuous*
2u.6) *Superidempotent* $u(A,A) > A$
2u.7) *StrMon* $(A < A') \wedge (B < B') \Rightarrow (u(A,B) < u(A',B))$
(where StrMon stands for Strictly Monotonic).

The conditions for the i(A,B) [intersection] are similar to above.
2i.1) $i(A,B)$ should satisfy the B.C.
2i.2) *Commutative:* $i(A,B) = i(B,A)$
2i.3) *Monotonic:* $(B \le B') \Rightarrow i(A,B) \le i(A,B')$
2i.4) *Associative:* $i(i(A,B),C) = i(A,i(B,C))$
2i.5) $i(A,B)$ *is continuous*
2i.6) *Subidempotent:* $i(A,A) < A$
2i.7) *StrMon* $(A < A') \wedge (B < B') \Rightarrow (i(A,B) < i(A',B'))$

3. MULTIPLICATIVE FUZZY LOGIC

Many things measured in the real world are non-negative numbers. That is true especially for all underdeveloped sciences. We can create a fuzzy logic for $[0,\infty]$ to enable the interpretation of some operations in this domain. However we have to create an appropriate complementation.

If we want to take some guesses as to what kinds of laws of logic are impeccably true and should be preserved, the three that are commonly put forward as candidates are:

1a) $x+c(x)=1$ *The Law of the Excluded Middle (LEM)*
1b) $c(x \cdot c(x)) = 1$ *The Law of Noncontradiction (LNC)*
1c) $c(c(x)) = x$ *The Law of Involution or Self-Inverse (LSI)*

where $c(x)$ is the negation or complement function. LNC is usually written as $x \cdot c(x)= 0$ and called the Law of Contradiction. It's not necessary to have infinite valued logics satisfy relationships that should only be valid for crisp bivalent logic. The functions defined above satisfy a more general type of the *Law of the Excluded Middle* and the *Law of Noncontradiction*. Differentiation of (1) yields (where prime indicates derivative);

2a) $c'(x) = -1$
2b) $c(x) + x \cdot c'(x)= 0$

Equations (2a) and (2b) have the solutions

3a) $c(x) = k - x$
3b) $c(x) = k/x$

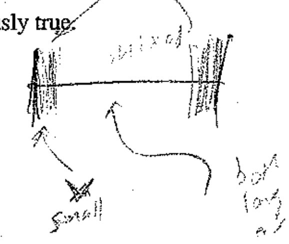
respectively, where k is a constant of integration. Eq.(3a) is the commonly used fuzzy complement with k=1. It is (3b) that is used here in *multiplicative logic*. We can relax the constraints given in (2) and still produce logic-like [or infinite-valued or fuzzy logical] results [8].

3c.1) The variables are not in [0,1] but instead in $[0,\infty]$, therefore proceeding formally according to the axioms of attractive rings [8] $c(0) = 1/0 = \infty$, and $c(\infty) = 1/\infty = 0$.

3c.2) $\forall a,b \in [0,\infty]$ $(A < B) \Rightarrow c(A) \ge c(B)$
WLOG (without loss of generality) for $y < x$, $1/y \ge 1/x$ QED

3c.3) c(A) is continuous
3c.4) $c(c(A)) = A$ $\forall A \in [0,\infty]$. Obviously true.

The desirable properties for the union:



3u.1) $u(A,B)$ should satisfy the B.C.
3u.2) $u(A,B) = u(B,A)$
3u.3) $A \le A'$ *implies* $u(B,A) \le u(B,A')$
3u.4) Clearly $Max(Max(A,B),C) = Max(A,Max(B,C))$
3u.5) $u(A,B)$ *is continuous*
3u.6) $u(A,A) > A$ *superidempotent*
If this condition were $u((A,A) \ge A$ [*super*]*idempotent*) it would probably be better since the best we can do with the $Max(A,B)$ is idempotency.
3u.7) $A < A' \wedge B < B' \Rightarrow u(A,B) < u(A',B)$

If the condition is $A < A' \wedge B < B' \Rightarrow u((A,B) \le u(A',B))$ it is satisfied since $A < A' \wedge B < B' \Rightarrow u((A,B) \le u(A',B))$ e.g. for $A < A' \wedge B < B'$ then $Max(A,B) \le Max(A',B)$.

Similar comments apply to the intersection.

3i.1) $i(A,B)$ should satisfy the B.C.
3i.2) commutative $i(A,B) = i(B,A)$
3i.3) monotonic $B \le B' \Rightarrow i(A,B) \le i(A,B')$
3i.4) associative $i(i(A,B),C) = i(A,i(B,C))$
$Min(Min(A,B),C) = Min(A,Min(B,C))$ is obviously true.
3i.5) $i(A,B)$ *is continuous*
3i.6) $i(A,A) < A$ *subidempotent*
If the condition were $i(A,A) \le A$ e.g. [sub]idempotent, it would probably be better since the best we can do with the $Min(A,B)$ is idempotency.
3i.7) StrMon $A < A' \wedge B < B' \Rightarrow i(A,B) < i(A',B)$

This condition $A < A' \wedge B < B' \Rightarrow (A,B) \le u(A',B)$ is satisfied since for $A < A' \wedge B < B'$ then $Max(A,B) \le Max(A',B)$.

We can also prove:

Theorem: The Max() and Min() functions as norm and conorm s satisfy De Morgan's laws with the complement $c(x) = 1/x$.

Proof: WLOG assume x>y, $u(x,y) = Max(x,y) = x$ and $i(x,y) = Min(x,y) = y$

Then $\neg Max(x, y) = Min(\neg x, \neg y) = Min(1/x, 1/y) = 1/x$
But directly $\neg Max(x, y) = \neg x = 1/x$

Also, $\neg Min(x, y) = Max(\neg x, \neg y) = Max(1/x, 1/y) = 1/y$
and direct computation shows that $\neg Min(x, y) = \neg y = 1/y$

QED

There is only a minor problem in actual computation, and that is not to use the interval as $[0, \infty]$ but the interval $[\varepsilon, \infty]$ where we can make $\varepsilon \approx 0$ or $\varepsilon \ll 1$, in other words, as close to zero as possible. In practice the numbers can be normalized especially for datamining so that there is room for "outliers" without having to renormalize (*see Section 7*). Therefore there is no reason not to interpret the nonnegative numbers that often show up in neural networks and datamining as fuzzy logics. This brings us to the fact that "dimensionless groups" or physics and fluid dynamics are products of variables, and therefore the "clusters" found by datamining techniques should be products. Also from dimensional analysis we know that dimensional analysis performs dimension reduction admirably, whereas in datamining, we are forced to rely on linear-algebra (or linear) techniques for dimension reduction and feature extraction. If we switched to "multiplication" and left the world of addition and linearity behind, especially using the fuzzy logic shown here clustering and datamining would be much easier and more productive. The system HUBSAN [28] method does exactly this (and also with other fuzzy logics). It has been explained in [4,5,6,11,12]. Furthermore we can create smoother versions of Min() and Max() operators [8].

4. IDEMPOTENT AND CONTINUOUS MAX-MIN

Consider the functions given below

4) $$H_h(x, y) = \left(\frac{1}{2}\right)(x+y)^{h+1}$$

5) $$M_m(x, y) = 2^{m-1}\left(\frac{(x-y)}{\sqrt{(x-y)^2}}\right)^{m+1}$$

from [8]. It's easy to see that the $H_0(x,y)$ takes one as its maximum value, and zero for its minimum for $0 \le x,y \le 1$. In addition, for h=1 it's linear in both x and y. The function $M_0(x,y)$ is always positive, symmetric and zero for x= y. It's essentially $(x-y)^2/\text{abs}[2(x-y)^2]$ or abs(x-y)/2 since the positive square root is taken in the denominator by convention. Hence the elementary operations are being used to define the abs(·) function instead of defining abs(·) as a primitive operation or a primitive function. The plots for these functions for some values of the parameters m and h are shown in Figure 1 below.

It can be seen that $H_0(x,y)$ is a plane and that $M_0(x,y)$ is not smooth because of the discontinuity of the derivative along x=y. However, the algebraic sum and differences of these functions are exactly Max(x,y) and Min(x,y).

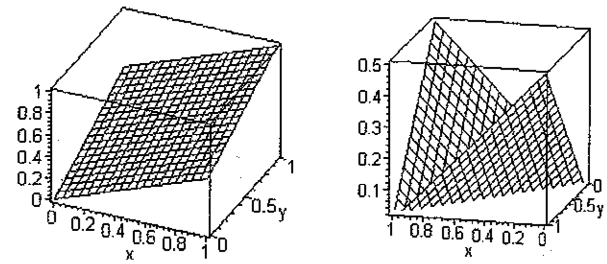


Figure 1: $H_0(x,y)$ and $M_0(x,y)$ respectively.

It's easy to verify that

6a) $$Max(x, y) = H_0(x, y) + M_0(x, y)$$

6b) $$Max(x, y) = H_0(x, y) - M_0(x, y)$$

It can be seen that Max(x,y) and Min(x,y) are not the only continuous and idempotent functions that implement union or intersection. In particular, since $M_m(x,y)$ is zero along the diagonal (i.e. x=y) and $H_1(x,y)$ increases linearly along the diagonal from zero to one (i.e. is idempotent) then all functions of form

7a) $$V_m(x, y) = H_0(x, y) + M_m(x, y)$$

7b) $$A_m(x, y) = H_0(x, y) - M_m(x, y)$$

are idempotent and continuous since the form of $H_0(x,y)$ guarantees it, but only the max-min functions as defined in Eqs. (6) are also associative. Analogous results can be obtained for other values of the parameters m, and h. It can be seen that $V_2(x,y)$ and $A_2(x,y)$ do not satisfy the crisp B.C. along the edges but only at the points corresponding to the binary values {0,1}. Other binary connectives and even nonstandard logic-like functions can be derived from the basic functions $H_h(x,y)$ and $M_m(x,y)$. For example the functions

8) $$U_{mn}(x, y) = [H_0(x, y) + M_m(x, y)]^n$$

9) $$I_{mn}(x, y) = [H_0(x, y) - M_m(x, y)]^n$$

are continuous but not idempotent and satisfy the boundary conditions not only at the end points but along the *edges*. By dropping the idempotency requirement even more flexibility is gained. However,

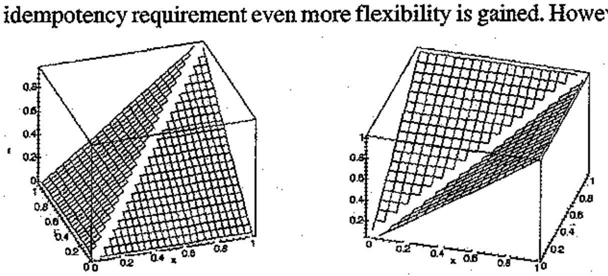


Figure 2: Min(), and Max() in terms of $H_0(x,y)$ and $M_0(x,y)$

we can use a fuzzy operator, that is an operator that is neither OR nor AND. Such a form can be created easily from any fuzzy norm and conorm. For example, if we use the standard Zadeh norm and conorm $max(x, y)$ and $min(x, y)$ functions we can create a fuzzy operator (a fuzzy norm/conorm) via

$$10) \qquad F(x, y) = \rho \cdot Max(x, y) + (1 - \rho) \cdot Min(x, y)$$

which can be found in [6]. Obviously, a fuzzy-operator can be created exactly the same way from any norm/conorm. Using the functions in Eq. (1) and (2) we can create a fuzzy operator

$$11) \quad F(x, y) = H_0(x, y) + \xi M_0(x, y) \quad \text{where} \quad \xi \in [-1, 1]$$

Therefore we can treat ξ as a fuzzy variable and use it in regression-like operations in cases in which we do not know in advance if the relationships sought should be multiplicative (power laws and AND-like relations) or additive (OR-like relations). Obviously there is nothing to stop anyone with experimenting with mixing fuzzy-logical {*,+} arithmetic {*,+}. It is quite easy to set up other fuzzy operators from any truth valuation type as shown above. As an example, suppose the data from some datamining project yielded the K-map as given in Fig. (3a). The grouping/clustering via approximation gives the result in Fig. (3b). The simplification of the K-map (KH-map) of Fig (3b) is

$$12) \qquad F = \bar{x}_2\bar{x}_3\bar{x}_4 + x_1\bar{x}_2\bar{x}_3 + \bar{x}_1x_3x_4$$

one minterm for each group/cluster. Each minterm in Eq (12) represents an edge (or a hyperplane) on the binary hypercube.

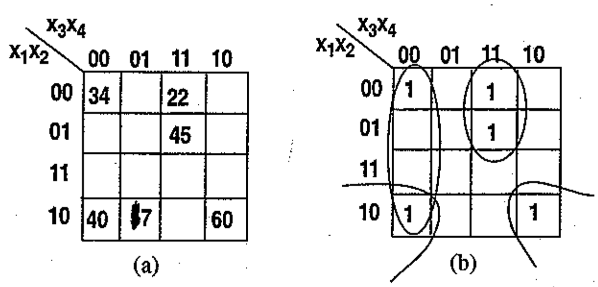


Figure 3: K-map simplification. The K-map in (b) is assumed to be derived from some data, such as (a) after thresholding the number of occurences of each possibility. The reduction to {0,1} and grouping is shown in (b).

This equation is the set of association rules for this problem at the threshold of 22. Each minterm also represents a cluster. The interpretation of this minimization varies according whether the clustering was supervised or unsupervised. The minterms are also the analogues of the dimensionless groups used in the Buckingham Pi Theorem [27], [4]. The circuit in Fig (4) represents Eq. (5). There are two ways the results of the foregoing can be interpreted. Eq. (12) can be interpreted as the result of an supervised clustering/datamining method that is the top-level clustering of data and hence the association rule(s) of the data. In the supervised mode all

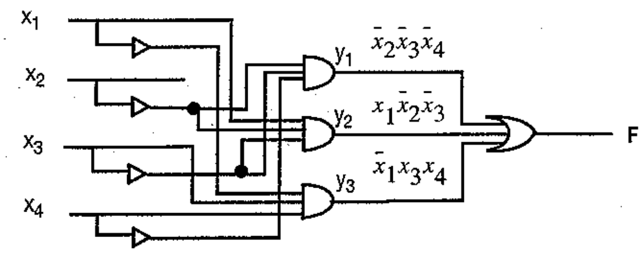


Figure 4: Minimization. This circuit represents the logical assocation rules that is produced from the data via the K-map method. The second stage where the y's are OR'ed is optional depending on purpose [i.e. supervised or unsupervised].

the minterms may represent one gigantic [nonspheroidal] nonlinearly separable cluster.

5. DATAMINING VIA MINIMIZATION

A second interpretation which leads to [unsupervised] datamining is by ignoring the last stage Fig (4) and instead making the equivalence

$$13) \qquad \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} \bar{x}_2\bar{x}_3\bar{x}_4 \\ x_1\bar{x}_2\bar{x}_3 \\ \bar{x}_1x_3x_4 \end{bmatrix}$$

Here the y_k are obviously the nonlinear clusters of data that a good datamining or knowledge discovery package should discover. [These clusters, or actually combinations of clusters may also be user-chosen and can be done at various levels of approximation (graining) to yield 'mixed' e.g. combination of supervised and unsupervised, datamining]. And the results can be further improved by training a special kind of an artificial neural network specifically customized for this problem [see below]. Furthermore, the results can be transformed back into the arithmetic domain (where *interval* and *ratio* scaled values may be used [14],[25],[23],) and still be interpreted using [a special] fuzzy logic so as to create association rules. The axioms of fuzzy logic can be found in many books ([8], [11], or [9]). Also in [8] is the special fuzzy logic that will be used for training of real-valued (interval-scaled or ratio-scaled) neural networks of the type here and in [4]. The crucial observation that relates real-valued variables to fuzzy-logic and association rules is in multiplicative logic (*vide supra*). It should be noted that some of the weights are negative. Clearly, using the fuzzy-logic above, the negative weights are to be interpreted as complements [or using probabilistic interpretations, they denote negative correlations]. Furthermore, the groups, $x_2^{-w_{12}}x_3^{-w_{13}}x_4^{-w_{14}}$, $x_1^{w_{21}}x_2^{-w_{22}}x_3^{-w_{23}}$, and $x_1^{-w_{31}}x_3^{w_{33}}x_4^{w_{34}}$ serve functions similar to *dimensionless groups of physics* [18], [27] and the exact relationships amongst the input variables should be sought in terms of these groups. Hence, the method achieves a nonlinear dimension reduction, better than PCA which is linear. At the same time, we have achieved the solution to one of the problems associated with neural networks; that is, we now know how many output (or hidden layer) neurons a neural network should have for some specific problem at hand as exemplified by the data we possess. We can now modify the digital circuit of Fig(4) to create a neural network which can be interpreted using the specific fuzzy logic shown above. This multiplicative neural network is customized for the problem and also does nonlinear regression. In addition, (i) we know how many output neurons we should have (ii) can perform nonlinear separation and (iii) does not need a second stage (for classification) unless we want to cluster the clusters.. In general the outputs (using the suppressed summation notation of Einstein) for this network are of the type

$$14) \quad \ln(y_i) = \sum_k w_{ik} \ln(x_k) \quad \text{or} \quad y_i = \prod_{k=1}^{n} x_k^{w_{ik}}$$

This network is obviously a [nonlinear] polynomial network, and thus does not have to "approximate" polynomial functions as the standard neural networks. The clustering is naturally explicable in terms of logic so that association rules follow easily. Furthermore, it quite easy interpret using fuzzy logic.

6. THE COMPLETE ALGORITHM

Making use of the various ingredients sketched out above we can produce a coarse-grained informal high-level method for clustering and rule extraction:

I) *Normalize* input vectors to $\{0, 1\}^n$. This is the *first approximation*. In high dimensions almost all data is in the corners [9]. This is also the hash address of each input vector. Thus we also have created a datawarehousing structure in which records can be fetched in O(1), the Holy Grail of file processing and database. The method will be illustrated, without loss of generality, via examples. We start by normalizing every component of each input vector y_j to the interval [0,1], that is f: $\Re^n \to [0, 1]^n$. The function $f(x) = [x - x_{min}]/[x_{max} - x_{min}]$ easily accomplishes this. (The interval $[-1, 1]$ may also be used, especially for time series.)

In the second step of the first phase we reduce every component of the vector via g: $[0, 1] \to \{0, 1\}$. This can be done quite easily via the *Heaviside Unit Step Function U(x)*. For each component of every input vector, let $x_j = U(x_j - \beta)$ where the bias $\beta = 0.5$. Save the occurrence counts of the binary input vectors in the *KH-map data structure* [11,12,15]. For very large dimensions hashing will be much more effective and efficient than the array structure. For smaller dimensions the array vs hash address is immaterial, since it is very easy to create a bucket-splitting algorithm to handle all sizes; however, it is not within the scope of this manuscript.

II) *Cluster Formation:* Select a threshold τ and create a new KH-map using $x_k = U(x_k - \tau)$. Apply the Quine-McCluskey algorithm and minimize the Boolean function. The resulting Boolean function is a high-level association rule. If the algorithm is running in the unsupervised mode, then each minterm is a [nonlinear] cluster. If the algorithm is running in the supervised mode, then the complete Boolean function is the nonlinearly separable cluster (such as the XOR). The association rule(s) at the same time determine the architecture of novel neural network architecture. They determine the number of output nodes, and the connections of the input nodes to the output nodes. The minterms (association rules) are the nonlinearly coupled groups of variables analogous to dimensionless groups of physics and thus perform nonlinear dimension reduction of the problem/data. The best results will be obtained if relationships are sought between these groups of variables.

III) Decrement the threshold and repeat I-III as many times as desired or needed to get as clear a picture of the data as possible.

IV) *Neural Network Classifier*: The novel neural network [4] is (i) a fuzzy decoder or (ii) a multiplicative neural network classifier/categorizer that performs nonlinear separation of inputs while reducing the dimensionality of the problem. Since the resulting neural network is customized for the data, then the number of output nodes is the *dimension of the problem/data.* The input variables can/should be renormalized to achieve good results using the neural network presented here and the special fuzzy logic first presented in [8]. Train the neural network to fine-tune the results using the renormalized variables.

V) *Rapid Nonlinear Regression*: Select a fuzzy logic truth valuation scheme and/or norm/conorm. Instant nonlinear regression without overfitting!

VI) Use the exponents of the variables in the nonlinear groups of variables (fuzzy minterms?) as the nonlinear mapping for an SVM feature space.

VII) *Interpretation*: The results can be interpreted in many ways using fuzzy logic, Boolean algebra, association rules, neural networks, or nonlinear mapping to the feature space of SVMs.

7. NORMALIZATION AND OUTLIERS

There are very good reasons why regression (used often by statisticians) should use equations of form

$$15) \quad y = \prod_i^n \left(\sum_j^m x_j \right)^{\alpha_i}$$

It can be explained easily using fuzzy logic [6]. Furthermore, for fine-tuning the neural network we should renormalize the input variables. For each component of the input vector, x, the transformation,

16) $$G: x \to (1 - F(x_{min}))\left[\frac{F(x) - F(x_{min})}{F(x_m) - F(x_{min})}\right] + F(x_{min})$$

for $x_{min} < x \le x_m$ will map $[x_{min}, x_m] \to [F(x_{min}), 1]$, where F(x) is any cdf (cumulative density function), x_{min} is the minimum data value in the input, x_{max} is the maximum value achieved by the inputs and x_m is something like the mean or the median of the data. Typically the corresponding pdf (probability density function), f(x) will be defined mostly in $[0, \infty]$. For $x_m \le x \le x_{max}$ the mapping will be $[x_m, x_{max}] \to [1, x_{max}]$ via the transformation

17) $$G: x \to (x_{max} - 1)\left[\frac{F(x) - F(x_m)}{F(x_{max}) - F(x_m)}\right] + 1$$

Hence the method, at least this part of it, is implicitly probabilistic. For a symmetric pdf (probability density function), approximately half the inputs will be greater than x_m, and the other half will be greater. Therefore the inputs can be thought of as fuzzy variables [small, large]. Because of this transformation, the small variables will have complements in the large interval and vice versa. Hence the transformation G, along with the complement c(x)=1/x implements a[n extended] fuzzy logic over the positive real-valued inputs. This also neatly takes care of the out-of-bounds (out-of-data) problem in the test data; we do not have to renormalize again, however the interpretation cannot be done strictly in terms of logic.

The interpretation of the "extended" fuzzy logic (in the sense that it is about interpretation of mathematical models e.g. those models employing arithmetic) we need to generalize De Morgan's laws since we already have a complement. It is easily seen that the implication $A \Rightarrow B$ is really $t(A) \le t(B)$. Using the Heaviside step function we can write the implication as $H(t(B) - t(A))$ where we define $H(0) = 1$, and $t(A)$ is the truth value assigned to the variable A. This definition will work both for fuzzy logic and crisp logic. Similarly we can extend De Morgan's laws by changing the equalities to inequalities. However, the results are dependent on the values of the variables e.g. the inequalities

18a) $$\overline{AB} \le \bar{A} + \bar{B}$$

18b) $$\overline{A + B} \le \bar{A}\bar{B}$$

are true only for $A + B \le 1$ (using the complement c(x)=1/x). We can write these concisely by using the Heaviside step function as used above.

19a) $$H(S[1 - (A + B)][\overline{AB} - (\bar{A} + \bar{B})]) = 1$$

19b) $$H(S[1 - (A + B)][\overline{A + B} - (\bar{A}\bar{B})]) = 1$$

where S(.) is the signum function e.g. sign of the argument. Since the function computes to $\{0,1\}$ the equations can be written without the equal sign and may be interpreted as logical statements or assertions. The use of the signum function reverses the direction of the inequality. Using these substitutions for De Morgan's laws one can treat ordinary multiplication as a conjunction, and addition as disjunction so that multiplicative neural networks which deal with unnormalized quantities (e.g. quantities in $[0, \infty]$) can be interpreted in terms of something simpler, e.g. fuzzy logical ideas. For the inverse complement these are given as

20a) $$H\left(S[1 - (A + B)]\left[\frac{1}{AB} - \left(\frac{1}{A} + \frac{1}{B}\right)\right]\right) = 1$$

20b) $$H\left(S(1 - [A + B])\left[\frac{1}{A + B} - \frac{1}{AB}\right]\right) = 1$$

The dimensionless groups (section 5) that arise from the clustering are extensions of simple products and can be interpreted as fuzzy intersections. It is not difficult to extend the "extended" De Morgan's laws to the interval $[-\infty, +\infty]$.

8. CONCLUSION

The method produces nonlinear [nonspheroidal] clusters, performs nonlinear dimension reduction akin to dimensional analysis, can be used in supervised or unsupervised modes, can be used to find the nonlinear transformations for Support Vector Machines, and much more.

9. REFERENCES

[1] Christianini & Taylor *Support Vector Machines*, CUP, Cambridge, UK.,2000.

[2] Han, J and M. Kamber *Data Mining,* Morgan Kaufmann, New York, 2001.

[3] Hecht-Nielsen, R *Neurocomputing*, Addison-Wesley, Reading, MA, 1990.

[4] Hubey H.M. *Curse of Dimensionality,* submitted to Journal of Data Mining and Knowledge Discovery, 2000.

[5] Hubey, H.M. *Fuzzy Logic and Calculus of Beauty, Moderation and Triage*, The Proceedings of the 2000 International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences METMBS2000, June 26-29, Las Vegas, 2000.

[6] Hubey, H.M. *Fuzzy Operators*, Proceedings of the 4th World Multiconference on Systemics, Cybernetics, and Informatics (SCI2000), July 23-26, 2000, Orlando, FL, 2000.

[7] Hubey, H.M. *Mathematical and Computational Linguistics*, Mir Domu Tvoemu, Moscow, Russia, 1994.

[8] Hubey, H.M. *The Diagonal Infinity,* World Scientific, Singapore, 1999.

[9] Hubey, H.M. *Vector Phase Space for Speech Analysis via Dimensional Analysis*, Journal of the International Quantitative Linguistics Association, Vol 6, No 2, August 1999, 117-148, 1999.

[10] Hubey, H.M. *Mathematical Foundations of Linguistics*, Lincom Europa, Muenchen, Germany, 1999.

[11] Hubey, H.M., I. Sigura, K. Kaneko, and P. Zhang, *Dimension Reduction in Datamining*, Proceedings of the International Conference on Computing and Information Technologies, Upper Montclair, NJ, pp. 275-282, 12 Oct 2001.

[12] Hubey, H.M. *O(1) Detection of Chace Events, Real-time Processing of High-Dimensional Streaming Data, and Datamining*, AAAI Symposium Series: Chance Discovery, North Falmouth, MA, USA, November 15-17, 2002.

[14] Klir, G. and B. Yuan *Fuzzy Sets and Fuzzy Logic*, Prentice-Hall, Englewood Cliffs, NJ, 1995.

[14] Leighton, T *Introduction to Parallel Algorithms and Architectures*, Morgan Kaufmann, San Mateo, California, 1992.

[15] Mano, M *Digital Design*, Prentice Hall, Englewood Cliffs, New Jersey, 1991.

[16] Maranell, G. (ed) *Scaling: A Sourcebook for Behavioral Scientists,* Aldine Publishing Company, Chicago, 1974.

[17] Mitchell, T. *Machine Learning*, McGraw-Hill, NY, 1997.

[18] Olson, R *Essentials of Engineering Fluid Mechanics*, Intext Educational Publishers, NY, 1973.

[19] Reed, Russell and Robert Marks, *Neural Smithing*, MIT Press, Cambridge, MA, 1999.

[20] Schmidt-Nielsen, K. *Scaling: Why is Animal Size So Important?*, University of Cambridge Press, Cambridge, 1984.

[21] Stallings, Wm.Computer Organization and Architecture. Macmillan, New York, 1993.

[22] Shiva, S. *Comp Design & Archictecture*, Harper Collins, New York, 1991.

[23] Stevens, S. (ed) *Handbook of Experimental Psychology,* John Wiley & Sons, New York, 1966.

[24] Tomek, I. *Comp Arch and Org*, Computer Science Press, New York, 1990.

[25] Torgerson, W. *Theory and Methods of Scaling*, Wiley, New York, 1958.

[26] Tveter, D. *The Pattern Recognition Basis of Artificial Intelligence*, IEEE Comp Society, Los Alamitos, CA, 1998.

[27] White, F. *Fluid Mechanics*, McGraw-Hill, New York, 1979.

[28] US Patent Pending 60/294314