

Clustering Human Feelings Using Neural Networks and Metric Topology for Small Sample Size

Yun Bai
Kyle Miller
Weian Ou
Nandita Singh

April 28, 2004

Abstract: Manufacturers of consumer goods often conduct surveys to determine customer satisfaction as well as guide the design of future products. In order to make use of the information it is necessary to determine the most important underlying common factors in the survey results. However, inconsistency in data obtained from humans is often problematic for conventional linear statistics. Conducting surveys over large numbers of people, a method commonly used to counter this weakness, is not always feasible or cost effective. Therefore we have developed two methods to overcome the drawbacks of linear statistical methods: an artificial neural network modified to analyze small survey data over a small sample (10-30) of human subjects and a method of analysis based on the theory of metric topology.

Work done for Johnson Controls, Inc. under the direction of James A Kleiss and Colleen Serafin, in partial fulfillment of the requirements of Michigan State University MTH 844, advised by Professor Ralph Svetic.

Table of Contents

Introduction	2
Factor Analysis	3
Neural Networks	3
Neural Network Architectures	4
Learning Process	4
Adaptive Resonance Theory	5
Implementation	7
Metric Topology Model	10
Discussion	11
Recommendations	13
Neural Network Model	13
Metric Topology Model	13
Future Work	13
Neural Network Model	13
Metric Topology Model	13
References	14
Appendix A	14
Appendix B	15

Introduction

In today's competitive market, consumers can find several alternatives to any given product. Therefore, it is necessary, from the perspective of product designers, to maximize the desirability of their products in order to vie with the competition.

In order to achieve this goal, Johnson Controls, Inc. (JCI) conducts surveys of its customers. The base of JCI's surveys requires customers to rank a particular product on a discrete scale between many adjective pairs. This method is called a semantic differential (SD) scale. In the case of this project JCI has provided a sample survey using a 7-grade scale. The data is then analyzed to identify subsets of the adjective pairs that define common underlying factors. This information allows JCI to objectively measure how well, and why, a given product fits the needs of customers. Adjective pairs used in the survey are shown in Table 1.

Table 1. Adjective pairs used in sample survey.

Index	Adjective Ratings (1-7)	Index	Adjective Ratings (1-7)
0	Inaccurate – Precise	13	Out of Reach – Reachable
1	Unpredictable – Predictable	14	Tight – Roomy
2	Ambiguous – Obvious	15	Worthless – Valuable
3	Illogical – Logical	16	Obscure – Distinguishable
4	Inconsistent – Consistent	17	Disorganized – Organized
5	Hidden – Exposed	18	Slow – Quick
6	Unintuitive – Intuitive	19	Not Controllable – Controllable
7	Challenging – Easy	20	Distressed – Soothed
8	Underdone – Overdone	21	Tense/Anxious – Relaxed
9	Unresponsive – Responsive	22	Agitated – Peaceful
10	Incomprehensible – Intelligible	23	Uncomfortable – Comfortable
11	Not Useful – Useful	24	Displeased – Pleased
12	Blurry – Clear		

Currently JCI employs a linear method called factor analysis to examine their data. However, nonlinearities in human survey data can decrease the accuracy of the results obtained from this method.

The natural characteristics of artificial neural networks, on the other hand, are able to overcome this shortcoming. For that reason, we developed an artificial neural network model incorporating self organizing ART1.5 architecture, with the capability to interpret complex, nonlinear underlying relationships in survey data.

In addition to the neural network model, we developed another model based on the theory of metric topology. This model seeks to overcome the shortcomings of linear statistics by using the definition of the distance metric in a vector space to measure similarity of survey results.

Factor Analysis

Given a set of data vectors, the goal of factor analysis is to reduce the dimension of a correlation matrix generated by the data set, in order to identify the underlying factors. The original correlation matrix is replaced by a low dimension factor matrix.

The steps of the factor analysis method are:

- 1) Subtract from each data vector its mean value across each dimension.
- 2) Calculate the covariance matrix of original variables.
- 3) Find eigenvalues and eigenvectors of the covariance matrix. Order the eigenvectors by eigenvalues, from highest to lowest.

This gives the components in order of significance; the components of lesser significance can then be ignored.

Factor analysis reveals the correlation between the data vectors and provides a reasonable basis for identifying the significant factors. However, nonlinearity of the data set decreases the accuracy of results.

The focus of this project is to create a method that is not subjected to the same weaknesses as factor analysis. To this end, JCI has provided a sample survey along with its factor analysis results for the purpose of comparison.

Neural Networks

The following is a general introduction to the subject of artificial neural networks. Simon Haykin, a noted expert in the field of artificial neural networks, gives this general definition [1]: “A neural network is a machine that is designed to model the way in which the brain performs a particular task or function of interest; the network is usually implemented using analog electronic components or simulated in software on a digital computer.”

He coined two important properties of neural networks:

1. Massively parallel distributed structure.
“A neural network is made up of simple processing units called neurons, which have a natural processing unit. These neurons have a natural propensity for storing experimental knowledge and making it available for use. It resembles brain in the following two aspects:
 - (a) The network acquires knowledge from its environment through a learning process.
 - (b) Interneuron connection strengths, known as synaptic weights, are used to store the acquired knowledge.”
2. Ability to learn and therefore generalize.
Haykin adds, “Here generalization means the capability of the neural networks to produce reasonable output given input that has not been encountered during the training process.”

Neural Network Architectures

Neural network architecture¹ denotes the general interaction between neurons, or the manner in which those neurons are interconnected. Here are three basic types of network architectures:

(a) Single-layered feedforward networks

Under this architecture, the neurons are organized in two layers: One is an input layer, composed of source neurons, which receive external input and relay it to other neurons. The other is an output layer, composed of computation neurons that carry out computation and thereby produce network output, as shown in Figure 1.

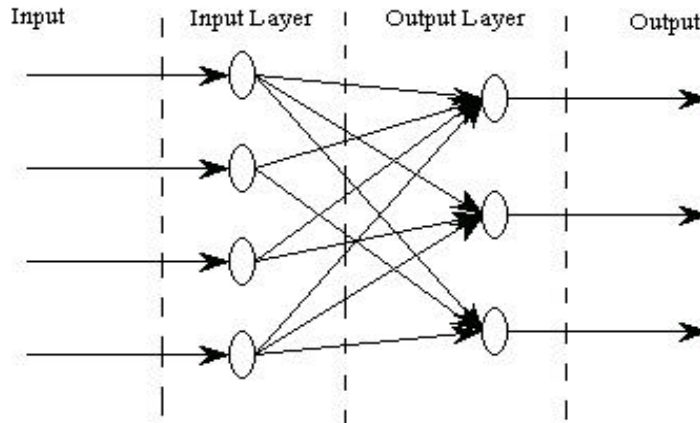


Figure 1. Single-layered feedforward network

(b) Multi-layered feedforward networks

This architecture distinguishes itself from a single-layered feedforward network in that it has one or more layers between the input layer and the output layer, sometimes called hidden layers, which are composed of computational neurons. Typically, the input of each layer consists of the output of the preceding layer. This changes the level of complexity in the networks behavior significantly. Haykin observes that, “by adding one or more hidden layers, the network is enabled to extract higher-order statistics” (Simon Haykin, 1994).

(c) Recurrent networks

The recurrent network architecture is defined by the characteristic that the network contains some neurons that are interconnected with neurons in the same layer. In other words, the output of some neurons is received as input by other neurons in the same layer, thereby affecting their behavior.

Learning Process

The ability of artificial neural networks to solve problems rests in their ability to learn. The learning process refers to changing of weight values assigned to the connections between neurons. The specific learning algorithm employed by a network is determined

¹ This theory of neural network and its architecture is adapted from Simon Haykin, “*Neural Networks, A Comprehensive Foundation.*”

by the architecture. Figure 2 shows simple model of a learning process from Simon Haykin [1].

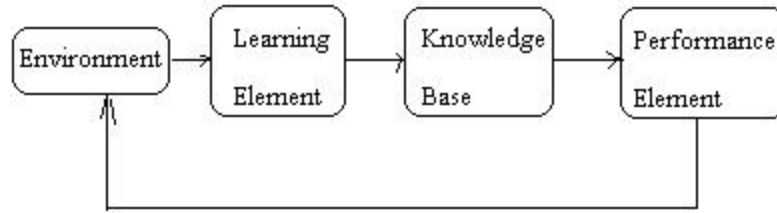


Figure 2. Learning Process.

The learning element uses information from the environment to improve the knowledge base, which the network uses in the performance of its task. The feedback mechanism enables the machine to evaluate its performance generated by the knowledge base and revise itself if necessary. This cycle is repeated until the network has been sufficiently trained. Three basic schemes in terms of the training method are:

(a) Unsupervised learning

In this method, the network learns without the supervision of a “teacher.” In other words, there are no input-output examples provided to the network to measure performance.

(b) Supervised learning

Supervised learning relies on a “teacher” to measure the network’s performance; this is usually provided in the form of a set of input-output examples. The input signals are fed into the network and its output is compared with the example output. Based on a measurement of error, the network is adjusted until the output produced by the network is satisfactory.

(c) Back propagation

This method is characterized by propagating error information backwards through the network, which is used to update connection weights in multiple stages. Each stage of the propagation seeks to minimize error. This method is quite successful with a multi-layered feed forward network.

Adaptive Resonance Theory

Adaptive Resonance Theory (ART) 1.5² is a simplified version of the ART2 architecture which features unsupervised learning and self-organizing pattern classification.

Architecture and Mechanism

As shown in Figure 3, ART1.5 architecture as suggested by Shigekazu Ishihara et al [2] consists of two layers and a reset mechanism. The neurons in the bottom layer (F1 layer) receive the input data and the neurons in the top layer (F2 layer) represent the clustering

² This theory of ART1.5 and learning algorithm is adapted from Shigekazu Ishihara, Keiko Ishihara, Mitsuo Nagamachi, Yukihiro Matsubara, *An automatic builder for a Kansei Engineering expert system using self-organizing neural networks.*

units. The reset mechanism facilitates the competitive learning process (see next section). Each input vector will cause a single F2 layer neuron to activate. This neuron represents the cluster the input vector belongs to, and thereby classifies input patterns. For the purpose of pattern classification, the network uses two sets of connection weights: a bottom-up F1 \rightarrow F2 adaptive filter and a top-down F2 \rightarrow F1 filter. These weights serve as a type of memory. The term top-down is misleading in this case, since the output from the F2 layer is not relayed back to the F1 layer, as it is in other resonance theory networks. For the sake of convention, however, the term is maintained.

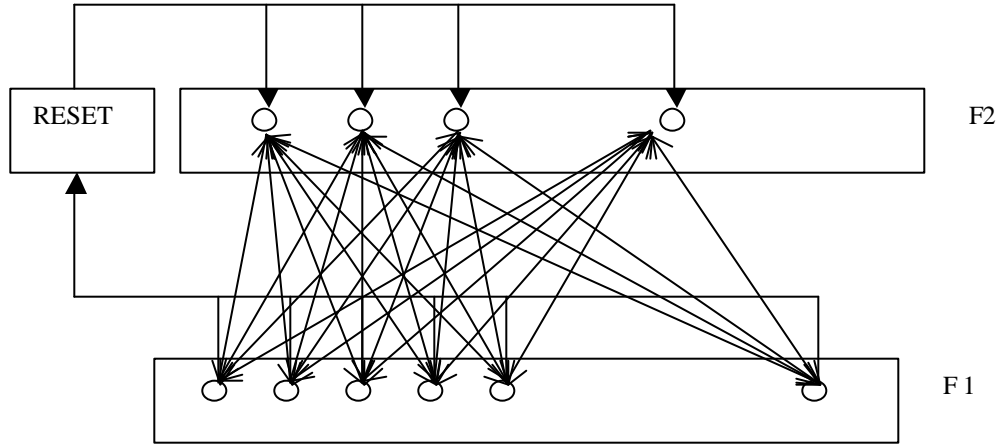


Figure 3. Structure of ART 1.5.

Competitive Learning Method

In terms of neurobiology each neuron in the F2 layer has an inhibitory effect on the other F2 layer neurons. In effect, only one F2 layer neuron can be active at one time. The implementation of this competitive behavior is as follows. For each input signal a search process for the neuron with the highest activation level begins, utilizing the bottom-up connection weights. Once found, this neuron's top-down connection weights are compared with the input signal. If the neuron passes this test it is declared the winner, and the learning process begins. Otherwise it is inhibited, so that it cannot participate in the next search, and a new search begins. The neuron is only inhibited for the current input pattern.

The method for implementing the search process is as follows. Let x_i be the activity of i^{th} F1 unit, and T_j is activation level of j^{th} F2 unit. Let b_{ij} be the bottom-up weight from the i^{th} F1 node to the j^{th} F2 node. We have

$$T_j = \sum_i x_i b_{ij} \quad (1)$$

and
$$m = j, \text{ if } T_j = \max T_j. \quad (2)$$

In the case that there are multiple j that satisfy (2), simply choose the first one. Note that inhibited neurons do not participate. After the neuron with the maximum activation level has been selected as a candidate, it is compared to the network's vigilance

parameter. This parameter determines the suitable similarity between the input signal and the stored top-down weights using the following inequality

$$\frac{\vec{x} \cdot \vec{t}_m}{\|\vec{x}\| \|\vec{t}_m\|} > r, \quad (3)$$

where \vec{x} is the input vector in the F1 layer, and \vec{t}_m is the vector of top-down weights from the m^{th} F2 neuron. The vigilance parameter r is a user-tunable variable, which determines the rigorousness of classification. If the inequality is satisfied, the neuron is declared the “winner”. After a winner is selected, the top-down and bottom-up connection weights corresponding to the winning neuron are changed so that the neuron learns the new input pattern using the following formula

$$\begin{aligned} \frac{d}{dt} b_{ij} &= r_b (x_i - b_{ij}), \\ \frac{d}{dt} t_{ji} &= r_t (x_i - t_{ji}), \end{aligned} \quad (4)$$

where r_b and r_t are the learning rates of the bottom-up and top-down weights, respectively, ($0 < r_b, r_t \leq 1$), b_{ij} is the bottom-up weight, t_{ji} is the top-down weight, and x_i is the input from the i^{th} F1 neuron. This, in effect, causes the top-down weights to serve as a prototype of each cluster.

If the inequality is not satisfied, we say the input vector and the top-down weights are not similar enough, then the F2 neuron is “inhibited” and a reset signal is sent to the F2 neurons. At this point, a new search begins. Inhibited neurons are not included in subsequent searches. This process is repeated until a suitable winning neuron is found. If no such neuron is found, a new neuron is created to learn the input signal.

Implementation

In this paper we explored two models of clustering input data, thereby determining relevant underlying factors; one is based on artificial neural networks while the other is based on the theory of metric topology.

For neural network model we modified the ART 1.5 learning algorithm, which greatly improved the accuracy and stability of clustering results.

Training the network

As a point of operation, input vectors are normalized to maintain a reasonable bound on calculated values. Also, in order to avoid the possibility that input patterns learned early during the operation of the network would become too dissimilar from the cluster as later patterns were learned by a given neuron, the learning rate is decreased over time. Ishihara, in his paper, suggests that the learning rate be equal to the inverse of the number of times a given neuron has been declared the winner. In order to gain further stability between patterns learned early and later during operation, we modified this learning rule so that the top-down learning rate is the square of the bottom-up learning rate. This causes the top-down weights to act simultaneously as both long-term memory and a

prototype vector for each cluster. Since bottom-up weights will be influenced more by recently learned patterns, they obviously act as short-term memory. Refer to Appendix A for an outline of the algorithm.

Vigilance parameter

The vigilance parameter, $0 < \rho < 1$, determines the rigorousness of classification in the network. This, in effect, controls the number of clusters obtained after running the network. A relatively low value of ρ ensures fewer clusters while higher values yield more clusters.

Multiple iterations with random input order

Because of the nature of the learning method for artificial neural networks, order of input is an unavoidable issue. In this implementation, different orders of input data will yield slightly different results. In order to overcome this instability in clustering, we designed a method of running the network many times, each time with a random input order. The results of each run are then collected together. Using the prototype vectors of each cluster, namely the top-down weights of the F2 neuron, the clusters of each run are added to the most similar collection. The results show that after a suitable number of iterations the clusters become both stable and reasonable.

Ranking the members of clusters

Within each cluster, ranking of members is very important to interpretation of results. The most important members can be used as representatives of the overall cluster and therefore determine the important underlying factors of the data set. In this approach, there are two methods for ranking: the frequency an input vector appears in a given cluster and the inner product of the input vector and the prototype vector. Ideally, the frequency of important vectors should be close to the total number of iterations. Similarly, the inner product, which measures the angle between two vectors, should be highest among those vectors most influential inside the cluster. Our results show that for this sample survey, those ideals are indeed realized. Important vectors among the clusters have both high frequency and high inner product values.

Neural Network clustering effectiveness

Clusters obtained after several iterations are shown in Table 2, with members listed in order of rank. Elements of the second column are the indices of the adjective pairs (Table 1). It is clear that after 10 iterations the clusters are stable, but member ranking varies. However, after 100 iterations, member ranking also becomes more stable.

Table 2. Clusters obtained on varying the number of iterations.

Number of iterations	Clusters based on $\rho = 0.96$
10	{0,1,9,23}, {2,6,7,11,12,17,21}, {3,4,10,16,18,22}, {20,5,8,19,13,15,24}, {14}
20	{0,9,23,1}, {2,6,12,7,11,17,21}, {3,4,10,16,18,22}, {5,13,19,20,15,24,8}, {14}

30	{0,1,9,23}, {2,6,7,11,12,17,21}, {3,4,10,18,16,22}, {5,19,20,13,15,8,24}, {14}
40	{0,1,9,23}, {6,11,12,2,7,17,21}, {3,4,10,16,18,22}, {13,5,19,15,20,24,8}, {14}
50	{0,1,23,9}, {7,2,11,6,12,17,21}, {18,22,3,10,4,16}, {13,15,5,19,20,24,8}, {14}
60	{0,23,9,1}, {6,11,2,7,12,17,21}, {3,10,4,16,18,22}, {5,20,19,13,15,8,24}, {14}
70	{9,1,23,0}, {6,2,7,11,12,17,21}, {4,10,16,3,18,22}, {13,15,5,19,20,24,8}, {14}
80	{0,23,9,1}, {6,12,7,11,2,17,21}, {18,22,10,4,3,16}, {13,5,19,15,20,24,8}, {14}
90	{1,9,23,0}, {6,11,2,7,12,17,21}, {4,18,3,10,16,22}, {13,5,15,19,20,24,8}, {14}
100	{23,9,0,1}, {11,12,2,6,7,17,21}, {10,18,22,3,4,16}, {13,19,5,15,20,24,8}, {14}
200	{23,0,9,1}, {6,7,2,11,12,17,21}, {3,4,10,16,18,22}, {19,5,13,20,15,24,8}, {14}
300	{23,0,9,1}, {6,7,2,11,12,17,21}, {3,4,10,16,18,22}, {19,5,13,20,15,24,8}, {14}
400	{23,0,9,1}, {6,11,12,7,2,17,21}, {3,4,10,18,16,22}, {13,5,19,15,20,24,8}, {14}
500	{23,9,0,1}, {6,11,12,7,2,17,21}, {10,3,4,16,18,22}, {13,5,19,15,20,24,8}, {14}

Clusters obtained by varying the vigilance parameter between 0.95 and 0.97 are shown in Table 3. We can see that the number of clusters generated increases with the value of ρ .

Table 3. Clusters obtained on varying the vigilance parameter.

Vigilance parameter, ρ	Clusters ordered in ranks
95.0%	{8,23,9,20,0,19,5,1,13,15,24}, {2,7,17,11,12,21,6,10,3,4,16,18,22}, {14}
95.2%	{23,9,0,1,8,5,19,20,13,15,24}, {3,16,4,10,22,18,2,7,11,12,21,17,6}, {14}
95.4%	{5,19,13,20,8,15,23,0,9,24,1}, {6,2,12,17}, {4,3,10,16,22,18,7,11,21}, {14}
95.6%	{5,19,20,13,8,15,24,9,23,0,1}, {7,11,12,17,2,6,21}, {10,8,16,22,4,3}, {14}
95.8%	{0,23,9,1}, {7,11,12,2,6,17,21}, {18,10,16,22,3,4}, {19,13,5,20,15,24,8}, {14}
96.0%	{0,1,9,23}, {7,11,12,6,2,17,21}, {10,18,22,4,3,16}, {13,5,19,15,20,24,8}, {14}
96.2%	{23,0,9,1}, {17,6,2,7,11,12,21}, {3,16,4,10,18,22},

	{13,5,19,15,20,24}, {8}, {14}
96.4%	{0,1,9,23}, {6,2,7,12,11,17,21}, {3,4,10,16,18,22}, {15,13,19,5,20,24}, {8}, {14}
96.6%	{23,0,1,9}, {7,11,12,17,2,6,21}, {3,4,10,16,18,22}, {20,24,15,19,5,13}, {8}, {14}
96.8%	{0,1,9,23}, {2,11,12,7,6,17}, {18,22,10,3,4,16}, {20,24,15,19,5,13}, {8}, {14}, {21}
97.0%	{0,1,9,23}, {11,2,17,12,7,6}, {3,4,10,16,18,22}, {20,24,15,5,19}, {8}{13}{14}{21}

Metric Topology Model

Order of input is a factor in the artificial neural network approach. Because of this, we sought other methods that did not share this issue. The metric topology method, which is not influenced by input order, arose from a different method for determining similarity. In the artificial neural network model, similarity between input patterns is primarily measured using an inner product. Other methods of determining similarity of vectors is a distance metric.

Suppose a, b are two vectors in R^n space (vectors of dimension n , whose components are real numbers). Then the distance between the two vectors is

$$Dist(a, b) = \sum_{i=1}^n (a_i - b_i)^2, \quad (5)$$

Unlike the neural network above, input vectors are not normalized in this algorithm. The method is as follows. Surround each vector with an n dimensional ball of fixed radius. Find the ball with maximum members; in case of more than one, simply choose the first. Denote all vectors in the ball as belonging to the same cluster, and rank them by distance from the center of the ball. Then remove the vectors and repeat the process until all vectors are clustered. In this way, the radius used can be varied until a reasonable set of clusters is found. (This method was suggested by Dr. T. Y. Li, Professor of Mathematics, Michigan State University).

Unlike the artificial neural network this method is completely independent of input order. However, in some cases the results generated by this approach are not entirely satisfactory. Different radii produce considerably different member populations. This is because the algorithm, as is, is somewhat greedy in that the first cluster found would most likely remove vectors that may more reasonably belong to other clusters. These vectors will tend to rank low in their respective clusters.

Table 4. Clusters obtained on varying the radii.

Distance	Number of Clusters	Clusters
0	25	{0}, {1}, {2}, {3}, {4}, {5}, {6}, {7}, {8}, {9}, {10}, {11}, {12}, {13}, {14}, {15}, {16}, {17}, {18}, {19}, {20}, {21}, {22}, {23}, {24}
26.6833	17	{12,2,7,17}, {3,4,10}, {0,9}, {11}, {18,22}, {20,24}, {1}, {5}, {6}, {8}, {13}, {14}, {15}, {16}, {19}, {21}, {23}

28.5835	10	{12,2,7,17,11,6},{4,10,3,18,16},{9,0,23,1},{19,5,15}, {22},{20,24},{21},{8},{13},{14}
31.1778	6	{12,2,7,17,11,6,21},{3,4,10,22,16,18},{15,19,24,13,20,5}, {1,9,0,23},{8},{14}
32.0399	7	{7,12,17,2,11,6,21},{22,18,10,24,3,4},{15,19,13,20,5},{2 3,9,1,0},{16},{8},{14}
32.9326	6	{12,2,7,17,11,6,21,10,4},{22,18,24,3,16},{15,19,13,20,5}, {23,9,1,0},{8},{14}
33.8591	6	{4,10,3,18,16,22,12,24,15,2},{20,19,5,23,8,9},{21,17,7,1 1,6},{0,1},{13},{14}
34.4929	6	{2,12,6,7,11,17,21,3,4,10},{24,20,22,15,19,5,16,18},{9,0, 23,1},{13},{8},{14}
34.7378	6	{2,12,6,7,11,17,21,3,4,10},{24,20,22,15,19,5,16,18},{9,0, 23,1},{13},{8},{14}
36.4054	5	{4,10,3,18,16,22,12,24,15,2,17,11,7},{20,19,5,23,8,9,13, 0},{21,6},{1},{14}
36.6865	4	{10,4,3,22,16,18,12,24,2,17,15,7,21,11},{20,19,5,23,8,9, 13,0,1},{6},{14}
38.0720	4	{17,11,12,7,21,2,6,10,4,16,1,3,9,0,22,18},{15,19,24,13,2 0,5},{23,8},{14}
40.3811	2	{4,10,3,18,16,22,12,24,15,2,17,11,7,6,21,14,13,19,5},{0, 9,1,23,20,8}
40.8646	3	{21,17,12,7,11,22,2,6,18,23,10,9,16,0,4,1,24,3,20,14},{1 5,19,13,5},{8}
41	2	{24,20,22,15,19,4,5,16,18,10,13,3,8,0,21,23,9,14,1,11,12, 17},{6,2,7}

Discussion

The results obtained from the three different models were compared: factor analysis being used by JCI at present, the neural network model and the metric topology model. We observe that results from all three methods give somewhat reasonable results. For Table 5, a vigilance parameter of 96% with 100 iterations was used for the neural network model, and a radius of 34.7378 was used for the metric topology method.

Table 5. Comparison of Factor Analysis, Neural Network and Metric Topology model.

Factor Analysis	Neural Network	Metric Topology
<i>Cluster 1</i>	<i>Cluster 1</i>	<i>Cluster 1</i>
Agitated – Peaceful	Incomprehensible –	Incomprehensible –
Illogical – Logical	Intelligible	Intelligible
Inconsistent – Consistent	Slow – Quick	Inconsistent – Consistent
Incomprehensible –	Agitated – Peaceful	Illogical – Logical
Intelligible	Inconsistent – Consistent	Agitated – Peaceful
Slow – Quick	Illogical – Logical	Obscure – Distinguishable
Obscure – Distinguishable	Obscure – Distinguishable	Slow – Quick

Tight – Roomy Displeased – Pleased		Blurry – Clear Displeased – Pleased Ambiguous – Obvious Disorganized – Organized Worthless-Valuable
<i>Cluster 2</i>	<i>Cluster 2</i>	<i>Cluster 2</i>
Unpredictable – Predictable Unintuitive – Intuitive Unresponsive – Responsive Disorganized – Organized Blurry – Clear Not Useful – Useful Challenging – Easy Ambiguous – Obvious Inaccurate – Precise Uncomfortable – Comfortable Tense/Anxious – Relaxed	Out of Reach – Reachable Hidden – Exposed Not Controllable – Controllable Worthless – Valuable Distressed – Soothed Displeased – Pleased Underdone – Overdone	Distressed – Soothed Not Controllable – Controllable Hidden – Exposed Uncomfortable – Comfortable Underdone – Overdone Unresponsive – Responsive Out of Reach – Reachable
<i>Cluster 3</i>	<i>Cluster 3</i>	<i>Cluster 3</i>
Not Controllable – Controllable Hidden – Exposed Out of Reach – Reachable Worthless – Valuable	Inaccurate – Precise Unpredictable – Predictable Unresponsive – Responsive Uncomfortable – Comfortable	Tense/Anxious – Relaxed Challenging-Easy Not Useful- Useful Unintuitive-Intuitive
<i>Cluster 4</i>	<i>Cluster 4</i>	<i>Cluster 4</i>
Distressed – Soothed Underdone - Overdone	Challenging – Easy Not Useful – Useful Blurry – Clear Unintuitive – Intuitive Ambiguous – Obvious Disorganized – Organized Tense/Anxious – Relaxed	Inaccurate-Precise Unpredictable – Predictable
	<i>Cluster 5</i>	<i>Cluster 5</i>
	Tight-Roomy	Tight-Roomy

The neural network model produces very reasonable results. Though there is no obvious standard with which one can judge the performance of any one of these methods.

The metric topology results clearly indicate, that the first cluster is always the largest. As a consequence of this greedy method, the integrity of other clusters is sacrificed.

Cluster 4, of the neural network and metric topology models, is a very interesting case. This cluster has only one member: “*Tight-Roomy*.” Throughout the entire project this adjective pair was consistently isolated in its own cluster. However, this is to be expected since “*Tight-Roomy*” is clearly an expression of space, whereas none of the other adjective pairs carry this strong feeling of a spatial relation. This expression is intuitively unrelated to the other adjective pairs.

Recommendations

Neural Network Model

For the provided sample survey we recommend:

1. Choose vigilance parameter of 96%.
2. Run the network in 10 iterations for stable clusters.
3. Run the network in 100 iterations for stable ranking of the members of the clusters.
4. Try a different vigilance level for other samples until satisfied.

For future surveys, a vigilance parameter of above 90% and high number of iterations will naturally yield the best results. The vigilance level should be tuned until the most rational results are found.

Metric Topology Model

For this model, a wide number of clusters can be generated by varying the radius used and the best setting be selected depending on the users requirement. We also recommend further work and testing be done on this method before its value can be accurately assessed.

Future Work

Neural Network Model

For artificial neural network models, a different learning algorithm could be explored that would provide more stable clusters in fewer iterations. In addition to improving the ART1.5 method, more sophisticated architectures should be tested and explored for their ability to handle this type of problem.

Metric Topology Model

The metric topology method requires significant research before its value can be assessed. This method is only the beginning of a more sophisticated implementation. Using other user-tunable parameters, such as density, could yield much better results. In addition, utilizing principles of overlapping clustering and fuzzy-logic could possibly greatly strengthen this method. Further research is necessary.

References

- [1] Simon Haykin, “*Neural Networks, A Comprehensive Foundation,*” 2nd edition, Prentice Hall, Upper Saddle River, New Jersey, 1998.
- [2] Shigekazu Ishihara, et al., “*An automatic builder for a Kansei Engineering expert system using self-organizing neural networks,*” International Journal of Industrial Ergonomics, 15(1):13-24, 1995.
- [3] Charles R. MacCluer, “*Industrial Mathematics; modeling in industry, science, and government,*” Prentice Hall, Upper Saddle River, New Jersey, 2000.
- [4] Joey Rogers “*Object-Oriented Neural Networks in C++,*” Academic Press, INC., San Diego, CA, 1997.
- [5] Robert J Schalkoff , “*Artificial Neural Networks*” McGraw-Hill Education, Europe, 1997.
- [6] Teuvo Kohonen, “*Self-organization and associative memory,*” Springer-Verlag New York Inc., New York, 1989.

- [7] Daniel Klerfors, “*Artificial Neural Networks, What are they? How do they work? In what areas are they used?*”

www.hj.se/~de96klda/NeuralNetworks.htm

- [8] Anil K Jain, Richard C Dubes, “*Algorithm for clustering data,*” Prentice Hall, Englewood cliffs, New Jersey, 1988.

Appendix A

Initialize:

Reset = True

x_i = activation at F1 layer

s_i = input signal

b_{ij} = bottom-up-weight = 0

t_{ji} = top-down-weight = 0

a = number of times F2 neuron is committed = 0

set F2 Layer neurons to “uncommitted” status.

Begin

While input vector:

1. Activate F1 layer nodes with an input signal:

$$x_i = s_i$$

While committed neuron found in F2 layer:

2. Activate committed j^{th} unit of F2 layer that receives maximum input signal multiplied by bottom-up-weight.
3. Check for the angle between input signal x and top-down-weight t_{ji} :

If $\frac{\dot{x} \cdot t_j}{\|x\| \|t_j\|} > r$,

a = a+1

Goto Step 6

Else

Reset = False

Inhibit the jth node of F2 layer

Goto Step 2.

End While.

4. If all the committed nodes fail the test, choose an uncommitted node for new category.
5. Goto step 2.
6. Update weight:

$$r_t = 1/a^2$$

$$r_b = 1/a$$

$$\frac{d}{dt} t_{ji} = r_t (x_i - t_{ji})$$

$$\frac{d}{dt} b_{ij} = r_b (x_i - b_{ij})$$

$$t_{ji}(\text{new}) = t_{ji}(\text{old}) + \frac{d}{dt} t_{ji}$$

$$b_{ij}(\text{new}) = b_{ij}(\text{old}) + \frac{d}{dt} b_{ij}$$

Check for stopping condition

End While

End

Appendix B

Table 6. Tasks evaluated in the survey.

Index	Task	Index	Task
1	After you drive	8	Look
2	Audio	9	Outside the car
3	Check gauges	10	Pre-driving adjustments
4	Cruise	11	Steering column controls/special
5	Drink holders	12	Storage
6	Gears and pedals	13	Windows
7	HVAC		