Sanya Mitaim
Bart Kosko
Signal and Image Processing Institute
Department of Electrical Engineering—Systems
University of Southern California
Los Angeles, California

Neural Fuzzy Agents for Profile Learning and Adaptive Object Matching

Abstract

A neural fuzzy system can learn an agent profile of a user when it samples user question-answer data. A fuzzy system uses if-then rules to store and compress the agent's knowledge of the user's likes and dislikes. A neural system uses training data to form and tune the rules. The profile is a preference map or a bumpy utility surface defined over the space of search objects. Rules define fuzzy patches that cover the surface bumps as learning unfolds and as the fuzzy agent system gives a finer approximation of the profile. The agent system searches for preferred objects with the learned profile and with a new fuzzy measure of similarity. The appendix derives the supervised learning law that tunes this matching measure with fresh sample data. We test the fuzzy-agent profile system on object spaces of flowers and sunsets and test the fuzzy agent matching system on an object space of sunset images. Rule explosion and data acquisition impose fundamental limits on the system designs.

1 Smart Agents: Profile Learning and Object Matching

How can we teach an agent what we like and dislike? How can an agent search new databases on our behalf? These are core questions for both human agents and intelligent software agents. We explore these questions with the joint tools of fuzzy rule-based systems and neural learning. These tools exploit the filter and set-theoretic structure of agent search.

An intelligent agent can act as a smart database filter (Grosky, 1994; Maes, 1994). The agent can search a database or a space of objects on behalf of its user. The agent can find and retrieve objects that the user likes. Or the agent can find and then ignore or delete objects that the user does not like. Or it can perform some mix of both. The agent acts as a filter because it maps a set of objects to one or more of its subsets. The agent is ''smart'' (Brooks, 1995; Maes, 1995a; Steels, 1995) to the degree that it can quickly and accurately learn the user's tastes or object profile and to the degree that it can use that profile map to search for and to rank preferred objects. Figure 1 shows how an agent can learn and store user tastes as a bumpy preference surface defined over search objects.

Agent search depends on set structure in a still deeper way. The search system itself may have many parts to its design and may perform many functions in many digital venues (Colombetti & Dorigo, 1994; Yamauchi & Beer, 1994). But at some abstract level the agent partitions the object space into two fuzzy

Presence, *Vol. 7, No. 6, December 1998*, 617–637

Figure 1. *Profile learning. A neural fuzzy agent learns a user's utility surface as the user samples a database of classic paintings. The twelve bumps or extrema on the preference map show how much the user (or the agent that acts on the user's behalf) likes or dislikes the 12 paintings. Here the evolving utility surface forms in the "mind's eye" of a neural fuzzy agent based on nineteenth-century English philosopher John Stuart Mill.*

or multivalued sets with blurred borders. The agent partitions the space into the fuzzy set of objects that it assumes the users likes and into the complement fuzzy set of objects that it assumes the user does not like. All search objects belong to both of these fuzzy sets to some degree. Then the agent can rank some or all of the objects in the preferred set and can pick some of the extremal objects as its output set.

The agent needs a profile of its user so that it can group objects and rank them. The agent must somehow learn what patterns of objects the user likes or dislikes and to what degree the user likes or dislikes them (Maes, 1995b; Rasmus, 1995). This profile is some form of the user's implicit preference map. The user may state part of this map in ordinal terms: ''I like these red flowers more than I like those blue flowers. I like the large purple flowers about the same as I like the small red-white flowers.'' The objects may be fuzzy patterns or fuzzy clusters in some feature space (Krishnapuram & Keller, 1993; Pal & Bezdek, 1995; Pal, Bezdek, & Hathaway, 1996).

Microeconomic theory ensures that under certain technical conditions these complete ordinal rankings define a numerical utility function. The utility function is unique up to a linear transformation (Debreu, 1983; Hildenbrand & Kirman, 1976; Owen, 1995). So we can in theory replace the ordinal claim ''I like object A at least as much as I like object B'' with some cardinal relation $u(A) \geq u(B)$ and vice versa. The utility function $u: O \rightarrow R$ converts the ordinal preference structure into a numerical utility surface in an object space O of low or high dimension (Debreu, 1983; Hildenbrand & Kirman, 1976; Owen, 1995). The user likes the surface's peak objects and dislikes its valley objects.

We use neural fuzzy systems to learn the user's profile or utility surface as a set of adaptive fuzzy if-then rules. The rules compress the profile into modular units. The rules grow the profile from a first set of sample data or question-answer queries and change the profile's shape as the agent samples more preference data. The modular structure of the rules lets the user add or delete knowledge chunks or heuristics.

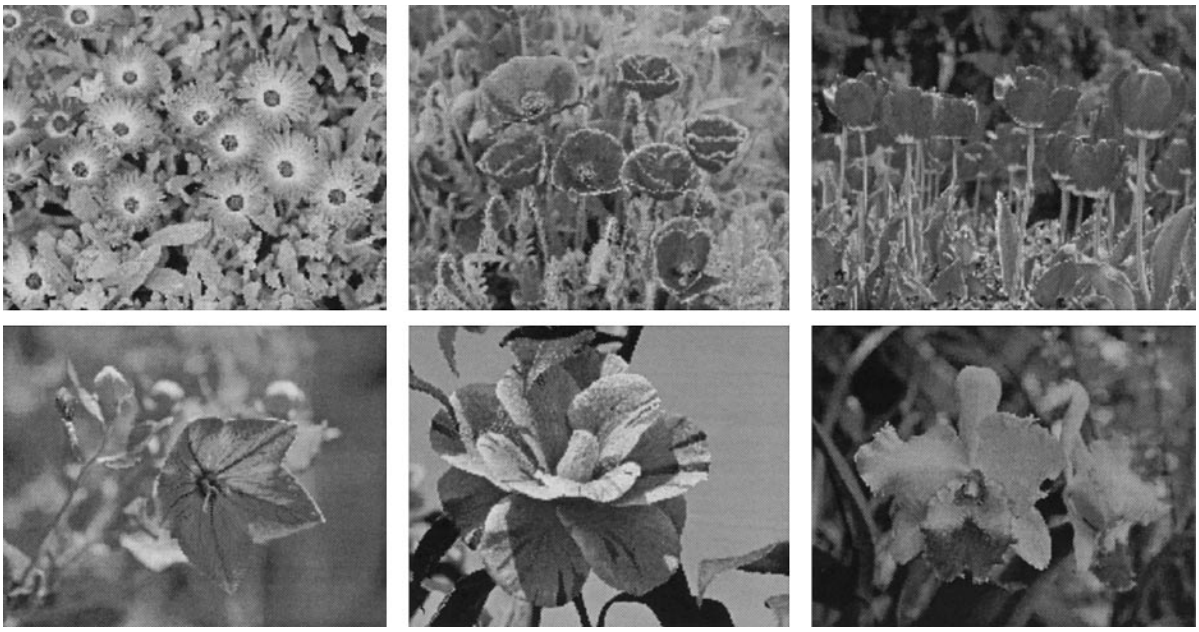


Figure 2. *Search Objects. Samples of flower images in the test database. (With permission: Hitachi Viewseum, Copyright ©1995, 1996, 1997, Hitachi, America, Ltd. All rights reserved.)*

These fuzzy systems are universal approximators (Kosko, 1994) but they suffer from exponential rule explosion in high dimension (Kosko, 1995b). Their first set of rules give a quick but rough approximation of the user's profile. Each rule defines a fuzzy patch or subset of the object space (or product object space). Mean-square optimal rules cover the extrema or bumps of the profile surface (Kosko, 1995b). Then other rule patches tend to quickly fill in between these bumps as learning unfolds. Figure 2 shows some of the flower test images we used to form a 4-D feature space of objects. Figure 3 shows how a neural fuzzy system with 100 rules approximates a 2-D profile surface. The utility profiles grow finer as the user states more numerical ranks for test objects or pattern clusters. Rule explosion remains the chief limit to this approach.

We also combine neural learning and fuzzy set theory to search for preferred objects. We cast this search problem as one of fuzzy similarity matching and define a new measure for the task and show how supervised learning updates this measure. The user gives the system matching degrees in the unit interval for a test space of sunset

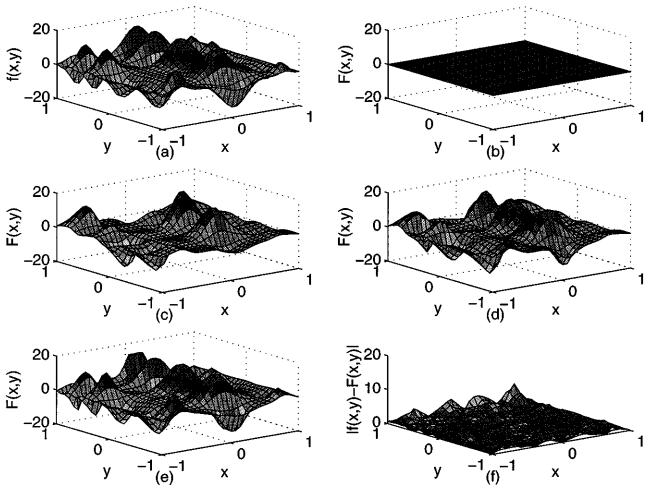


Figure 3. *Fuzzy function approximation. 2-D sinc standard additive model (SAM) function approximation with 100 fuzzy if-then rules and supervised gradient descent learning. (a) Desired function or approximand* f. *(b) SAM initial phase as a flat sheet or constant approximator* F. *(c) SAM approximator* F *after it initializes its centroids to the samples:* $c_j = f(m_j)$. *(d) SAM approximator* F *after 100 epochs of learning. (e) SAM approximator* F *after 6000 epochs of learning. (f) Absolute error of the fuzzy function approximation* ($|f - F|$).

images. Supervised gradient descent tunes the measure and defines a similarity surface over the sunset object space. Similar objects have nearly the same utility, but objects with the same utility need not be similar. Other systems might combine the ''smart'' techniques of fuzzy profile learning with fuzzy object matching to aid in the agent search process.

2 Neural Fuzzy Function Approximation: Patch the Bumps

This section reviews the basic structure of additive fuzzy systems. The appendices review and develop the more formal mathematical structure that underlies the neural fuzzy agent systems.

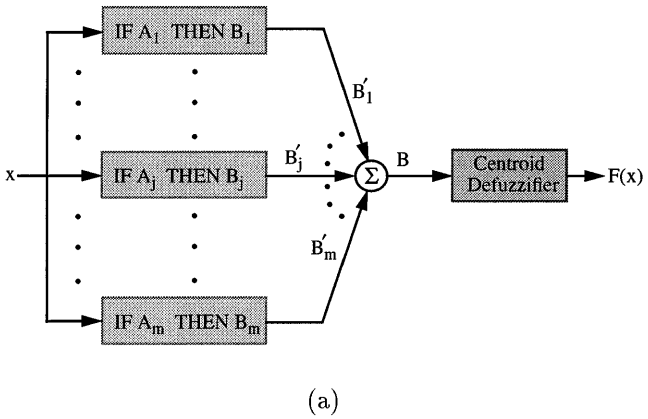
A fuzzy system F: $R^n \rightarrow R^p$ stores m rules of the word form ''IF $X = A_j$ THEN Y $= B_j$'' or the patch form $A_j \times B_j \subset X \times Y = R^n \times R^p$. The if-part fuzzy sets $A_j \subset R^n$ and then-part fuzzy sets $B_j \subset R^p$ have set functions $a_j : R^n \rightarrow [0, 1]$ and $b_j : R^p \rightarrow [0, 1]$. The system can use the joint set function a_j or some factored form such as $a_j(x) = a_j^1(x_1) \cdots a_j^n(x_n)$ or $a_j(x) =$ $\min(a_j^1(x_1), \ldots, a_j^n(x_n))$ or any other conjunctive form for input vector $x = (x_1, \ldots, x_n) \in R^n$.

An additive fuzzy system (Kosko, 1991, 1994) sums the ''fired'' then-part sets B'_j:

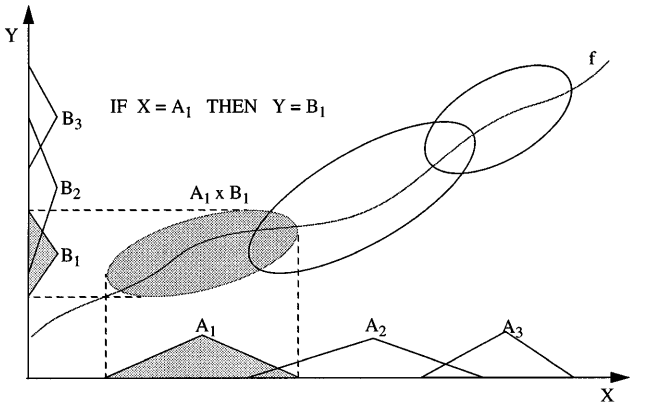
$$B(x) = \sum_{j=1}^{m} w_j B'_j = \sum_{j=1}^{m} w_j a_j(x) B_j. \qquad (1)$$

Figure 4a shows the parallel fire-and-sum structure of the standard additive model (SAM). These systems can uniformly approximate any continuous (or bounded measurable) function f on a compact domain (Kosko, 1994).

Figure 4b shows how three rule patches can cover part of the graph of a scalar function $f : R \rightarrow R$. The patch cover shows that all fuzzy systems $F : R^n \rightarrow R^p$ suffer from *rule explosion* in high dimensions. A fuzzy system F needs on the order of k^{n+p-1} rules to cover the graph and thus to approximate a vector function $f : R^n \rightarrow R^p$. Optimal rules can help deal with the exponential rule explosion. Lone or local mean-squared optimal rule



(a)



(b)

Figure 4. *Feedforward fuzzy function approximators. (a) The parallel associative structure of the additive fuzzy system. Each input vector $x \in R^n$ fires each of m fuzzy rules to some degree. This gives a convex sum as the output:* $F(x) = \text{Centroid}(\Sigma_{j=1}^{m} w_j a_j(x) B_j) = \Sigma_{j=1}^{m} p_j(x) c_j$, *where* $p_1(x) + \cdots + p_m(x) = 1$, $p_j(x) \geq 0$, *and* c_j *is the centroid of the* j*th then-part fuzzy set* B_j. *Each input* $\times$ *gives a new set of convex coefficients* $p_1(x), \ldots, p_m(x)$. *(b) Fuzzy rules define Cartesian rule patches* $A_j \times B_j$ *in the input-output space and cover the graph of the approximand f. This leads to exponential rule explosion in high dimensions. Optimal lone rules cover the extrema of the approximand.*

patches cover the extrema of the approximand f (Kosko, 1995b). They ''patch the bumps.'' Better learning schemes move rule patches to or near extrema and then fill in between extrema with extra rule patches if the rule budget allows.

The scaling choice $B'_j = a_j(x)B_j$ gives a *standard additive model* or SAM. Appendix A shows that taking the

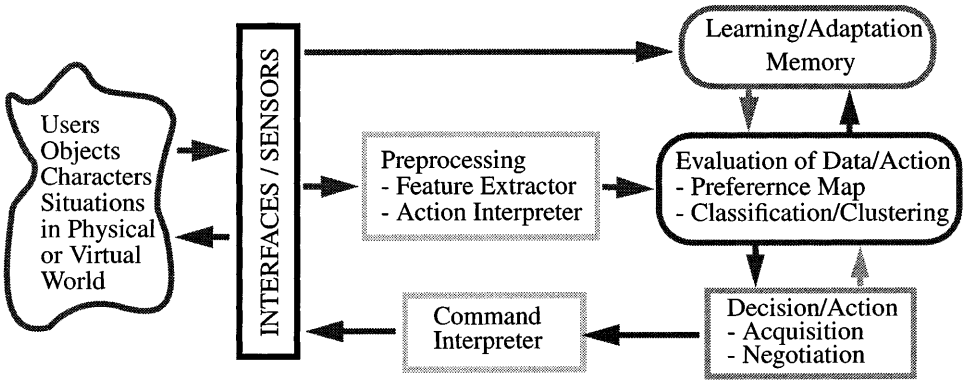


Figure 5. *Agent environment. Schematic view of an autonomous agent in a physical or virtual world. The agent interacts with objects or characters in the environment and adapts itself to better execute its goals.*

centroid of $B(x)$ in (1) gives (Kosko, 1991, 1994, 1995a, 1995b) the SAM ratio

$$F(x) = \frac{\sum_{j=1}^{m} w_j a_j(x) V_j c_j}{\sum_{j=1}^{m} w_j a_j(x) V_j} = \sum_{j=1}^{m} p_j(x) c_j. \qquad (2)$$

Here V_j is the finite positive volume or area of then-part set B_j and c_j is the centroid of B_j or its center of mass. The convex weights $p_1(x), \ldots, p_m(x)$ have the form

$$p_j(x) = \frac{w_j a_j(x) V_j}{\sum_{i=1}^{m} w_i a_i(x) V_i}.$$

Figure 3 shows how supervised learning moves and shapes the fuzzy rule patches to give a finer approximation as the system samples more user choices. Appendix B derives the supervised SAM learning algorithms for Laplace and sinc set functions (Kosko, 1996; Mitaim & Kosko, 1996). Supervised gradient descent changes the SAM parameters with error data. At each time instant t the system takes an input-output pair (x_t, y_t) from a training data set or from sensor data. A user may define this input-output data pair during the Q&A session or in a feedback or evaluation processes. Then the fuzzy system computes output vector $F(x_t)$ from input vector x_t. The learning laws update each SAM parameter to minimize the squared-error $E(x_t) = \frac{1}{2}(f(x_t) - F(x_t))^2$. This process repeats as needed for a large number of sample data pairs (x_t, y_t). Learning moves and shapes the rule patches that define the SAM system F and gives a finer approximation of f. Figure 3f displays the absolute error of the fuzzy function approximation.

3 Agent Architecture

Figure 5 shows our schematic view of an intelligent agent. The agent can reside in a physical world (robot) or in a virtual world (softbot) (Jennings & Wooldridge, 1996; Maes, Darrell, Blumberg, & Pentland, 1996). The interface / sensor module transforms the information into a bit stream. The preprocessor compresses the pattern of objects or actions. The compressed patterns might be colors or textures used in image search or filtering (Niblack, Barber, Equitz, Flikner, Glassman, Petkovic, Yanker, & Faloutsos, 1993; Swain & Ballard, 1991), keywords used in text search or e-mail classifiers or news-filtering agents (Maes, 1994), or object features that agents use if they bargain or negotiate (Chavez & Maes, 1996; Parsons & Jennings, 1996; Reilly & Bates, 1995; Rosenschein & Zlotkin, 1994; Sandholm & Lesser, 1995).

A learning and memory module records the compressed patterns of the utility surface. The surface changes over time as the user gives more Q&A samples. This gives a bumpy surface that tends to better and better match the user's underlying preference map.

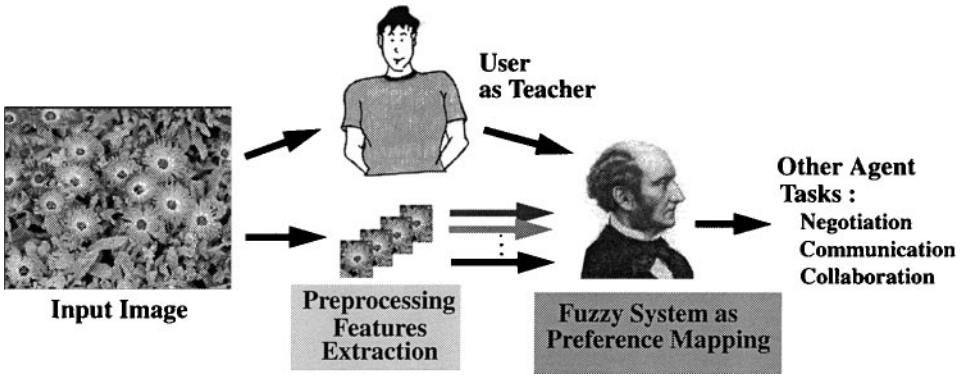


Figure 6. *Data acquisition. A fuzzy agent can learn a user's unknown preference map. The user acts as a teacher or supervisor and gives the system question-answer training samples. Then supervised gradient descent tunes the fuzzy system to better approximate the user's preference map.*

The decision-maker module receives the data from the evaluation module and then decides what to do (Maes, 1995b). A classifier agent sends the control signal to that class to which the object belongs (Maes, 1994). Then an agent must decide which step to take next. The agent may need to bargain or negotiate with other agents (Chavez & Maes, 1996; Reilly & Bates, 1995).

This paper deals largely with the block that computes the ''value'' or ''worth'' of an object or action. The preference map $u : O \rightarrow R$, defines the value of each object. The user prefers object O_1 to object O_2 (or $O_1 \succeq O_2$ in preference notation) if and only if $u(O_1) \geq u(O_2)$. Information agents need some form of these preference maps to decide search issues on their user's behalfs (Keeney & Raiffa, 1976; Kirman, Nicholson, Lejter, Dean, Santos, 1993; Mullen & Wellman, 1995; Wellman & Doyle, 1991). A fuzzy function approximator can give a good approximation of the preference map if the fuzzy system does not need too many rules and if the system can sample enough accurate user preference data. We also suggest a method to elicit consistent user preference data.

4 Profile Learning with Sunsets and Flowers

Users can define preference maps on an image space of sunsets or flowers. Each person has his own likes or dislikes that define his own fuzzy pattern of object clusters. The clusters depend on the features that define the objects. Recent work on object recognition (Swain & Ballard, 1991) and content-based image retrieval (Niblack et al., 1993) suggests that features define the ''look'' of the images. These features include colors, shapes, and textures. Research in machine vision seeks invariant features that can map all images into smaller clusters (Caelli & Reye, 1993; Chang & Smith, 1995; Funt, 1995; Niblack et al., 1993; Pentland, Picard, & Sclaroff, 1994; Picard & Minka, 1995; Swain & Ballard, 1991; Wu, Narasimhalu, Mehtre, & Gao, 1995).

Figure 6 shows a block diagram of a neural fuzzy agent that learns a user profile in a space of images. We used a multi-dimensional histogram of an image as features for our fuzzy agent prototype. Niblack et al. (1993) and Swain and Ballard (1991) used color histograms to recognize images and to structure their image database retrieval systems. The histogram technique itself ignores the spatial correlation of pixels in images. This has led many researchers to suggest other local features (Caelli & Reye, 1993; Niblack et al., 1993; Pentland et al., 1994). We use the image dispersion σ_{ij} as an extra feature (Pratt, 1991):

$$\sigma_{ij} = \frac{1}{W^2}\left[\sum_{m=-w}^{w}\sum_{n=-w}^{w} \times [x(i+m, j+n) - \bar{x}(i+m, j+n)]^2\right]^{1/2} \quad (3)$$

where $W = 2w + 1$ and where

$$\bar{x}(i, j) = \frac{1}{W^2} \sum_{m=-w}^{w} \sum_{n=-w}^{w} x(i + m, j + n) \qquad (4)$$

defines the sample mean in the $W \times W$ window centered at pixel location (i, j).

For each image we obtain its 4-D normalized histogram. The first three components are hue (h), saturation (s), and intensity (v) in the hue-saturation-intensity color space (Pratt, 1991). The other component is the standard deviation σ of the intensity component. We view this normalized 4-D histogram as an input discrete probability density function to the fuzzy system and write it in the form

$$T(h, s, v, \sigma) = \sum_{i=1}^{N_h} \sum_{j=1}^{N_s} \sum_{k=1}^{N_v} \sum_{l=1}^{N_\sigma} t_{i,j,k,l} \delta(h - \bar{h}_i) \times \delta(s - \bar{s}_j)\delta(v - \bar{v}_k)\delta(\sigma - \bar{\sigma}_1). \qquad (5)$$

Here N_h, N_s, N_v, and N_σ are the number of bins on axes of hue, saturation, intensity, and standard deviation. So the total number of histogram bins is $N = N_h \times N_s \times N_v \times N_\sigma$. The term $\bar{h}_i$ is the bin center of the ith hue and likewise for $\bar{s}_j$, $\bar{v}_k$, and $\bar{\sigma}_l$. The term $t_{i,j,k,l}$ is a normalized frequency of occurrence of the feature vector $(\bar{h}_i, \bar{s}_j, \bar{v}_k, \bar{\sigma}_l)$. We write the N-bin histogram T in the more compact form

$$T(h, s, v, \sigma) = T(x) = \sum_{n=1}^{N} t_n \delta(x - \bar{x}_n). \qquad (6)$$

The vector $\bar{x}_n$ has the center of the histogram bin as its components: $\bar{x}_n = (\bar{h}_{i_n}, \bar{s}_{j_n}, \bar{v}_{k_n}, \bar{\sigma}_{l_n})$ as in (5). The normalized frequency of occurrence t_n replaces the corresponding t_{i_n,j_n,k_n,l_n} in (5).

This histogram T is the input to the fuzzy system. Appendix A shows that this gives a generalized SAM ratio (2) (Kosko, 1991, 1996) as a *set* SAM system:

$$F(T) = \frac{\sum_{j=1}^{m} a_j(T) V_j c_j}{\sum_{j=1}^{m} a_j(T) V_j} = \sum_{j=1}^{m} p_j(T) c_j. \qquad (7)$$

The convex coefficients $p_j(T) \geq 0$ and $\sum_{j=1}^{m} p_j(T) = 1$ have the form

$$p_j(T) = \frac{a_j(T) V_j}{\sum_{i=1}^{m} a_i(T) V_i}. \qquad (8)$$

The correlation of a fuzzy set function $a_j : X \subset R^4 \rightarrow [0, 1]$ with a 4-D histogram of an image T has the form

$$a_j(T) = \int_X a_j(h, s, v, \sigma) T(h, s, v, \sigma)\, dh\, ds\, dv\, d\sigma \qquad (9)$$

$$= \sum_{n=1}^{N} t_n a_j(\bar{x}_n). \qquad (10)$$

The value $a_j(T)$ states the degree to which fuzzy set T belongs to fuzzy set A_j. The set correlation $a_j(T)$ need not lie in the unit interval. It can take on any finite nonnegative value: $a_j(T) \in [0, \infty)$. The set SAM ratio in (7) still gives an output as a convex sum of the then-part set centroids c_j as the point SAM in (2).

We tested the fuzzy agents with 88 flower images and 42 sunset images. Figure 2 shows some of the test images. We assigned subjective values to all images as numbers from 0 to 10. The value 10 stands for ''It is maximally beautiful'' or ''I really love it.'' The value 0 stands for ''It is minimally beautiful'' or ''I really hate it.'' The histogram bins were 8:4:4:4 for h:s:v:σ. So there were a total of 512 bins. The fuzzy system also had 512 fuzzy rules. We initialized the fuzzy agent so that it would be ''indifferent'' to all images (a score of 5) and trained it with supervised gradient-descent learning. The initial maximum absolute error was 5, and the mean absolute error was 2.45. The fuzzy agent converged after 40,000 epochs to our preference map and gave a score close to ours. This held for almost all test images. The maximum absolute error was 0.96 and the mean absolute error was 0.18. This error stemmed from too few features. Using more features tends to improve the system's accuracy but at the expense of greater rule complexity.

We used a histogram based on color and variance because it captured the relative amount of colors in the image that affect much of human perception (Niblack et al., 1993; Swain & Ballard, 1991). We can also compute histograms easily and they are translation and rotation

invariant (Swain & Ballard, 1991). Our systems for profile learning and searching did not depend on how we chose object features. The fuzzy agent could use other inputs from this image database or from others. These input features might include shapes (Niblack et al., 1993; Pentland et al., 1994), textures (Caelli & Reye, 1993; Niblack et al., 1993; Pentland et al., 1994; Picard & Minka, 1995), wavelet transforms (Chang & Smith, 1995; Vetterli & Kovačević, 1995), or other statistical measures (Pentland et al., 1994).

5 Adaptive Fuzzy Object Matching

This section presents fuzzy equality as a measure of similarity between objects and shows how to tune it. A search or filter agent matches objects in the databases to the query object and acts on the match results. Supervised learning tunes the fuzzy equality measure to better approximate the user's perception of similar objects.

A fuzzy system can assist in database search in many ways. Fuzzy matching is perhaps the simplest way. The fuzzy equality measure (Kosko, 1996) between two fuzzy sets can define the similarity between objects. The equality measure $E(A, B)$ measures the degree to which fuzzy set A equals fuzzy set B. It measures how well A matches B and vice versa. Suppose fuzzy sets A and B are nonempty. Then, $E(A, B) = E(B, A) \in [0, 1]$, $E(A, A) = 1$, and $E(A, \emptyset) = 0$ for the empty set $\emptyset$. The equality measure depends on the *counting* or *cardinality* (Kosko, 1991) function c of a fuzzy set as

$$E(A, B) = \text{Degree}(A = B) = \frac{c(A \cap B)}{c(A \cup B)} = \frac{\int \min(a(x), b(x))\, dx}{\int \max(a(x), b(x))\, dx} \quad (11)$$

where

$$c(A) = \sum_{i=1}^{N} a_i \quad \text{or} \quad c(A) = \int_{R^n} a(x)\, dx \quad (12)$$

for an integrable fuzzy set function $a : X \rightarrow [0, 1]$. The fuzzy equality measure rests on the theory of fuzzy sets as points in unit hypercubes or fuzzy cubes. Appendix C reviews this unit-cube geometry of discrete fuzzy sets (Kosko, 1991, 1996).

Consider an example. Let $a = (0.8 \;\; 0.4 \;\; 0)$ and $b = (0.1 \;\; 0.5 \;\; 0.2)$ be discrete set functions for fuzzy sets A and B in $X = \{x_1, x_2, x_3\}$. So the set function or fit vector $a = (a_1 \; a_2 \; a_3)$ defines the fuzzy set A as $a_1 = a(x_1) = 0.8$, $a_2 = a(x_2) = 0.4$ and $a_3 = a(x_3) = 0$. The fit vector b defines the fuzzy set B as $b_1 = b(x_1) = 0.1$, $b_2 = b(x_2) = 0.5$, and $b_3 = b(x_3) = 0.2$. Then fuzzy set A equals fuzzy set B to degree one-third:

$$E(A, B) = \text{Degree}(A = B) = \frac{c(A \cap B)}{c(A \cup B)} \quad (13)$$

$$= \frac{\sum_{i=1}^{3} \min(a_i, b_i)}{\sum_{i=1}^{3} \max(a_i, b_i)} \quad (14)$$

$$= \frac{0.1 + 0.4 + 0}{0.8 + 0.5 + 0.2} = \frac{1}{3}. \quad (15)$$

A fuzzy system maps two objects (or their two vectors of ''features'') to the output fuzzy sets A and B. Then the equality measure gives a value near 1 if the two objects match well or ''look alike.'' It gives a value near 0 if they match poorly.

We use the same histogram features as in the prior section to match images. Let T_A and T_B be the histograms of two images. Again we view these two normalized N-bin histograms as discrete probability density functions whose domain $X = \{\bar{x}_1, \ldots, \bar{x}_N\}$ is a set of vectors $\bar{x}_i$ that define the bin centers. This gives the same form as in (6). Then we compute the correlation of a set function a_j with two histograms T_A and T_B as in (10) with

$$A_j = a_j(T_A) = \sum_{n=1}^{N} T_A(\bar{x}_n) a_j(\bar{x}_n) \quad (16)$$

$$B_j = a_j(T_B) = \sum_{n=1}^{N} T_B(\bar{x}_n) a_j(\bar{x}_n). \quad (17)$$

This gives two m-D vectors of set values $(A_1, \ldots, A_m)$ and $(B_1, \ldots, B_m)$ from m fuzzy rules. The standard ad-

ditive structure of fuzzy systems suggests that the output fuzzy set should equal the sum of the scaled then-part sets (Kosko, 1996). So we define the then-part sets to be the same as the if-part sets. So the output fuzzy sets A and B from the histograms T_A and T_B have the form

$$A(x) = \sum_{j=1}^{m} A_j a_j(x) \tag{18}$$

$$B(x) = \sum_{j=1}^{m} B_j a_j(x) \tag{19}$$

where $x = (h, s, v, \sigma) \in X$. The input to the system is an N-bin histogram on the discrete domain $X = \{\bar{x}_1, \ldots, \bar{x}_N\}$. Then we can view the output sets A and B as discrete sets and rewrite (18)–(19) as

$$A(\bar{x}_n) = \sum_{j=1}^{m} A_j a_j(\bar{x}_n) \tag{20}$$

$$B(\bar{x}_n) = \sum_{j=1}^{m} B_j a_j(\bar{x}_n) \tag{21}$$

for $n = 1, \ldots, N$. Then the fuzzy equality (in the discrete case) in (11) measures the degree to which fuzzy set A equals or matches fuzzy set B:

$$E(A, B) = \frac{\sum_{i=1}^{N} \min\left(A(\bar{x}_i), B(\bar{x}_i)\right)}{\sum_{i=1}^{N} \max\left(A(\bar{x}_i), B(\bar{x}_i)\right)} \tag{22}$$

This in turns measures the ''similarity'' between two images.

The similarity measure depends on how we define the m fuzzy rules. Tuning or learning schemes can move or reshape the fuzzy sets to approximate desired matching values. Appendix C derives the learning laws that tune the set-function parameters in E.

Figure 7 shows a block diagram of how a fuzzy agent matches images. The simulation used a 4-D version of the 1-D Laplace set function $a_j(x) = \exp\{-|(x - m_j)/d_j|\}$ in (107)–(108). We trained the fuzzy matching system on a space of sunset images with the histogram intersection in Swain and Ballard (1991):

$$S(H, I) = \frac{\sum_{i=1}^{N} \min\left(H_i, I_i\right)}{\sum_{i=1}^{N} H_i}. \tag{23}$$

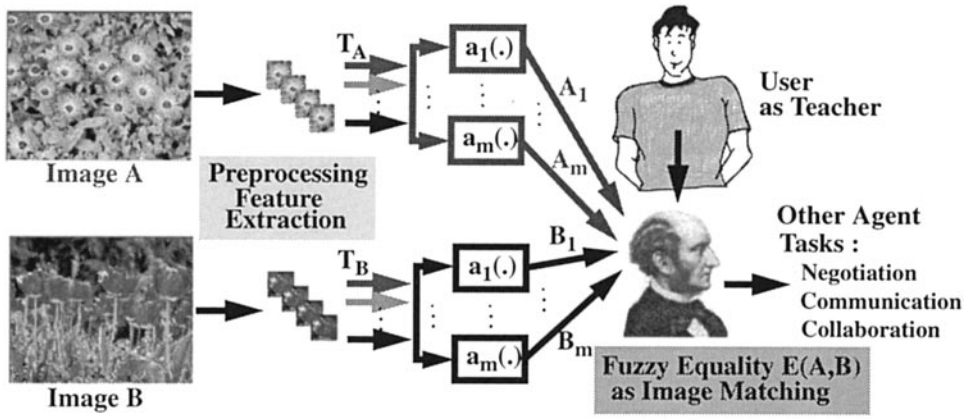
The fuzzy system gave a rough approximation of the histogram intersection. We may not be able to find a closed-form formula for matching in the general case. Then the fuzzy matching process might learn from Q&A sessions or from other user feedback.

6 The Agent-User Interface: The Q&A Bottleneck

How does an agent get numerical values for sample objects? What questions should the agent ask the user in a Q&A session? How many objects must a user rank? These questions reveal the practical weakness of any search system that depends on numbers. Cardinal data eases numerical processing but comes at the expense of a question-answer bottleneck.

This section reviews some of the techniques used in decision theory to rank objects. We show how to apply the technique to obtain numerical values for all sample objects. Other criteria can help agents ask users new questions in Q&A sessions.

Suppose a user states a subjective numerical value for each sample object in a list. There are problems with this absolute valuation beyond its artificial nature and the sheer inconvenience it forces on the user. Miller (1956) observed that the largest number of objects that our minds can process at one time is the ''magic number'' 7 ± 2. We tend to forget how we have ranked objects at the top of the list when we rank objects at the bottom of the list. Relative rankings can increase the ''capacity'' of our information processing (Miller, 1956). Then techniques in decision theory allow us to rank objects with only *pairwise* comparisons (Cook & Kress, 1992; Keeney & Raiffa, 1976; Kendall & Gibbons, 1990; Saaty, 1977). We can compute these relative object weights and convert them to the user's absolute weights.



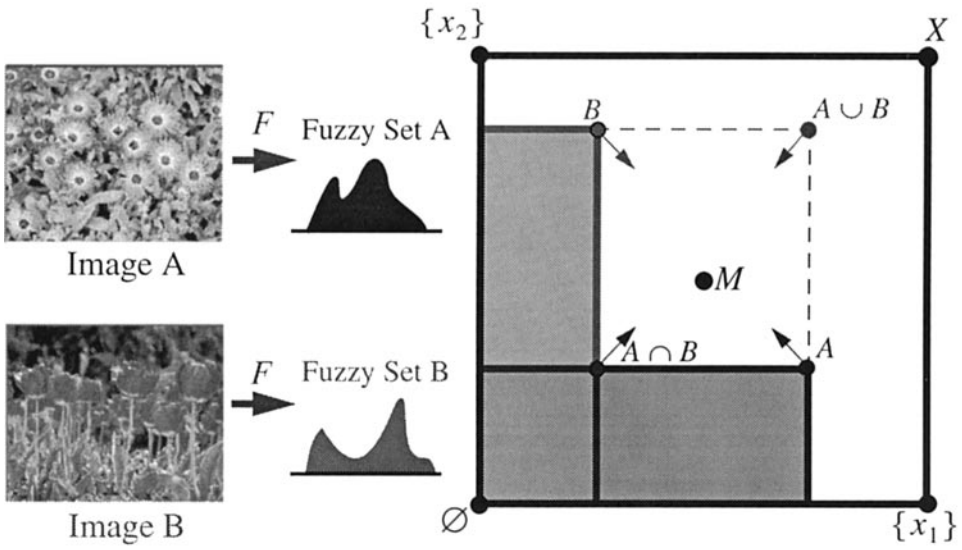


Figure 7. *Adaptive fuzzy search. Fuzzy equality measures the likeness of two objects* A *and* B. *Supervised learning tunes the fuzzy equality measure* $E(A, B)$ *inside the fuzzy-cube state space to better approximate the user's perception of similar images. The equality measure grows to unity as the* A *and* B *set points approach each other. The cube midpoint* M *is the maximally fuzzy set whereh* $E(M, M^c) = 1$. *Binary sets* V *lie at the* 2^n *cube vertices and they alone give* $E(V, V^c) = 0$.

Saaty's analytic hierarchy process (AHP) (Saaty, 1977, 1986, 1994) can find the numerical values. AHP computes the relative weights $w = (w_1, \ldots, w_n)$ of n objects from their pairwise comparisons. Let a_{ij} be a ratio scale (Saaty, 1977) of comparison of object O_i and O_j. Then a reciprocal matrix $A = [a_{ij}]$ has its elements of the form $a_{ij} = 1 / a_{ji}$ for $i, j = 1, \ldots, n$. So the diagonal entries are always unity: $a_{ii} = 1$ for all $i = 1, \ldots, n$. The claim ''I like object O_1 twice as much as I like object O_2'' gives $a_{12} = 2$. Its reciprocal $a_{21} = \frac{1}{2}$ gives the claim ''I like object O_2 half as much as I like object O_1.'' The principle eigenvector w of a matrix A obeys the equation $Aw = \lambda_{max} w$. The Perron-Frobenius theorem of matrix algebra ensures that λ_{max} is the unique maximum (positive) eig-

envalue of A (Franklin, 1968). The components of w are always positive and allow us to recover the relative object weights (Saaty, 1977).

Suppose reciprocal matrices A and B for objects O_1, O_2, O_3, and O_4 have the values

A	O_1	O_2	O_3	O_4	w
O_1	1	1/2	5	1	0.238
O_2	2	1	10	2	0.476
O_3	1/5	1/10	1	1/5	0.048
O_4	1	1/2	5	1	0.238

B	O_1	O_2	O_3	O_4	v
O_1	1	1/2	5	1	0.242
O_2	2	1	7	3	0.494
O_3	1/5	1/7	1	1/4	0.056
O_4	1	1/3	4	1	0.208

The matrix A contains many claims such as ''I like object O_2 twice as much as I like object O_1 and ten times as much as I like object O_3. I like the object O_1 as much as I like O_4.'' Users may prefer to say that they like an object O_2 ten times as much as they like O_3 than to say that they like O_3 one-tenth as much as they like O_2. Agents can offer users both options.

The matrix A is ''consistent'' but B is not. A matrix is consistent when its elements a_{ik} obey $a_{ik} = a_{ij}a_{jk}$ for all $i, j, k = 1, \ldots, n$ (Saaty, 1977). Each row of a consistent matrix is a multiple of the first row. Consistency also implies transitivity: the claim $a_{23} = a_{21}a_{13}$ implies that, if we prefer O_2 to O_1 ($O_1 \succ O_2$) and if we prefer O_1 to O_3 ($O_1 \succ O_3$), then we prefer O_2 to O_3 ($O_2 \succ O_3$). But pairwise comparisons are often inconsistent. So the weight matrix may look more like B than like A. The maximum eigenvalue λ_{max} obeys $\lambda_{max} \geq n$. The equality $\lambda_{max} = n$ holds if and only if the reciprocal matrix A is consistent. The consistency measure $\mu = (\lambda_{max} - n)/(n - 1)$ can help the agent decide whether it needs to ask a user to verify the rankings (Saaty, 1977, 1994).

The principle eigenvectors, $w = (w_1, w_2, w_3, w_4)$ and $v = (v_1, v_2, v_3, v_4)$, reflect the relative weights of objects from comparison matrices A and B. The preference order from A is $O_2 \succ O_1 \sim O_4 \succ O_3$, where $\sim$ denotes the indifference preference between two objects. The preference order from B is $O_2 \succ O_1 \succ O_4 \succ O_3$. Then a linear (affine) transformation L relates the weights to the user's subjective values: $u(O_i) = L(w_i) = cw_i + d$ for some $c > 0$ and some $d \in R$. This holds because we assume that w measures the relative utility of objects and because a linear transformation preserves the structure of a utility function (Debreu, 1983; Hildenbrand & Kirman, 1976; Owen, 1995).

An agent must interact with the user to get the transform coefficients c and d. The agent picks any two objects that have different weights and asks the user to give nonnegative weights for both objects. Then the agent can solve for the coefficients c and d and find the rest of the object weights. Arbitrary positive values for two mid-rank objects can give negative values for low-rank objects. So the highest-rank and lowest-rank objects are often the best choice. Suppose for matrix B that $u(O_2) = 10$ and $u(O_3) = 1$. Then $c = 20.55$, and $d = -0.15$. This gives $u(O_1) = cv_1 + d = 20.55 \times 0.242 - 0.15 = 4.82$, and $u(O_4) = 20.55 \times 0.208 - 0.15 = 4.12$.

Consistency gives a linear ranking complexity. A consistent user needs to give only $n - 1$ pairwise rankings to construct a matrix. Then we can deduce the other entries from $a_{ij} = a_{ik}a_{kj}$ for all $i, j, k = 1, \ldots, n$. But humans are seldom consistent. So an agent may need the user to give $\frac{1}{2}(n^2 - n)$ rankings to form a matrix. This may not be practical for large n. The agent can estimate the matrix entries with the geometric mean of all paths in the matrix from the first n rankings (Harker, 1987). The agent can also use other criteria (Harker, 1987; Millet & Harker, 1990) to ask the user for additional pairwise rankings. This can reduce the number of rankings from $\frac{1}{2}(n^2 - n)$ rankings to on the order of n rankings.

A final remark concerns scaling fuzzy profiles. A linear transformation L both preserves the structure of a preference map (Debreu, 1983; Hildenbrand & Kirman, 1976; Owen, 1995) and preserves how the agent's neural fuzzy system models the preference map. This always holds for the SAM fuzzy system F. Suppose the linear

transformation L consists of the matrix C and the vector d. Then (2) gives

$$L(F(x)) = CF(x) + d = C\left(\sum_{j=1}^{m} p_j(x)c_j\right) + d \quad (24)$$

$$= \sum_{j=1}^{m} p_j\,(x)\,(Cc_j) + d \quad (25)$$

$$= \sum_{j=1}^{m} p_j(x)(Cc_j + d) \quad (26)$$

$$\text{since} \sum_{j=1}^{m} p_j(x) = 1$$

$$= \sum_{j=1}^{m} p_j(x)c_j' \quad (27)$$

where $c_j' = Cc_j + d = L(c_j)$. So an agent can change the profile to match the user's new scale or new set of data. A neural fuzzy agent may also want to rescale its profiles for large hierarchical systems in which each system level has its own ratio scale.

7 Conclusion

Neural fuzzy systems can assist agents in many ways. We have shown how these adaptive function approximators can both help learn a user's preference map and help choose preferred search objects cast as features of low dimension. The color histogram we used did not give a complete set of features. Other neural fuzzy systems can more fully combine these two fuzzy tasks to aid in agent database search. Future research may depend on advances in pattern recognition and machine vision. Neural fuzzy systems might also assist agents when agents bargain (Chavez & Maes, 1996; Reilly & Bates, 1995; Rosenschein & Zlotkin, 1994) or cooperate (Dorigo, Maniezzo, & Colorni, 1996; Moukas, 1996) with other agents. Then an agent may try to learn a second or third user's profile as well as learn its master's profile.

Agents could also help neural fuzzy systems approximate functions from training samples. Today most neural fuzzy systems work with just one fuzzy system and one supervised or unsupervised learning law. Rule explosion in high dimensions may force the user to replace the lone fuzzy system with several smaller systems. Agents can help combine these fuzzy systems (Kosko, 1995a, 1996) if they pick and change the weights or rankings of each system based on sample data or domain knowledge. Agents can also pick which learning law or which set of parameters to use as the system tunes its rules on-line. Still more complex hybrids can use nested agents within multi-system function approximators and use the approximators to help higher-level agents learn profiles and search databases and perhaps perform other agent tasks.

The neural fuzzy agent needs to improve how it acquires knowledge (Kilpatrick, Gunsch, & Santos, 1996; Santos & Banks, 1996). The agent should not ask the user too many questions. The agent needs to learn the user's profile fast enough before it tires the user. Efficient agents would make the user state rankings that are at most linear in the number of search objects or search-object clusters. Our system asks the user a large number of numerical questions even though the user may not want to and perhaps cannot give precise numerical answers to these questions. Researchers have long searched for techniques that can lessen the number of numerical questions the system must ask the user (Kilpatrick, Gunsch, & Santos, 1996; Santos & Banks, 1996). The bootstrap and other statistical methods (Efron & Tibshirani, 1993) may offer more efficient ways for an adaptive agent to sample its user and its environment. Ordinal or chunking techniques (Laird, Newell, & Rosenbloom, 1987; Miller, 1956; Newell & Rosenbloom, 1981) may also ease the burden of preference acquisition. But all such techniques tend to increase the complexity of the neural and fuzzy systems.

8 Acknowledgement

A research grant from the Annenberg Center for Communication at the University of Southern California partly funded this research.

References

Brooks, R. (1995). Intelligence Without Reason. In Steels, L. & Brooks, R. (Eds.), *The Artificial Life Route to Artificial Intelligence: Building Embodied, Situated Agents,* (pp. 25–81). Lawrence Erlbaum Associates.

Caelli, T. & Reye, D. (1993). On the Classification of Image Regions by Colour, Texture and Shape. *Pattern Recognition, 26* (4), 461–470.

Chang, S. F., & Smith, J. R. (1995). Extracting Multi-Dimensional Signal Features for Content-Based Visual Query. *SPIE Symposium on Visual Communications and Signal Processing.*

Chavez, A., & Maes, P. (1996). Kasbah: An Agent Marketplace for Buying and Selling Goods. *Proceedings of the Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology.*

Colombetti, M., & Dorigo, M. (1994). Training Agents to Perform Sequential Behavior. *Adaptive Behavior, 2*(3), 247–275.

Cook, W. D., & Kress, M. (1992). *Ordinal Information and Preference Structures: Decision Models and Applications.* Prentice Hall.

Debreu, G. (1983). Representation of a Preference Ordering by a Numerical Function. In *Mathematical Economics: Twenty Papers of Gerard Debreu* (pp. 105–110). Cambridge University Press.

Dorigo, M., Maniezzo, V., & Colorni, A. (1996). Ant System: Optimization by a Colony of Cooperating Agents. *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics, 26*(1), 29–41.

Efron, B., & Tibshirani, P. J. (1993). *An Introduction to the Bootstrap.* Chapman & Hall.

Franklin, J. N. (1968). *Matrix Theory.* Prentice Hall.

Funt, B. V. (1995). Color Constant Color Indexing. *IEEE Transactions on Pattern Analysis and Machine Intelligence, (17)* 5, 522–529.

Grosky, W. I. (1994). Multimedia Information Systems. *IEEE Multimedia,* (1) 1, 12–24.

Harker, P. T. (1987). Incomplete Pairwise Comparison in the Analytic Hierarchy Process. *Mathematical Modelling 9*(11), 837–848.

Hildenbrand, W., & Kirman, A. P. (1976). *Introduction to Equilibrium Analysis.* North Holland.

Jennings, N. R., & Wooldridge, M. (1996). Software Agents. *IEE Review* (42), 1, 17–20.

Keeney, R. L., & Raiffa, H. (1976). *Decision with Multiple Objectives: Preferences and Value Tradeoffs.* John Wiley & Sons.

Kendall, M., & Gibbons, J. D. (1990). *Rank Correlation Methods.* (5th ed.). Edward Arnold, A division of Hodder & Stoughton.

Kilpatrick, F. A., Gunsch, G. H., & Santos, E., Jr. (1996). Induction and State-Space Search for an Intelligent Training System. *Proceedings of the Midwest Artificial Intelligence and Cognitive Science Conference.*

Kirman, J., Nicholson, A., Lejter, M., Dean, T., & Santos, E., Jr. (1993). Using Goals to Find Plans with High Expected Utility. *Proceedings of the Second European Workshop on Planning,* 158–170.

Kosko, B. (1991). *Neural Networks and Fuzzy Systems.* Prentice Hall.

Kosko, B. (1994). Fuzzy Systems as Universal Approximators. *IEEE Transactions on Computers, 43*(11), 1329–1333.

Kosko, B. (1995a). Combining Fuzzy Systems. *Proceedings of the IEEE International Conference on Fuzzy Systems* (*IEEE FUZZ-95*), 1855–1863.

Kosko, B. (1995b). Optimal Fuzzy Rules Cover Extrema. *International Journal of Intelligent Systems, 10* (2), 249–255.

Kosko, B. (1996). *Fuzzy Engineering.* Prentice Hall.

Krishnapuram, R., & Keller, J. M. (1993). A Possibilistic Approach to Clustering. *IEEE Transactions on Fuzzy Systems, 1,* 98–110.

Laird, J. E., Newell, A., & Rosenbloom, P. S. (1987). SOAR: An Architecture for General Intelligence. *Artificial Intelligence, 33,* 1–64.

Maes, P. (1994). Agents that Reduce Work and Information Overload. *Communications of the ACM, 37* (7), 31–40.

Maes, P. (1995a). Artificial Life Meets Entertainment: Lifelike Autonomous Agents. *Communications of the ACM, 38* (11), 108–114.

Maes, P. (1995b). Modeling Adaptive Autonomous Agents. In C. G. Langton (Ed.), *Artificial Life: An Overview* (pp. 135–162). MIT Press.

Maes, P., Darrell, T., Blumberg, B., & Pentland, A. (1996). The ALIVE System: Wireless, Full-body Interaction with Autonomous Agents. *Multimedia Systems.*

Miller, G. A. (1956). The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information. *The Psychological Review, 63* (2), 81–97.

Millet, I., & Harker, P. T. (1990). Globally Effective Questioning in the Analytic Hierarchy Process. *European Journal of Operational Research, 48,* 88–97.

Mitaim, S., & Kosko, B. (1996). What is the Best Shape for a

Fuzzy Set in Function Approximation? *Proceedings of the Fifth IEEE International Conference on Fuzzy Systems* (*FUZZ-96*), *2*, 1237–1243.

Moukas, A. (1996). Amalthaea: Information Discovery and Filtering using a Multiagent Evolving Ecosystem. *Proceedings of the Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology.*

Mullen, T., & Wellman, M. P. (1995). A Simple Computational Market for Network Information Services. *Proceedings of the First International Conference on Multi-Agent Systems,* 283–289.

Newell, A., & Rosenbloom, P. S. (1981). Mechanisms of Skill Acquisition and the Law of Practice. In J. R. Anderson (Ed.), *Cognitive Skills and Their Acquisition* (pp. 1–55). Lawrence Erlbaum Associates.

Niblack, W., Barber, R., Equitz, W., Flikner, M., Glassman, E., Petkovic, D., Yanker, P., & Faloutsos, C. (1993). The QBIC Project: Querying Images by Content Using Color, Texture, and Shape. Research Report RJ 9203 (81511), IBM.

Owen, G. (1995). *Game Theory* (3rd ed.). Academic Press.

Pal, N. R., & Bezdek, J. C. (1995). On Cluster Validity for the Fuzzy c-Means Model. *IEEE Transactions on Fuzzy Systems, 3* (3), 370–379.

Pal, N. R., Bezdek, J. C., & Hathaway, R. J. (1996). Sequential Competitive Learning and the Fuzzy c-Means Clustering Algorithms. *Neural Networks, 9,* 787–96.

Parsons, S., & Jennings, N. R. (1996). Negotiation through Argumentation—A Preliminary Report. *Proceedings of the International Conference on Multi-Agent Systems.*

Pentland, A., Picard, R. W., & Sclaroff, S. (1994). Photobook: Tools for Content-Based Manipulation of Image Databases. *SPIE: Storage and Retrieval for Image and Video Database II, 2185,* 34–47.

Picard, R. W., & Minka, T. P. (1995). Vision Texture for Annotation. *Multimedia Systems, 3,* 3–14.

Pratt, W. K. (1991). *Digital Image Processing* (2nd ed.). Wiley Interscience.

Rasmus, D. W. (1995). Intelligent Agents: DAI Goes to Work. *PC AI,* 27–32.

Reilly, W. S., & Bates, J. (1995). Natural Negotiation for Believable Agents. Technical Report CMU-CS-95-164. Carnegie Mellon University, Pittsburgh, PA.

Rosenschein, J. S., & Zlotkin, G. (1994). Consenting Agents: Designing Conventions for Automated Negotiation. *AI Magazine, 15* (3), 29–46.

Saaty, T. L. (1977). A Scaling Method for Priorities in Hierarchical Structures. *Journal of Mathematical Psychology, 15,* 234–281.

Saaty, T. L. (1986). Axiomatic Foundation of the Analytic Hierarchy Process. *Management Science, 32* (7), 841–855.

Saaty, T. L. (1994). Highlights and Critical Points in the Theory and Application of the Analytic Hierarchy Process. *European Journal of Operational Research, 74,* 426–447.

Sandholm, T., & Lesser, V. (1995). Issues in Automated Negotiation and Electronic Commerce: Extending the Contract Net Framework. *Proceedings of the First International Conference on Multi-Agent Systems,* 328–335.

Santos, E., Jr., & Banks, D. O. (1996). Acquiring Consistent Knowledge. Technical Report AFIT / EN / TR96-01, Air Force Institute of Technology.

Steels, L. (1995). The Artificial Life Roots of Artificial Intelligence. In C. G. Langton (Ed.), *Artificial Life: An Overview* (pp. 75–110). MIT Press.

Swain, M. J., & Ballard, D. H. (1991). Color Indexing. *International Journal of Computer Vision, 7* (1), 11–32.

Vetterli, M., & Kovačević, J. (1995). *Wavelets and Subband Coding.* Prentice Hall.

Wellman, M. P., & Doyle, J. (1991). Preferential Semantics for Goals. *Proceedings of the Ninth National Conference on Artificial Intelligence* (*AAAI-91*), 698–703.

Wu, J. K., Narasimhalu, A. D., Mehtre, B. M., & Gao, Y. J. (1995). CORE: A Content-Based Retrieval Engine for Multimedia Information Systems. *Multimedia Systems, 3,* 25–41.

Yamauchi, B., & Beer, R. (1994). Integrating Reactive, Sequential, and Learning Behavior Using Dynamical Neural Networks. D. Cliff, P. Husbands, J. A. Meyer, & S. Wilson (Eds.), In *Proceedings of the Third International Conference on Simulation of Adaptive Behavior,* (pp. 382–391). MIT Press.

Appendix A. The Standard Additive Model (SAM) Theorem

This appendix derives the basic ratio structure (2) of a standard additive fuzzy system.

SAM Theorem. Suppose the fuzzy system $F : R^n \rightarrow R^p$ is a standard additive model: $F(x) = \text{Centroid}(B(x)) = \text{Centroid}(\sum_{j=1}^{m} w_j a_j(x) B_j)$ for if-part joint set function a_j: $R^n \rightarrow [0, 1]$, rule weights $w_j \geq 0$, and then-part fuzzy set $B_j \subset R^p$. Then $F(x)$ is a convex sum

of the m then-part set centroids:

$$F(x) = \frac{\sum_{j=1}^{m} w_j a_j(x) V_j c_j}{\sum_{j=1}^{m} w_j a_j(x) V_j} = \sum_{j=1}^{m} p_j(x) c_j. \tag{28}$$

The convex coefficients or discrete probability weights $p_1(x), \ldots, p_m(x)$ depend on the input x through

$$p_j(x) = \frac{w_j a_j(x) V_j}{\sum_{i=1}^{m} w_i a_i(x) V_i}. \tag{29}$$

V_j is the finite positive volume (or area if $p = 1$) and c_j is the centroid of then-part set B_j:

$$V_j = \int_{R^p} b_j(y_1, \ldots, y_p)\, dy_1 \cdots dy_p > 0 \tag{30}$$

$$c_j = \frac{\int_{R^p} y b_j(y_1, \ldots, y_p)\, dy_1 \cdots dy_p}{\int_{R^p} b_j(y_1, \ldots, y_p)\, dy_1 \cdots dy_p}. \tag{31}$$

Proof. There is no loss of generality to prove the theorem for the scalar-output case $p = 1$ when $F : R^n \to R^p$. This simplifies the notation. We need but replace the scalar integrals over R with the p-multiple or volume integrals over R^p in the proof to prove the general case. The scalar case $p = 1$ gives (30) and (31) as

$$V_j = \int_{-\infty}^{\infty} b_j(y)\, dy \tag{32}$$

$$c_j = \frac{\int_{-\infty}^{\infty} y b_j(y)\, dy}{\int_{-\infty}^{\infty} b_j(y)\, dy}. \tag{33}$$

Then the theorem follows if we expand the centroid of B and invoke the SAM assumption $F(x) = \text{Centroid}(B(x)) = \text{Centroid}(\Sigma_{j=1}^{m} w_j a_j(x) B_j)$ to rearrange terms:

$$F(x) = \text{Centroid}(B(x)) \tag{34}$$

$$= \frac{\int_{-\infty}^{\infty} y b(y)\, dy}{\int_{-\infty}^{\infty} b(y)\, dy} \tag{35}$$

$$= \frac{\int_{-\infty}^{\infty} y \sum_{j=1}^{m} w_j b'_j(y)\, dy}{\int_{-\infty}^{\infty} \sum_{j=1}^{m} w_j b'_j(y)\, dy} \tag{36}$$

$$= \frac{\int_{-\infty}^{\infty} y \sum_{j=1}^{m} w_j a_j(x) b_j(y)\, dy}{\int_{-\infty}^{\infty} \sum_{j=1}^{m} w_j a_j(x) b_j(y)\, dy} \tag{37}$$

$$= \frac{\sum_{j=1}^{m} w_j a_j(x) \int_{-\infty}^{\infty} y b_j(y)\, dy}{\sum_{j=1}^{m} w_j a_j(x) \int_{-\infty}^{\infty} b_j(y)\, dy} \tag{38}$$

$$= \frac{\sum_{j=1}^{m} w_j a_j(x) V_j \dfrac{\int_{-\infty}^{\infty} y b_j(y)\, dy}{V_j}}{\sum_{j=1}^{m} w_j a_j(x) V_j} \tag{39}$$

$$= \frac{\sum_{j=1}^{m} w_j a_j(x) V_j c_j}{\sum_{j=1}^{m} w_j a_j(x) V_j} \quad \square \tag{40}$$

Generalizing the SAM system leads to the *set* SAM F that maps fuzzy sets A in the input space R^n to vector points y in the output space R^p. So the set SAM $F : F(2^{R^n}) \to R^p$ has as its domain the fuzzy power set $F(2^{R^n})$ or the set of all fuzzy subsets $A \subset R^n$ with arbitrary set function $a : R^n \to [0, \infty)$. The point SAM is a special case of the set SAM for a singleton input fuzzy set $A = \{x_0\} \subset R^n$: $a(x) = \delta(x - x_0)$ where δ is a Dirac delta function in the continuous case or a unit bit vector in the discrete case. Correlation computes the ''fired'' fit value of the jth set $a_j(A)$ as (Kosko, 1996)

$$a_j(A) = \int a(x) a_j(x)\, dx. \tag{41}$$

Then the fired fit value $a_j(x_0)$ of the singleton set $A =$

$\{x_0\}$ follows from the sifting property of delta pulses:

$$a_j(A) = \int a(x)a_j(x)\,dx \tag{42}$$

$$= \int \delta(x - x_0)a_j(x)\,dx \tag{43}$$

$$= a_j(x_0). \tag{44}$$

The set SAM equation follows from the SAM additive combiner $B(A) = \Sigma_{j=1}^{m} w_j a_j(A) B_j$ (Kosko, 1996):

$$F(A) = \text{Centroid}(B(A)) \tag{45}$$

$$= \text{Centroid}\left(\sum_{j=1}^{m} w_j a_j(A) B_j\right) \tag{46}$$

$$= \frac{\sum_{j=1}^{m} w_j a_j(A) V_j c_j}{\sum_{j=1}^{m} w_j a_j(A) V_j}$$

$$= \sum_{j=1}^{m} p_j(A) c_j. \tag{47}$$

where the convex coefficients $p_1(A), \ldots, p_m(A)$ depend on the input fuzzy set A through

$$p_j(A) = \frac{w_j a_j(A) V_j}{\sum_{i=1}^{m} w_i a_i(A) V_i}. \tag{48}$$

Appendix B. Supervised SAM Learning

Supervised gradient-descent can tune all the parameters in the SAM model (2) (Kosko, 1995, 1996). A gradient-descent learning law for a SAM parameter ξ has the form

$$\xi(t + 1) = \xi(t) - \mu_t \frac{\partial E}{\partial \xi} \tag{49}$$

where μ_t is a learning rate at iteration t. We seek to minimize the squared error

$$E(x) = \tfrac{1}{2}(f(x) - F(x))^2 \tag{50}$$

of the function approximation. Let ξ_j^k denote the kth parameter in the set function a_j. Then the chain rule gives the gradient of the error function with respect to ξ_j^k, with respect to the then-part set centroid c_j, and with respect to the then-part set volume V_j:

$$\frac{\partial E}{\partial \xi_j^k} = \frac{\partial F}{\partial F}\frac{\partial F}{\partial a_j}\frac{\partial a_j}{\partial \xi_j^k}, \frac{\partial E}{\partial c_j} = \frac{\partial E}{\partial F}\frac{\partial F}{\partial c_j}, \quad \text{and} \quad \frac{\partial E}{\partial V_j} = \frac{\partial E}{\partial F}\frac{\partial F}{\partial V_j} \tag{51}$$

where

$$\frac{\partial E}{\partial F} = -(f(x) - F(x)) = -\epsilon(x) \tag{52}$$

$$\frac{\partial F}{\partial a_j} = \frac{\left(\sum_{i=1}^{m} w_i a_i(x) V_i\right)(w_j V_j c_j) - w_j V_j \left(\sum_{i=1}^{m} w_i a_i(x) V_i c_i\right)}{\left(\sum_{i=1}^{m} w_i a_i(x) V_i\right)^2} \tag{53}$$

$$= \frac{[c_j - F(x)] w_j V_j}{\sum_{i=1}^{m} w_i a_i(x) V_i} = [c_j - F(x)] \frac{p_j(x)}{a_j(x)}. \tag{54}$$

The SAM ratio (2) gives (Kosko, 1995a, 1996)

$$\frac{\partial F}{\partial c_j} = \frac{w_j a_j(x) V_j}{\sum_{i=1}^{m} w_i a_i(x) V_i} = p_j(x) \tag{55}$$

and

$$\frac{\partial F}{\partial V_j} = \frac{w_j a_j(x)[c_j - F(x)]}{\sum_{i=1}^{m} w_i a_i(x) V_i} = \frac{p_j(x)}{V_j}[c_j - F(x)]. \tag{56}$$

Then the learning laws for the centroid and volume have the final form

$$c_j(t + 1) = c_j(t) + \mu_t \epsilon(x) p_j(x) \tag{57}$$

and

$$V_j(t + 1) = V_j(t) + \mu_t \epsilon(x) \frac{p_j(x)}{V_j}[c_j - F(x)]. \tag{58}$$

Learning laws for set parameters depend on how we define the set functions. The scalar Laplace set function

has the form $a_j(x) = \exp\{-|(x - m_j)/d_j|\}$. The partial deviatives of the set function with respect to its two parameters m_j and d_j have the form

$$\frac{\partial a_j}{\partial m_j} = \text{sign}\,(x - m_j)\,\frac{1}{|d_j|}\;a_j(x) \tag{59}$$

$$\frac{\partial a_j}{\partial m_j} = \text{sign}\,(d_j)\,\frac{|x - m_j|}{|d_j^2|}\;a_j(x) \tag{60}$$

where we define the sign function as

$$\text{sign}\,(x) = \begin{cases} 1 \text{ if } x > 0 \\ -1 \text{ if } x < 0 \\ 0 \text{ if } x = 0 \end{cases}. \tag{61}$$

Substitute (59)–(60) in (51) and in (49) to obtain the learning laws

$$m_j(t+1) = m_j(t) + \mu_t \epsilon(x)[c_j - F(x)]p_j(x)\,\text{sign}\,(x - m_j)\,\frac{1}{|d_j|} \tag{62}$$

$$d_j(t+1) = d_j(t) + \mu_t \epsilon(x)[c_j - F(x)]p_j(x)\,\text{sign}\,(d_j)\,\frac{|x - m_j|}{d_j^2}. \tag{63}$$

The partial derivatives for the scalar sinc set function $a_j(x) = \sin((x - m_j)/d_j)/((x - m_j)/d_j)$ have the form

$$\frac{\partial a_j}{\partial m_j} = \begin{cases} \left(a_j(x) - \cos\left(\left|\frac{x - m_j}{d_j}\right|\right)\right)\frac{1}{x - m_j} & \text{for } x \neq m_j \\ 0 & \text{for } x = m_j \end{cases} \tag{64}$$

$$\frac{\partial a_j}{\partial d_j} = \left(a_j(x) - \cos\left(\left|\frac{x - m_j}{d_j}\right|\right)\right)\frac{1}{d_j}. \tag{65}$$

So this scalar set function has the learning laws

$$m_j(t+1) = m_j(t) + \mu_t \epsilon(x)[c_j - F(x)]\,\frac{p_j(x)}{a_j(x)} \times \left(a_j(x) - \cos\left(\left|\frac{x - m_j}{d_j}\right|\right)\right)\frac{1}{x - m_j} \tag{66}$$

$$d_j(t-1) = d_j(t) + \mu_t \epsilon(x)[c_j - F(x)]\,\frac{p_j(x)}{a_j(x)} \times \left(a_j(x) - \cos\left(\left|\frac{x - m_j}{d_j}\right|\right)\right)\frac{1}{d_j}. \tag{67}$$

Like results hold for the learning laws of product n-D set functions. A factored set function $a_j(x) = a_j^1(x_1) \cdots a_j^n(x_n)$ leads to a new form for the error gradient. The gradient with respect to the parameter m_j^k of the jth set function a_j has the form

$$\frac{\partial E}{\partial m_j^k} = \frac{\partial E}{\partial F}\frac{\partial F}{\partial a_j}\frac{\partial a_j}{\partial a_j^k}\frac{\partial a_j^k}{\partial m_j^k} \quad \text{where } \frac{\partial a_j}{\partial a_j^k} = \prod_{i \neq k}^{n} a_j^i(x_i) = \frac{a_j(x)}{a_j^k(x_k)}. \tag{68}$$

We used product of the scalar sinc set function to define the if-part fuzzy set $A_j \subset R^n$ in the fuzzy profile approximator, and we used product of the scalar Laplace set function for the fuzzy equality measure. But we used the set SAM system instead of the simple point SAM. The learning laws follow from the structure of the set SAM.

We now derive learning laws for the set SAM. The chain-rule terms in (51) become

$$\frac{\partial E}{\partial \xi_j^k}(A) = \frac{\partial E}{\partial F}(A)\,\frac{\partial F}{\partial a_j}(A)\,\frac{\partial a_j}{\partial \xi_j^k}(A) \tag{69}$$

$$\frac{\partial E}{\partial c_j}(A) = \frac{\partial E}{\partial F}(A)\,\frac{\partial F}{\partial c_j}(A) \tag{70}$$

$$\frac{\partial E}{\partial V_j}(A) = \frac{\partial E}{\partial F}(A)\,\frac{\partial F}{\partial V_j}(A). \tag{71}$$

Then (52)–(56) give

$$\frac{\partial E}{\partial F}(A) = -(f(A) - F(A)) = -\epsilon(A) \tag{72}$$

$$\frac{\partial F}{\partial a_j}(A) = [c_j - F(A)]\,\frac{p_j(A)}{a_j(A)} \tag{73}$$

$$\frac{\partial F}{\partial c_j}(A) = p_j(A) \tag{74}$$

$$\frac{\partial F}{\partial V_j}(A) = \frac{p_j(A)}{V_j}\,[c_j - F(A)]. \tag{75}$$

The learning laws for then-part set centroids c_j and volumes V_j are

$$c_j(t+1) = c_j(t) + \mu_t \epsilon(A) p_j(A) \tag{76}$$

$$V_j(t+1) = V_j(t) + \mu_t \epsilon(A)\,\frac{p_j(A)}{V_j}\,[c_j - F(A)]. \tag{77}$$

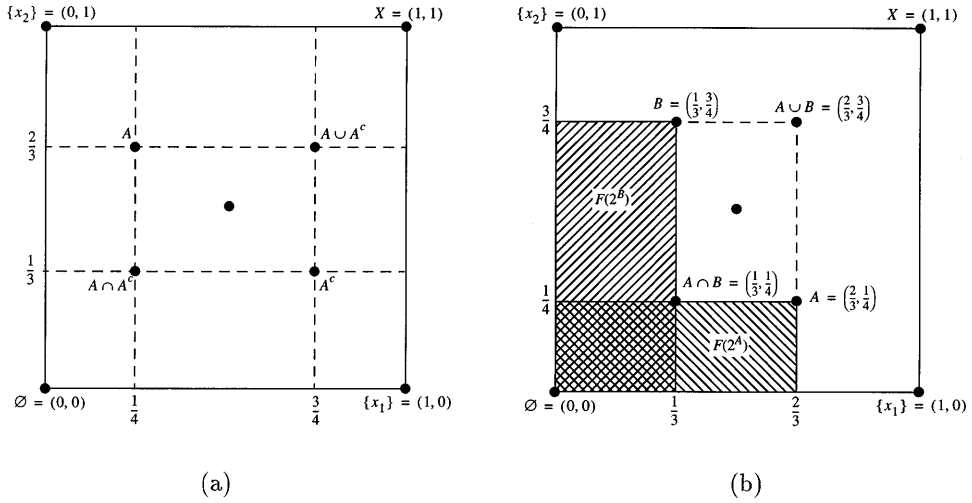


Figure 8. *Geometry of discrete fuzzy sets. Sets as points in a unit hypercube or fuzzy cube. Fuzzy set* $A \subset X = \{x_1, \ldots, x_n\}$ *defines a point in the fuzzy cube* $[0, 1]^n$. *Here* $X = \{x_1, x_2\}$, $A = (\frac{2}{3}, \frac{1}{4})$, *and* $B = (\frac{1}{3}, \frac{3}{4})$. *We define fuzzy-set intersection fitwise with pairwise minimum, union with pairwise maximum, and complementation with order reversal* ($a^c(x) = 1 - a(x)$). $F(2^A)$ *and* $F(2^B)$ *define the fuzzy power sets or the sets of all fuzzy subsets of* A *and* B. *Each set* $C \subset X$ *is a subset of* A *to some degree and so* C *belongs to* $F(2^A)$ *to some degree.* C *is a 100% subset of* A *if and only if* $c(x) \leq a(x)$ *for all* $x \in X$. *Then* $C \in F(2^A)$, *and so the set point* C *lies on or inside the hyper-rectangle* $F(2^A)$. *Partial subsets lie outside* $F(2^A)$.

But the partial derivative of the jth set function with respsect to its parameters ξ_j^k has the new form

$$\frac{\partial a_j}{\partial \xi_j^k}(A) = \frac{\partial}{\partial \xi_j^k} \int a_j(x) a(x)\, dx \tag{78}$$

$$= \begin{cases} \displaystyle\int \frac{\partial a_j}{\partial \xi_j^k}(x) a(x)\, dx & \text{continuous case} \\ \displaystyle\sum \frac{\partial a_j}{\partial \xi_j^k}(x) a(x) & \text{discrete case} \end{cases} \tag{79}$$

Then we substitute these partial derivatives into (49) to obtain the set-SAM learning rules.

Appendix C. Sets as Points: The Geometry of Discrete Fuzzy Sets

This appendix reviews the unit-cube geometry of discrete fuzzy system and derives the new adaptive equality measure. Let X be a set of n elements: $X = \{x_1, \ldots, x_n\}$. Any subset $A \subset X$ defines a point in the n-D unit hypercube I^n unit hypercube $I^n = [0, 1]^n$. The set of all fuzzy subsets of X or $F(2^X)$ fill in the cube. So the ordinary power set 2^X or the set of all 2^n subsets of X equals the Boolean n-cube B^n: $2^X = B^n$. Fuzzy subsets $A \subset X$ define the points *inside* or on the n-D unit hypercube (Kosko, 1991, 1996) as in Figure 8. A set $A \subset X$ is fuzzy when the ''laws'' of noncontradiction and excluded middle do not hold: $A \cap A^c \neq \emptyset$ and $A \cup A^c \neq X$.

Figure 8 shows an example when $X = \{x_1, x_2\}$. Then there are four binary subsets of X : $2^X = \{\emptyset, \{x_1\}, \{x_2\}, \{x_1, x_2\}\}$. The space $X = \{x_1, x_2\}$ lies at (1, 1). The empty set $\emptyset$ lies at the origin (0, 0) and the other two (standard) subsets $\{x_1\}$ and $\{x_2\}$ are at (1, 0) and (0, 1). A fuzzy subset $A \subset X$ defines the *f*uzzy un*it* or *fit* vectors $A = (a_1, a_2) \in I^2$ for $a_1, a_2 \in [0, 1]$. Figure 8a shows an example of a fuzzy set A. The geometrical view reveals the 2^n-fold symmetry of the set A and

its set operation products with respect to the midpoint. The midpoint is the maximal fuzzy set. It alone obeys $A = A^c$. The midpoint alone has spherical symmetry and lies equidistant to all 2^n cube vertices.

Figure 8b shows the 2-D cube with the fuzzy sets $A = (\frac{2}{3}, \frac{1}{4})$ and $B = (\frac{1}{3}, \frac{3}{4})$. We can define fuzzy-set intersection fitwise with pairwise minimum, union with pairwise maximum, and complementation with order reversal:

$$a \cap b(x) = \min\,(a(x), b(x)) \tag{80}$$

$$a \cup b(x) = \max\,(a(x), b(x)) \tag{81}$$

$$a^c(x) = 1 - a(x) \tag{82}$$

The subsethood theorem (Kosko, 1991) measures the degree to which a set A contains in a set B and does so in a simple ratio of cardinalities:

$$S(A, B) = \text{Degree}\,(A \subset B) = \frac{c(A \cap B)}{c(A)} \tag{83}$$

where c is a *counting* or *cardinality* (Kosko, 1991) measure

$$c(A) = \sum_{x_i \in X} a(x_i) \quad \text{or} \quad c(A) = \int_X a(x)\,dx \tag{84}$$

for integrable fuzzy set function $a : X \rightarrow [0, 1]$. This positive measure stems from the geometric interpretation of the fuzzy power sets $F(2^A)$ and $F(2^B)$ (Kosko, 1991, 1996). The subsethood measure extends the histogram intersection in (23). The subsethoods need not be symmetric: $S(A, B) \neq S(B, A)$. So we use a new symmetric measure (Kosko, 1996) of fuzzy equality as in (11):

$$E(A, B) = \text{Degree}(A = B) = \frac{c(A \cap B)}{c(A \cup B)} \tag{85}$$

$$= \frac{S(A, B)S(B, A)}{S(A, B) + S(B, A) - S(A, B)S(B, A)}\,. \tag{86}$$

Then we use the identities $\min\,(a, b) = \frac{1}{2}(a + b - |a - b|)$ and $\max\,(a, b) = \frac{1}{2}(a + b + |a - b|)$ to derive (90):

$$E(A, B) = \frac{c(A \cap B)}{c(A \cup B)} = \frac{\int \min\,(a(x), b(x))\,dx}{\int \max\,(a(x), b(x))\,dx} \tag{87}$$

$$= \frac{\int a(x) + b(x) - |a(x) - b(x)|\,dx}{\int a(x) + b(x) + |a(x) - b(x)|\,dx} \tag{88}$$

$$= \frac{1 - \dfrac{\int |a(x) - b(x)|dx}{\int a(x) + b(x)dx}}{1 + \dfrac{\int |a(x) - b(x)|dx}{\int a(x) + b(x)dx}} \tag{89}$$

$$= \frac{1 - \bar{d}(A, B)}{1 + \bar{d}(A, B)} \tag{90}$$

where fuzzy set $A \subset R^n$ has set function $a : R^n \rightarrow [0, 1]$ and $B \subset R^n$ has set function $b : R^n \rightarrow [0, 1]$ and

$$\|A - B\| = \int |a(x) - b(x)|dx \tag{91}$$

$$\|A + B\| = \int |a(x) + b(x)|dx \tag{92}$$

$$\text{and} \quad \bar{d}(A, B) = \frac{\|A - B\|}{\|A + B\|} = \frac{\int |a(x) - b(x)|dx}{\int |a(x) + b(x)|dx}\,. \tag{93}$$

Sums can replace the integrals in the discrete case.

We next derive a supervised learning law to tune the parameters of the set functions. Square error for a desired matching value D has the form $E = \frac{1}{2}(D - E)^2$. The chain rule gives the derivative of the squared error with respect to the kth parameter of the jth set function m_j^k as

$$\frac{\partial E}{\partial m_j^k} = \frac{\partial E}{\partial E}\frac{\partial E}{\delta \bar{d}}\frac{\partial \bar{d}}{\partial m_j^k}\,. \tag{94}$$

The derivatives have the form

$$\frac{\partial E}{\partial E} = -[D(A, B) - E(A, B)] \tag{95}$$

$$\frac{\partial E}{\partial \overline{d}} = -\frac{1 + E(A, B)}{1 + \overline{d}(A, B)} \tag{96}$$

$$\frac{\partial \overline{d}}{\partial m_j^k} = \frac{1}{\|A + B\|} \times \left(\frac{\partial}{\partial m_j^k} \|A - B\| - \overline{d}(A, B) \frac{\partial}{\partial m_j^k} \|A + B\| \right). \tag{97}$$

We now derive the derivatives of the ''norms'' $\|A - B\|$ and $\|A + B\|$ for the discrete sets A and B with respect to the parameter m_j^k in our image matching problems. The result follows from equations (16)–(17) and

$$\overline{d}(A, B) = \frac{\|A - B\|}{\|A + B\|} = \frac{\sum_{i=1}^{N} |A(\overline{x}_i) - B(\overline{x}_i)|}{\sum_{i=1}^{N} |A(\overline{x}_i) + B(\overline{x}_i)|} \tag{98}$$

and the assumption that each set has its own independent parameters (so $\partial a_i / \partial m_j^k = 0$ for $i \neq j$):

$$\frac{\partial}{\partial m_j^k} \|A - B\|$$

$$= \frac{\partial}{\partial m_j^k} \sum_{i=1}^{N} |A(\overline{x}_i) - B(\overline{x}_i)| \tag{99}$$

$$= \sum_{i=1}^{N} \operatorname{sign}(A(\overline{x}_i) - B(\overline{x}_i)) \frac{\partial}{\partial m_j^k} (A(\overline{x}_i) - B(\overline{x}_i)) \tag{100}$$

$$= \sum_{i=1}^{N} \left[\operatorname{sign}(A(\overline{x}_i) - B(\overline{x}_i)) \frac{\partial}{\partial m_j^k} \left(\sum_{l=1}^{m} A_l a_l(\overline{x}_i) - \sum_{l=1}^{m} B_l a_l(\overline{x}_i) \right) \right] \tag{101}$$

$$= \sum_{i=1}^{N} \left[\operatorname{sign}(A(\overline{x}_i) - B(\overline{x}_i)) \frac{\partial}{\partial m_j^k} \left(\sum_{l=1}^{m} a_l(\overline{x}_i)(a_l(T_A) - a_l(T_B)) \right) \right] \tag{102}$$

$$= \sum_{i=1}^{N} \left[\operatorname{sign}(A(\overline{x}_i) - B(\overline{x}_i)) \sum_{l=1}^{m} \left(\frac{\partial}{\partial m_j^k} a_l(\overline{x}_i)(a_l(T_A) - a_l(T_B)) \right) \right] \tag{103}$$

$$= \sum_{i=1}^{N} \operatorname{sign}(A(\overline{x}_i) - B(\overline{x}_i)) \left[[a_j(T_A) - a_j(T_B)] \frac{\partial a_j(\overline{x}_i)}{\partial m_j^k} + a_j(\overline{x}_i) \left(\frac{\partial a_j(T_A)}{\partial m_j^k} - \frac{\partial a_j(T_B)}{\partial m_j^k} \right) \right] \tag{104}$$

$$= [a_j(T_A) - a_j(T_B)] \sum_{i=1}^{N} \operatorname{sign}(A(\overline{x}_i) - B(\overline{x}_i)) \frac{\partial a_j(\overline{x}_i)}{\partial m_j^k} + \left(\frac{\partial a_j(T_A)}{\partial m_j^k} - \frac{\partial a_j(T_B)}{\partial m_j^k} \right) \sum_{i=1}^{N} \operatorname{sign}(A(\overline{x}_i) - B(\overline{x}_i)) a_j(x_i) \tag{105}$$

The derivation proceeds in like manner for $\partial \|A + B\| / \partial m_j^k$ as

$$\frac{\partial}{\partial m_j^k} \|A + B\| = [a_j(T_A) + a_j(T_B)] \sum_{i=1}^{N} \frac{\partial a_j(\overline{x}_i)}{\partial m_j^k} + \left(\frac{\partial a_j(T_A)}{\partial m_j^k} + \frac{\partial a_j(T_B)}{\partial m_j^k} \right) \sum_{i=1}^{N} a_j(\overline{x}_i) \tag{106}$$

since $a(x) \geq 0$ for all $x \in X$. The condition $a_j(T_A) = \Sigma_{i=1}^{N} t_A^i a_j(\overline{x}_i)$ and $a_j(T_B) = \Sigma_{i=1}^{N} t_B^i a_j(\overline{x}_i)$ from (10) gives

$$\frac{\partial a_j}{\partial m_j^k}(T_A) = \sum_{i=1}^{N} t_A^i \frac{\partial a_j}{\partial m_j^k}(\overline{x}_i) \tag{107}$$

$$\frac{\partial a_j}{\partial m_j^k}(T_B) = \sum_{i=1}^{N} t_B^i \frac{\partial a_j}{\partial m_j^k}(\overline{x}_i). \tag{108}$$

Appendix B derives the partial derivatives of the Laplace set function a_j with respect to its two parameters in equations (59)–(60). Then substitute (107)–(108) into (105)–(106) to obtain (97) and substitute (95)–(97) to

obtain (94) and the learning law for each parameter in the form of (49):

$$m_j^k(t+1) = m_j^k(t) - \mu_t(D(A,B) - E(A,B))\frac{1+E(A,B)}{1-\overline{d}(A,B)}\frac{1}{\|A+B\|}\Bigg[(a_j(T_A) - a_j(T_B))\sum_{i=1}^{N}\operatorname{sign}(A(\overline{x}_i) - B(\overline{x}_i))\operatorname{sign}(\overline{x}_i - m_j^k)\frac{1}{|d_j^k|}a_j(\overline{x}_i) + \left(\sum_{=1}^{N}(t_A^i - t_B^i)\operatorname{sign}(\overline{x}_i - m_j^k)\frac{1}{|d_j^k|}a_j(\overline{x}_i)\right)\sum_{i=1}^{N}\operatorname{sign}(A(\overline{x}_i) - B(\overline{x}_i))a_j(\overline{x}_i) - \overline{d}(A,B)\left((a_j(T_A) - a_j(T_B))\sum_{i=1}^{N}\operatorname{sign}(\overline{x}_i - m_j^k)\frac{1}{|d_j^k|}a_j(\overline{x}_i) + \left(\sum_{i=1}^{N}(t_A^i - t_B^i)\operatorname{sign}(\overline{x}_i - m_j^k)\frac{1}{|d_j^k|}a_j(\overline{x}_i)\right)\sum_{i=1}^{N}a_j(\overline{x}_i)\right)\Bigg] \tag{109}$$

$$d_j^k(t+1) = d_j^k(t) - \mu_t(D(A,B) - E(A,B))\frac{1+E(A,B)}{1-\overline{d}(A,B)}\frac{1}{\|A+B\|}\Bigg[(a_j(T_A) - a_j(T_B))\sum_{i=1}^{N}\operatorname{sign}(A(\overline{x}_i) - B(\overline{x}_i))\operatorname{sign}(d_j^k)\frac{|\overline{x}_i - m_j^k|}{|d_j^k|^2}a_j(\overline{x}_i) + \left(\sum_{i=1}^{N}(t_A^i - t_B^i)\operatorname{sign}(d_j^k)\frac{|\overline{x}_i - m_j^k|}{|d_j^k|^2}a_j(\overline{x}_i)\right)\sum_{i=1}^{N}\operatorname{sign}(A(\overline{x}_i) - B(\overline{x}_i))a_j(\overline{x}_i) - \overline{d}(A,B)\left((a_j(T_A) - a_j(T_B))\sum_{i=1}^{N}\operatorname{sign}(d_j^k)\frac{|\overline{x}_i - m_j^k|}{|d_j^k|^2}a_j(\overline{x}_i) + \left(\sum_{i=1}^{N}(t_A^i - t_B^i)\operatorname{sign}(d_j^k)\frac{|\overline{x}_i - m_j^k|}{|d_j^k|^2}a_j(\overline{x}_i)\right)\sum_{i=1}^{N}a_j(\overline{x}_i)\right)\Bigg]. \tag{110}$$