models [19], [27]. Specifically, we consider the model presented in the text; it is easy to see that, for this model, an *LO* policy is suboptimal. We provide a counterexample to show this; this counterexample may be adapted to show that *LO* policies are suboptimal for other popular Markov state models (*generalized-LS* model and Young's model [23]). We assume that $p_1 = p_2 = p_3 = p_4 = 1$ and $p_5 = 0$.

We assume that there are two associations to be learned, A_1 and A_2, that currently reside in states U and C, respectively, and that three trials are remaining for instruction to complete. An *LO* policy would prescribe that A_2 be presented on the next trial since this would yield the maximum expected average retention (0.5). Independently of which associations are selected for presentation in the two remaining trials, it is easy to see that the expected average long-term retention of implementing an *LO* policy equals 0.5 (at the end of instruction, A_1 will have a probability of unity of residing in state L, and A_2 will have a zero probability of residing in state L). This value is suboptimal since a policy that presents first A_1 then A_2, and finally A_1 yields an expected average retention of 1.0; the (certain) joint-state trajectory for the two associations is then $UC \rightarrow SC \rightarrow CL \rightarrow LL$. In general, the instructor may maximize retention by taking advantage of these state transitions occurring during nonpresentation trials that do not lead to a state denoting less durable representations in memory transitions (here, from state S to state C, and from state C to itself). Such transitions occur in all Markov state models that can account for effects of lag on accuracy[25].

REFERENCES

[1] G. J. Groen and R. C. Atkinson, "Models for optimizing the learning process," *Psychol. Bull.*, vol. 66, pp. 309–320, 1966.

[2] K. V. Katsikopoulos, "Optimal instructional policies based on a random-trial incremental model of learning," *IEEE Trans. Syst., Man, Cybern. A*, vol. 30, pp. 490–494, July 2000.

[3] R. D. Bellman, *Dynamic Programming*. Princeton, NJ: Princeton Univ. Press, 1957.

[4] R. R. Bush and F. Mosteller, "A mathematical model for simple learning," *Psychol. Rev.*, vol. 58, pp. 313–323, 1951.

[5] G. H. Bower, "Application of a model to paired-associate learning," *Psychometrika*, vol. 26, pp. 255–280, 1961.

[6] R. D. Smallwood, "The analysis of economic teaching strategies for a simple learning model," *J. Math. Psychol.*, vol. 8, pp. 285–301, 1971.

[7] W. Karush and R. E. Dear, "Optimal stimulus presentation strategy for a stimulus sampling model of learning," *J. Math. Psychol.*, vol. 3, pp. 19–47, 1966.

[8] R. C. Atkinson and J. A. Paulson, "An approach to the psychology of instruction," *Psychol. Bull.*, vol. 78, pp. 49–61, 1972.

[9] M. F. Norman, "Incremental learning on random trials," *J. Math. Psychol.*, vol. 1, pp. 336–350, 1964.

[10] R. C. Calfee, "The role of mathematical models in optimizing instruction," *Scientia: Rev. Int. Synthese Sci.*, vol. 105, pp. 1–25, 1970.

[11] D. L. Fisher, R. A. Wisher, and T. Ranney, "Optimal static and dynamic training schedules: State models of skill acquisition," *J. Math. Psychol.*, vol. 40, pp. 30–47, 1996.

[12] R. E. Dear, H. F. Silberman, D. P. Estavan, and R. C. Atkinson, "An optimal strategy for the presentation of paired-associate items," *Behav. Sci.*, vol. 12, pp. 1–13, 1967.

[13] R. C. Atkinson, "Optimizing the learning of a second-language vocabulary," *J. Exper. Psychol.*, vol. 96, pp. 124–129, 1972.

[14] P. J. Lorton, "Computer-based instruction in spelling: An investigation of optimal strategies for presenting material," Ph.D. dissertation, Stanford Univ., Stanford, CA, 1972.

[15] J. H. Laubsch, "Optimal item allocation in computer-assisted instruction," *IAG J.*, vol. 3, pp. 295–311, 1972.

[16] R. G. Crowder, *Principles of Learning and Memory*. Hillsdale, NJ: Lawrence Erlbaum, 1976.

[17] F. N. Dempster, "The spacing effect: A case study in the failure to apply the results of psychological research," *Amer. Psychol.*, vol. 43, pp. 627–634, 1988.

[18] R. C. Atkinson and E. J. Crothers, "A comparison of paired-associate learning models having different acquisition and retention axioms," *J. Math. Psychol.*, vol. 1, pp. 285–315, 1964.

[19] K. V. Katsikopoulos and D. L. Fisher, "Formal requirements of Markov state models for paired associate learning," *J. Math. Psychol.*, to be published.

[20] D. A. Balota, J. M. Duchek, and R. Paullin, "Age-related differences in the impact of spacing, lag and retention interval," *Psychol. Aging*, vol. 4, no. 1, pp. 3–9, 1989.

[21] D. H. Spieler and D. A. Balota, "Characteristics of associative learning in younger and older adults: Evidence from an episodic priming paradigm," *Psychol. Aging*, vol. 11, no. 4, pp. 607–620, 1996.

[22] K. V. Katsikopoulos, D. L. Fisher, and S. A. Duffy, "Spacing effects: Evidence for rehearsal based accounts and for differential impact of aging across cognitive processes," , submitted for publication.

[23] D. H. Kausler, Ed., *Learning and Aging in Normal Memory*. New York: Academic, 1994.

[24] K. V. Katsikopoulos, "Characterizing and Optimizing the Performance of Younger and Older Adults in Paired Associate Tasks: A Markov Modeling Approach," Ph.D dissertation, Univ. Mass., Amherst, MA, 1999.

[25] J. L. Young, "Reinforcement-test intervals in paired associate learning," *J. Math. Psychol.*, vol. 8, pp. 58–81, 1971.

[26] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998.

[27] T. D. Wickens, *Models for Behavior: Stochastic Processes in Psychology*. San Fransisco, CA: Freeman, 1982.

Some Classification Algorithms Integrating Dempster–Shafer Theory of Evidence with the Rank Nearest Neighbor Rules

Nikhil R. Pal and Swati Ghosh

***Abstract*—We propose five different ways of integrating Dempster–Shafer theory of evidence and the rank nearest neighbor classification rules with a view to exploiting the benefits of both. These algorithms have been tested on both real and synthetic data sets and compared with the k-NN, m-MRNN, and k-NNDST, which is an algorithm that also combines Dempster–Shafer theory with the k-NN rule. If different features have widely different variances then the distance-based classifier, algorithms like k-NN and k-NNDST may not perform well, but in this case the proposed algorithms are expected to perform better. Our simulation results indeed reveal this. Moreover, the proposed algorithms are found to exhibit significant improvement over the m-MRNN rule.**

***Index Terms*—Nearest neighbor classifier, rank nearest neighbor, theory of evidence.**

I. INTRODUCTION

Pattern classification by distance functions is one of the earliest concept in automatic pattern recognition. The k-Nearest Neighbor (k-NN) rule is one of the most widely used pattern classification techniques proposed by Fix and Hodges [1]. This method usually leads to satisfactory results when the pattern classes exhibit clustering tendencies. Cover and Hart [2] showed that under certain conditions k-NN method approaches

Manuscript received November 19, 1997. This paper was recommended by Associate Editor C. C. White.

N. R. Pal is with the Electronics and Communication Sciences Unit, Indian Statistical Institute, Calcutta 700035, India.

S. Ghosh is with Advanced Engineering Sciences, ITT Industries, Inc., Washington, DC 20024 USA.

Publisher Item Identifier S 1083-4427(01)00859-1.

the optimal Bayes error rate. Given a sample point Dudani [3] proposed a method to assign a weight $w^{(i)}$ to the ith nearest neighbor $x^{(i)}$ as: $w^{(i)} = (d^{(k)} - d^{(i)})/(d^{(k)} - d^{(1)})$, $d^{(k)} \neq d^{(1)}$ and $w^{(i)} = 1$ when $d^{(k)} = d^{(1)}$. Here $d^{(1)}, \cdots, d^{(k)}$ are the distances of the k nearest neighbors from the point x arranged in increasing order. The unknown pattern x is assigned to the class in which the sum of weights, among the k-nearest neighbors, is the maximum. However, some authors claim that under certain conditions unweighted k-NN rule performs better than any weighted k-NN rule [4]. Instead of using Euclidean distance in classification Anderson [5] proposed a nonparametric classification rule for two univariate populations by ranking the training samples. Bagui [6], [7] extended the idea of Anderson to $s > 2$ populations. This univariate rank nearest neighbor (URNN) rule is then further extended by Bagui and Pal [8] to multivariate data resulting in the multivariate rank nearest neighbor (MRNN) rule. Denoeux [4] proposed a new classification procedure using the k-nearest neighbors and Dempster–Shafer (D–S) theory of evidence to get the k-NNDST rule [9].

In this paper we propose several algorithms for pattern classification that combine the underlying philosophy of RNN rule with D-S theory. We have tested our algorithms on several real and synthetic data sets and compared their performances with the ordinary k-NN, m-MRNN and the k-NNDST algorithms, and we have obtained encouraging results.

II. Nearest Neighbor Rules

We begin with a description of the 1-stage univariate rank nearest neighbor (1-URNN) rule.

The 1-URNN Algorithm [6], [7]

1) Let $\{x_{ij}^s\}$ $(i = 1 \cdots s; j = 1 \cdots n_i)$ be the training data, s be the number of classes, n_i be the number of training data from class i, and z be an incoming sample to be classified.
2) Sort $\{x_{ij}^s\}$ in ascending order to $\{\hat{x}_l,\ l = 1, 2 \cdots N; N = \sum_{i=1}^{s} n_i\}$.
3) If $z \in [\hat{x}_1, \hat{x}_N]$ then goto 5.
4) If z is either the smallest ($z \leq \hat{x}_1$) or the largest ($z \geq \hat{x}_N$) observation then classify z into the population of its immediate rank nearest neighbor and exit.
5) If immediate left-hand (LH) and right-hand (RH) neighbors of z both belong to the same population then classify z to that population and exit.
6) If the immediate left-hand (LH) and right-hand (RH) rank nearest neighbors of z belong to different populations, classify z into either population arbitrarily.

The asymptotic error rate of the 1-URNN rule for s populations is the same as that of 1-Nearest Neighbor (1-NN) of Cover and Hart [2] for s populations. Next we present the m-URNN, a multi-stage generalization of the 1-URNN rule with s populations.

The m-URNN Algorithm [8]

1) Sort training data $\{x_{ij}^s\}$ in ascending order to $\{\hat{x}_l, l = 1, 2, \cdots, N; N = \sum_{i=1}^{s} n_i\}$.
2) Fix $m \in N$ a positive integer given $z \in R$.
3) If $z \notin [\hat{x}_1, \hat{x}_N]$, classify z with its rank nearest neighbor, exit.
4) If $z = \hat{x}_l$ for some l, classify z with the label of $\hat{x}_l$, exit.
5) $j \leftarrow 1$.
6) While $(j \leq m)$

If left-hand and right-hand neighbors of z belong to the same population then classify z to that population, exit.

If $j = m$, classify z into either population arbitrarily, exit.

$j \leftarrow j - 1$.

Wend (end of while).

Bagui and Pal [8] extended this m-URNN to the m-stage Multivariate Rank Nearest Neighbor (m-MRNN) rule which classifies multivariate observations using the m-URNN rule first on each feature and then combines these feature-wise results to get the final decision for each multivariate observation. The schematic description of the Algorithm for m-MRNN is presented next.

The m-MRNN Algorithm

1) Let there be s p variate ($p \geq 1$) populations $(w_1, \cdots, w_s)$ and let $\{x_1^{(i)}, \cdots, x_n^{(i)}\} \subset R^p$ be the training data from population w_i.
2) Let $z \in R^p$ be the unknown observation to be classified.
3) Classify $z_k, k = 1, 2, \cdots p$ by applying m-URNN rule.
4) Let us define $\phi_{lk}^{(m)}$ $(l = 1, \cdots, s)$ $(k = 1, \cdots, p)$ to be 1 or 1/2 or 0 when z_k is classified to the lth population or randomized between the lth and jth $(l \neq j)$ populations and not classified to l, respectively.
5) Define $m_l = \sum_k \phi_{lk}^{(m)}$, m_l is the sum over all features of class l.
6a) If $m_i = m^*$, where m^* is the unique maximum of $\{m_l: 1 \leq l \leq s\}$, then classify z to population w_i.
6b) If $m^* = m_{i_1} = m_{i_2} = \cdots = m_{i_J}$, then classify z to w_i with probability $1/J$ for $i = i_1, i_2, \cdots, i_J$.

We conclude this section with a description of the well known k-NN rule.

The k-NN Algorithm

Let us consider a set of patterns $X = \{x_1, \cdots, x_N\} \subseteq R^p$ of known classification where each pattern belongs to one of the classes $W = \{w_1, w_2, \cdots, w_s\}$. The nearest neighbor (NN) classification rule assigns a pattern z of unknown classification to the class of its nearest neighbor, where $x_i \in X$ is the nearest neighbor to z if

$$D(x_i, z) = \underbrace{min}_{l}\{D(x_l, z) \qquad l = 1, 2, \cdots, N\},$$

D is the Euclidean distance between two patterns in R^p. This scheme is called the 1-NN rule since it classifies a pattern based on only one neighbor of z. The k-NN rule considers the k-nearest neighbors of z and uses the majority rule. Let t_l, $l = 1, 2, \cdots, s$ be the number of neighbors from class l in the k-nearest neighbors of z, $\sum_{l=1}^{s} t_l = k$. Then z is assigned to class j if $t_j = \underbrace{Max}_{l}\{t_l\}$.

III. Dempster–Shafer Theory of Evidence

Let X be the universal set and P(X) be its power set. Any function $g: P(X) \rightarrow [0, 1]$ is a fuzzy measure if it satisfies the following three axioms [9]:

- $g1$: $g(\phi) = 0$ and $g(X) = 1$.
- $g2$: For every $A, B \in P(X)$, if $S \subset B$ then $g(A) \leq g(B)$.
- $g3$: For every sequence $(A_i \in P(X) | i = 1, 2, \cdots)$ of subsets of X, if either $A_1 \subseteq A_2 \subseteq \cdots$ or $A_1 \supseteq A_2 \supseteq \cdots$, then $\lim_{i\to\infty} g(A_i) = g(\lim_{i\to\infty} A_i)$.

$g3$ is applicable only for infinite universe and in the present context since X is finite, $g3$ can be disregarded. Two important and well developed special types of fuzzy measures are belief and plausibility.

A belief measure is a function $Bel: P(X) \rightarrow [0, 1]$ that satisfies the axioms $g1$ through $g3$ of fuzzy measures and the following additional axiom:

$$Bel(A_1 \cup A_2 \cup \cdots \cup A_n) \geq \sum_i Bel(A_i) - \sum_{i<j} Bel(A_i \cap A_j) - \cdots + (-1)^{n-1} Bel(A_1 \cap \cdots \cap A_n)$$

for every n and for every collection of subsets of X.

There is a plausibility measure with each belief measure defined by $Pl(A) = 1 - Bel(A^c) \forall A \in P(X)$.

Every belief measure and its dual plausibility measure can be expressed in terms of a Basic Probability Assignment (BPA) function m. $m : P(X) \rightarrow [0,1]$ is called a BPA whenever $m(\phi) = 0$ and $\sum_{A \subset X} m(A) = 1$. Here $m(A)$ is interpreted as the degree of evidence supporting the claim that the "truth" is in A and in absence of further evidence no more specific statement can be made. A belief measure and a plausibility measure are uniquely determined by m through the formulas

$$Bel(A) = \sum_{B \subseteq A} m(B) \tag{1}$$

$$Pl(A) = \sum_{B \cap A \neq \phi} m(B) \quad \forall A \subset X. \tag{2}$$

From (1) and (2) we see that, $Pl(A) \geq Bel(A) \forall A \in P(X)$. Every set $A \in P(X)$ for which $m(A) > 0$ is called a focal element of m. Evidence obtained in the same context from two distinct sources and expressed by two BPAs m_1 and m_2 on some power set $P(X)$ can be combined by Dempster's rule of combination to obtain a joint basic assignment $m_{1,2}$ as

$$m_{1,2}(A) = \begin{cases} \dfrac{\sum_{B \cap C = A} m_1(B) m_2(C)}{1 - K}, & \text{if } A \neq \phi \\ 0, & \text{if } A = \phi \end{cases} \tag{3}$$

Here

$$K = \sum_{B \cap C = \phi} m_1(B) \cdot m_2(C).$$

Dempster's rule of combination is commutative and associative. Suppose m_1 and m_2 are two BPAs. If m_1 is vacuous, then m_1 and m_2 are combinable and $m_1 \oplus m_2 = m_2$. If m_1 is Bayesian (i.e., $m_1(A) = 0 \forall A$ such that $|A| > 1$) and m_1 and m_2 are combinable, then $m_1 \oplus m_2$ is Bayesian. A special type of BPA, $m(A) = s$ and $m(X) = 1 - s$ is called simple support function which we shall use extensively in our study. There are some criticism about how belief is treated in Dempster's framework and in this regard a transferable belief model [13], [14] is proposed which does not assume any probability measure on X. However, we do not pursue this model in this paper.

Now we shall present the k-NNDST [4] rule which integrates the voting feature of NN rule and evidence aggregation characteristic of D-S theory of evidence.

The k-NNDST Algorithm

Let X be a p dimensional training set (i.e., N-data points in R^p), $C = \{C_1, \cdots, C_s\}$ be the set of classes ,and z be an incoming sample to be classified based on X. Let Φ^k be the set of k-nearest neighbors of z in X according to the Euclidean distance measure. Any $x_i \in \Phi^k$ which has a class label q can be viewed as a piece of evidence suggesting that z could be a member of class q—it increases our belief that z could be a member of C_q but does not provide a 100% confidence. This thing can be modeled under the D-S framework by a BPA, m^i as

$$m^i(\{C_q\}) = \alpha_i \quad m^i(C) = 1 - \alpha_i \quad \text{and}$$
$$m^i(A) = 0 \quad \forall A \in \{P(C) - \{C, \{C_q\}\}\}$$
$$\text{where } 0 < \alpha_i < 1.$$

The choice of α_i is an important issue to be resolved. Denoeux [4] suggested $\alpha_i = \alpha_0 \Phi_q(d^i)$ where $\Phi_q(d) = e^{-\gamma_q d^\beta}$ with $\gamma_q > 0$ and $\beta \in \{1, 2, \cdots\}$.

Let $\Phi_q^k \subset \Phi^k$ be the set of elements of Φ^k from class q. Then for each $x_i \in \Phi_q^k$ we get a BPA m^i. Denoeux [4] combined all such BPAs to get

$$m_q(\{C_q\}) = 1 - \prod_{x^i \in \Phi_q^k} (1 - \alpha_i) \quad \text{and}$$
$$m_q(C) = \prod_{x^i \in \Phi_q^k} (1 - \alpha_i).$$

In this way, we can get at most s BPAs, m_q, $q = 1, 2, \cdots, s$. These BPAs are then combined to get the global BPA $m = \oplus_{q=1}^{s} m_q$ as

$$m(\{C_q\}) = \frac{m_q(\{C_q\}) \prod_{r \neq q} m_r(C)}{K}; \qquad q = 1, 2, \cdots, s$$

$$m(C) = \frac{\prod_{q=1}^{s} m_q(C)}{K}.$$

Here the normalizing factor K is

$$K = \sum_{q=1}^{s} m_q(\{C_q\}) \prod_{r \neq q} m_r(C) + \prod_{q=1}^{s} m_q(C).$$

The point z is then classified to class q^* such that $m(\{C_{q^*}\}) = \underbrace{Max}_{q} \{m(\{C_q\})\}$. Such a method is likely to produce a better performance than the ordinary k-NN rule. A better decision making strategy may be to use the pignistic probability distribution [13]–[15]. The class label can be assigned based on the maximum pignistic probability, which in the present context is the same as the maximum BPA decision. In the above method there are a few user defined parameters whose choice has significant impact on the performance of the classifier. Recently, Zouhal and Denoeux [16] suggested a method for estimation of these parameters minimizing an error function defined using the pignistic probability distribution and the actual label vector of the training data points.

IV. Proposed Algorithms

We propose five algorithms which integrate Dempster–Shafer theory with the concept of Rank Nearest Neighbor for multivariate s class problem ($s \geq 1$). There are two motivations behind this. First, each feature value of an unknown observation z (to be classified) as well as each of its k-nearest neighbors together provide some evidence about the class label of z. Hence, the aggregation of such evidences using Dempster's framework is expected to result in a good performance of the classifier. Second, it enables us to extend the m-URNN algorithm for dealing with high dimensional data. For the first four algorithms for each feature we find m rank nearest neighbors using the m-URNN rule and then for each feature we define a BPA. Thus we will have p BPAs for p features. These p BPAs are then combined by Dempster's rule of aggregation. The four algorithms maintain this basic architecture but differ in the definition of the feature-wise BPAs. In the fifth algorithm for each of the p features we first find the m rank nearest neighbors and define BPAs for all classes which have representatives in the m rank nearest neighbors. For each feature these class-wise BPAs are combined to get a single feature-wise BPA. The p such feature-wise BPAs are again combined using Dempster's rule of combination.

Our first algorithm, MRNNDST-1 classifies multivariate observations using the m-URNN rule first on each feature. Let $X = \cup_{j=1}^{s} X_j \subset R^p$ where X_j is the set of training data from the jth class and $x_i = (x_{i1}, x_{i2}, \cdots, x_{ip})^T \in X$. We sort the list of kth feature values x_{ik} for all data points ($i = 1, 2, \cdots, N$). Let us denote the list of N

values of the kth feature by X_L^k and its sorted version be $\hat{X}_L^k$. Now we insert z_k, the kth feature value of the incoming observation $z = (z_1, \cdots, z_k, \cdots, z_p)^t$, to the list $\hat{X}_L^k$ in its appropriate place, so that the augmented list remains sorted. Let c_i be the number of points from ith class out of the m left and right neighbors of z_k. If c_i is high then our confidence that z is in class i is high, but we cannot assume a 100% certainty. Thus, we can define a BPA for feature k as

$$m_k(\{I\}) = e^{P(c_I/2m)} \Big/ \sum_{i=1}^{s} e^{P(c_i/2m)} = \phi_{Ik} \quad \text{(say)}$$
$$I = 1, 2, \cdots s.$$

Here P is a user defined constant.

Note, $\sum_{I=1}^{s} m_k(\{I\}) = 1$ and $m_k(\phi) = 0$. That is m_k is a valid BPA. Next we combine these feature-wise BPAs using Dempster's rule to find the class label for the observations. Since it combines MRNN with DST, we name it MRNNDST-1, one indicates our first algorithm. A schematic description of the algorithm is given next.

Algorithm MRNNDST-1
Store:

$$X = \bigcup_{i=1}^{s} X_i, X_i = \text{set of data from class } i$$
$$= [x_{ij}]_{N \times p} = [X_L^1 \; X_L^2 | \cdots | X_L^p]$$
where $N = \sum_{i=1}^{s} n_i, |X_i| = n_i$
X_L^i is the list of values for
feature i for the N data points

Unknown observation $z = (z_1, \cdots, z_p)^T \in R^p$
Pick: $m \subset N^+$ and $P \subset R^+$
Process:
For $i = 1$ to s
  $c_i = 0$; /* counter for each class */
For $k = 1$ to p
  Sort X_L^k in ascending order to get $\hat{X}_L^k = \{\hat{x}_{1k}, l = 1, \cdots, N\}$
  If $z_k \notin [\hat{x}_{1k}, \hat{x}_{Nk}]$ and L is the class label of the rank nearest neighbor of z_k,
  OR, If $z_k = \hat{x}_{lk}$, and L is the class label of $\hat{x}_{lk}$
    Then $\phi_{Lk} = e^{P.1}$ and $\phi_{lk} = e^{P.0}, l = 1, \cdots, s, l \neq L$
  Else
    $r \leftarrow 1$;
    while $(r \leq m)$
      Find labels L_l and L_k of the rth LH and RH neighbors of z_k.
      If $L_l = i$ and $L_k = j$, add 1 to c_i and c_j
      $r \leftarrow r + 1$.
    wend (end of while).
/* Assignment of class label BPA */
  For $l = 1$ to s
    $\phi_{lk} = e^{P(c_l/2m)} / \sum_{j=1}^{s} e^{P(c_j/2m)}$
  Next k /*
  Combination of BPAs using D–S rule */
  For $l = 1$ to s

$$\phi_{lc} = \frac{\prod_{i=1}^{s} \phi_{li}}{K} \quad \text{where} \quad K = \sum_{l=1}^{s} \left(\prod_{i=1}^{s} \phi_{li} \right).$$

/* Computation of class for z */

$$M = \underbrace{ArgMax}_{l} \{\phi_{lc}\}$$

  Assign z to class M.

Note that here when z_k is to the left of $\hat{x}_{1k}$ or to the right of $\hat{x}_{Nk}$ or $z_k = \hat{x}_{tk}$ for some t, then we use only one RNN, as these are the cases which are less ambiguous.

Our second algorithm, MRNNDST-2 has more or less the same structure as MRNNDST-1. In this algorithm, we consider m-RNN instead of taking m-URNN. In m-URNN, when $z_k \leq \hat{x}_{1k}$ or $z_k \geq \hat{x}_{Nk}$ or $z_k = \hat{x}_{tk}$ the decision is made without considering all of m nearest neighbors of z_k because these are cases of less ambiguity. But in m-RNN, we use all m nearest neighbors of z_k in every case. In other words, when $z_k \leq \hat{x}_{1k}$, we consider the m right-hand side RNN; when $z_k \geq \hat{x}_{Nk}$, we use the m left-hand side RNN and in all other cases we use the m neighbors from each side, if available. The other major difference with MRNNDST-1 is that here the feature-wise BPAs are defined using distances of z_k from the m neighbors in X_L^k. MRNNDST-1 is primarily based on the majority rules like the k-NN but MRNNDST-2 is based on the average distances, i.e., the similarity of z_k to the points in Φ_k^i. Φ_k^i = set of m RNN of z_k. In this case the feature-wise BPAs are defined as

$$m_k(\{I\}) = e^{-(d_I/c_I)} \Big/ \sum_{i=1}^{s} e^{-(d_i/c_i)} = \phi_{Ik} \quad \text{(say)}$$
$$I = 1, 2, \cdots, s$$

where d_i = sum of the distances between z_k and the points of the ith class out of m nearest neighbor of z_k. Note that if $z_k \leq \hat{x}_{1k}$ the Φ_k^i contains only the m right-hand side RNN. Similarly, if $z_k \geq \hat{x}_{Nk}$ the Φ_k^i contains only the m left-hand side RNN. In all other cases m left-hand side and m right-hand side RNN are used.

Algorithm MRNNDST-2
Store:

$$X = \bigcup_{i=1}^{s} X_i, X_i = \text{set of data from class } i$$
$$= [x_{ij}]_{N \times p} = [X_L^1 | X_L^2 | \cdots | X_L^p]$$
where $N = \sum_{i=1}^{s} n_i, |X_i| = n_i$
X_L^i is the list of values for
feature i for the N data points

  Unknown observation $z = (z_1, \cdots, z_p)^T \in R^p$
Pick: $m \in N^+$
Process:
  For $i = 1$ to s
    $d_i = 0$; /* sum of distance of each class */
  For $k = 1$ to p
    Sort X_L^k in ascending order to get $\hat{X}_L^k = \{\hat{x}_{lk}, l = 1, \cdots, N\}$
      Let Φ_k^{il} = set of $\hat{x}_{jk}$ from class $l \subseteq \Phi_k^i$.
      $c_l = |\Phi_k^{il}|$ and $d_l = \sum_{\hat{x}_{jk} \subset \Phi_k^{il}} |z_k - \hat{x}_{jk}|$.
      For $l = 1$ to s
        /* Assignment of class label BPA */
        $\phi_{lk} = e^{-(d_l/c_l)} / \sum_{j=1}^{s} e^{-(d_j/c_j)}$
  Next k
  /* Combination of BPAs */
  For $l = 1$ to s

$$\phi_{lc} = \frac{\prod_{i=1}^{p} \phi_{li}}{K} \quad \text{where} \quad K = \sum_{l=1}^{s} \left(\prod_{i=1}^{s} \phi_{li} \right).$$

  /* Computation of class for z */
  $M = \underbrace{ArgMax}_{l} \{\phi_{lc}\}$
  Assign z to class M.

Our next algorithm, MRNNDST-3, is structurally the same as that of MRNNDST-2 but it uses a different ϕ function.

Algorithm MRNNDST-3

Algorithm MRNNDST-2, with $\phi_{lk} = e^{p\alpha_l d_l}$ d_l
where suffix l and k stand for lth class and kth variate.

Both of MRNNDST-1 and MRNNDST-2 produce BPAs on the set of classes. MRNNDST-1 produces a BPA using the voting concept like k-NN while MRNNDST-2 produces the same based on distances of the m-rank nearest neighbors. Too many representatives (many votes) from a particular class, say l, within the m-rank nearest neighbors provide a strong support that the z is from class l. Similarly, if the sum of distances (or the average distance) of points from, say, class l, within the m-rank nearest neighbors is very low, then this also generates a strong evidence that z is from class l. Therefore, combining the BPAs of MRNNDST-1 and MRNNDST-2 is expected to produce a more meaningful belief assignment. Our next algorithm MRNNDST-4, essentially does this. MRNNDST-4 can be schematically represented as follows.

Algorithm MRNNDST-4

Store:

$$X = \bigcup_{i=1}^{c} X_i,\ X_i = \text{set of data from class } i$$
$$= [x_{ij}]_{N\times p} = [X_L^1 | X_L^2 | \cdots | X_L^p],$$
where $N = \sum_{i=1}^{c} n_i,\ |X_i| = n_i,$
X_L^i is the list of values for
feature i for the N data points

Unknown observation $z = (z_1, \cdots, z_p)^T \in R^p$

Pick: $m \in N^+$, $P \in R^+$

Process:

Invoke MRNNDST-1 with X, z, m and P

resulting $\phi_{1c} = \dfrac{\prod_{i=1}^{c} \phi_{ic}}{K}$ where $K = \sum_{i=1}^{c}\left(\prod_{i=1}^{p} \phi_{ic}\right)$.

Invoke MRNNDST-2 with X, z and m

resulting $\phi_{2c} = \dfrac{\prod_{i=1}^{p} \phi_{ic}}{K}$ where $K = \sum_{i=1}^{c}\left(\prod_{i=1}^{p} \phi_{ic}\right)$.

Combine ϕ_{1c} and ϕ_{2c} with Dempster's rule as

$\phi_{ic} = \dfrac{\prod_{i_1 \cap i_2 = i} \phi_{1i_1}\phi_{2i_2}}{K}$, where $K = \sum_{i=1}^{c}\left(\prod_{i_1 \cap i_2 = i} \phi_{1i_1}\phi_{2i_2}\right)$.

/* Computation of class for z */

$M = \underbrace{ArgMax}_{i}\{\phi_{ic}\}$

Assign z to class M.

For algorithms MRNNDST-1 to MRNNDST-4, we have first found the m-rank nearest neighbors for each feature of the data point z. Then for each feature we aggregated the information present in the m-rank nearest neighbors or in the class-wise average distances or in both. And then defined one BPA for each feature. Finally, these BPAs are combined. Our next algorithm RNNDST uses a slightly different concept which is more similar in spirit to the k-NNDST rule. RNNDST like the other algorithms for each feature k, first finds the m-rank nearest neighbors. So, depending on the position of z_k in the sorted list $\hat{X}_L^k$, there would be m to $2m$ neighbors. Now for each neighbor we define a BPA as follows:

$$m_k^i(\{C_q\}) = \alpha_i \quad \text{and} \quad m_k^i(C) = 1 - \alpha_i, \quad \alpha_i = \alpha_0 e^{-d_i}$$

where the index i indicates the ith point in the m-rank nearest neighbors which is from class q and d_i is the distance of z_k from the ith neighbor. Hence, for each feature we can have m to $2m$ BPAs defined on the set of classes $C = \{C_1, C_2, \cdots, C_s\}$. Next for feature k, we combine the BPAs which are defined for a particular class. Thus after this step for each feature we can get at most s BPAs. Without loss of generality we assume that for each feature we have exactly s BPAs denoted by m_{kl}, $l = 1, \cdots, s$, $k = 1, \cdots, p$. We now aggregate m_{kl}, $l = 1, \cdots, s$ to get a feature-wise combined BPA, m_k using Dempster's rule. Finally the global BPA is obtained by combining m_k, $k = 1, \cdots, p$. In a nutshell

$$m_k = \Phi_{l=1}^{s} \quad m_{kl} = \oplus_{l=1}^{s} (\oplus_{\hat{x}_{ik} \in \Phi_k^{q_l}} m_k^i).$$

Here Φ_k^l = set of $\hat{x}_{ik}$ from class l that belong to the m-rank nearest neighbor of z_k.

Algorithm RNNDST

Store:

$$X = \bigcup_{i=1}^{s} X_i,\ X_i = \text{set of data from class } i$$
$$= [x_{ij}]_{N\times p} = [X_L^1 | X_L^2 | \cdots | X_L^p],$$
where $N = \sum_{i=1}^{s} n_i,\ |X_i| = n_i,$
X_L^i is the list of values for
feature i for the N data points

Unknown observation $z = (z_1, \cdots, z_p)^T \in R^p$

Pick: $m \in N^+$

Process:

For $k = 1$ to p
Sort X_L^k in ascending order to get $\hat{X}_L^k = \{\hat{x}_{tk}, t = 1, \cdots, N\}$
$c_l = |\Phi_k^l| \ \forall l.$
For $q = 1$ to s
For $i = 1$ to c_q
$d_i = |z_k - \hat{x}_{ik}|, \hat{x}_{ik} \in \Phi_k^q.$
$m^i(\{C_q\}) = \alpha_i$ where $\alpha_i = \alpha_0 e^{-d_i}$
$m^i(C) = 1 - \alpha_i$
Next i
Next q
For $q = 1$ to s
$m_q(\{C_q\}) = 1 - \prod_{i=1}^{c_q}(1 - m^i\{C_q\})$
$m_q(C) = 1 - m_q(\{c_q\})$
Next q
/* Assignment of class label BPA */
For $l = 1$ to s

$$\phi_{lk} = m_l \prod_{i \neq l}(1 - m_i)$$
$$\phi_{Ck} = 1 - \sum_{i=1}^{s} \phi_{ik}$$

Next k
/* Combination of BPAs */
For $l = 1$ to s

$$\phi_l = \frac{\prod_{k=1}^{p} \phi_{lk} + \sum_{k=1}^{p} \phi_{lk} \prod_{j \neq k} \phi_{Cj} + \sum_{k=1}^{p} \phi_{Ck} \prod_{j \neq k} \phi_{lj}}{K}$$
where $K = \sum_{i=1}^{s} \phi_i.$

/* Computation of class for z */

$M = \underbrace{ArgMax}_{i}\{\phi_i\}$

Assign z to class M.

TABLE I
COMPARISON OF DIFFERENT ALGORITHMS FOR X_1

Simulations	1			2			3			4		
Algorithms	P_1	$P2$	A	P_1	P_2	A	P_1	P_2	A	P_1	P_2	A
MRNNDST-1	10.7	4.0	7.35	5.3	5.3	5.3	12.0	10.7	11.35	12.0	6.7	9.35
MRNNDST-2	13.3	10.7	12.0	10.7	2.7	6.7	10.7	13.3	12.0	8.0	10.7	9.35
MRNNDST-3	8.0	8.0	8.0	13.3	2.7	8.0	5.3	12.0	8.65	6.7	4.0	5.35
MRNNDST-4	4.0	9.3	6.65	10.7	1.3	6.0	4.0	10.7	7.35	8.0	4.0	6.0
RNNDST	18.7	5.3	12.0	9.3	4.0	6.65	6.7	8.0	7.35	12.0	5.3	8.65
k-NN	9.3	4.0	6.65	5.3	2.7	4.0	0.0	2.7	1.35	2.7	2.7	2.7
k-NNDST	9.3	4.0	6.65	5.3	2.7	4.0	0.0	2.7	1.35	2.7	2.7	2.7
m-MRNN	17.3	28	22.65	16.0	16.0	16.0	9.3	16.0	12.65	17.3	12.0	14.65

TABLE II
COMPARISON OF DIFFERENT ALGORITHMS FOR X_2

Simulations	1			2			3			4		
Algorithms	P_1	P_2	A	P_1	P_2	A	P_1	P_2	A	P_1	P_2	A
MRNNDST-1	9.5	10.0	9.75	7.0	10.0	8.5	11.5	11.0	11.25	11.5	9.0	10.25
MRNNDST-2	14.5	16.0	15.25	14.5	12.0	13.25	13.25	12.0	12.75	18.0	10.0	14.0
MRNNDST-3	10.5	9.0	9.75	8.0	8.0	8.0	10.5	8.5	9.5	11.0	10.0	10.5
MRNNDST-4	10.0	12.5	11.25	8.5	8.0	8.25	12.5	11.0	11.75	15.0	8.0	11.5
RNNDST	8.5	10.0	9.25	8.5	11.5	10.0	11.0	9.0	10.0	15.0	10.0	12.5
k-NN	26.5	26.5	26.5	26.5	18.0	22.25	29.0	23.0	26.0	28.5	29.0	28.75
k-NNDST	24.0	25.0	24.5	21.0	17.0	19.0	24.5	19.5	22.0	26.5	22.0	24.25
m-MRNN	27.0	32.5	29.75	26.0	23.5	24.75	29.0	28.5	28.75	30.5	31.0	30.75

It may appear to the reader that this algorithm is computationally more expensive than the k-NNDST. This is usually false. For k-NNDST for every unknown point all the N distances are to be computed and sorted where the sorting complexity could be at best order $N \log_e N$. On the other hand, for RNNDST, the feature values are to be sorted only once and the ranks of the p components of z can be computed in order $p \log_2 N$ where p is much much smaller than N. Of course RNNDST computes more number of BPAs than k-NNDST, but k-NNDST requires some auxiliary computations for finding the value of γ_0, the parameter of the k-NNDST algorithm.

The performance of k-NN rule is usually good when the pattern classes have clustering tendency. There are some cases where the proposed algorithms can perform better than k-NN. Suppose for a four-dimensional (4-D) data set, out of the four features, three features have very low values while the fourth one takes very high values. In this case the distance of z from a point in the training set may be highly influenced by the fourth feature and the effect of the first three features may not be adequately reflected on the distance. For example, suppose the training set has two data points x_1 and x_2 from class 1 and 2, respectively, $x_1 = (0.1, 0.2, 0.15, 10.0)$ and $x_2 = (0.5, 0.3, 0.25, 12.0)$. Let an unknown data point z be $z = (0.1, 0.2, 0.15, 13.0)$. The distance of z from x_1 is 3 and that from x_2 is 1.086. Hence using the 1-NN rule z will be classified to class 2, although out of the four features, the values of the first three features of z exactly match with those of x_1. It, therefore, shows more evidence for class 1. Unless feature four is the only important feature (which is a very rare thing to assume) we would expect z from class 1. In the rank nearest neighbor based decision rules since we are defining BPAs feature-wise and then aggregating them we expect to get the desirable solution in such cases.

We now show some theoretical results to analyze the behavior of some of the algorithms

Theorem 1: For univariate case MRNNDST-1 is equivalent to the majority rule.

Proof: Let c_i be the number of points coming from the ith class in the m-rank nearest neighbors for all $i \in 1, 2, \cdots, s$.

For MRNNDST-1, we define $\phi_i = e^{Pc_i/2nc}/K$ and $\phi_j = e^{Pc_j/2nc}/K$ where K is a constant.

If $\phi_i > \phi_j <=> e^{Pc_i/2nc} > e^{Pc_j/2nc} <=> c_i/nc < c_j/nc$, since $nc \neq 0 <=> c_i > c_j$.

Theorem 2: If $d_i = d_j \; \forall i, j$ where d_i = sum of distances between z and points coming from ith class in the nc-rank nearest neighbor then the decision of MRNNDST-1, MRNNDST-2 and MRNNDST-3 in one dimensional (1-D) case is the same as the majority rule.

Proof: Let c_i be the number of points coming from the ith class in m-rank nearest neighbors for all $i \in 1, 2, \cdots, s$.

For MRNNDST-1, we define $\phi_i = e^{Pc_i/2nc}$ and $\phi_j = e^{Pc_j/2nc}$. MRNNDST-1 does not depend on d_i and the results follow from Theorem 1.

For MRNNDST-2, we define $\phi_i = e^{-d_i/c_i}/K$ and $\phi_j = e^{-d_j/c_j}/K$

If $\phi_i > \phi_j <=> e^{-(d_i/c_i)} > e^{-(d_j/c_j)} <=> d_i/c_i < d_j/c_j <=> 1/c_i < 1/c_j <=> c_i > c_j$ since $d_i = d_j$.

For MRNNDST-3, we define $\phi_i = e^{Pc_i/(1+d_i)}$ and $\phi_j = e^{Pc_j/(1+d_j)}$

If $\phi_i > \phi_j <=> e^{Pc_i/(1+d_i)} > e^{Pc_j/(1+d_j)} <=> c_i/1 + d_i > c_j/(1 + d_j) <=> c_i > c_j$ since $d_i = d_j$.

V. RESULTS

Before discussing the results we first present the simulation scheme. Let S be the data set. We partition S randomly into two subsets S_D (training set) and S_T (test set), such that $S_D \cap S_T = \phi$; $S_D \cup S_T = S$. For every data set S, we first use S_D as the training set and S_T as the test set (we call this case P_1) and then switch the data sets (call it P_2) and repeat the experiment. For both P_1 and P_2, we find the number of mistakes. The entire process of randomly partitioning S into S_D and S_T and computing the number of misclassification is called a simulation experiment. For each data set we made four simulations. We also report the average number of mistakes averaged over P_1 and P_2 as A in the tables with results.

We have used four data sets $X_1, \cdots, X_4$. X_1 — IRIS [12] is a 4-D ($p = 4$) data set. It contains 150 data points. Since IRIS is obtained from observations over three different physical classes of flowers, $s = 3$. But in their numerical representation, two of the classes have a large overlap while the third one is well separated from the other two.

Table I summarizes the results for X_1. For X_1, performance of k-NN, k-NNDST, MRNNDST-4 is comparable for some simulations. In all cases m-MRNN exhibited the worst performance.

TABLE III
COMPARISON OF DIFFERENT ALGORITHMS FOR X_3

Simulations	1			2			3			4		
Algorithms	P_1	P_2	A	P_1	P_2	A	P_1	P_2	A	P_1	P_2	A
MRNNDST-1	31.0	30.4	30.7	36.9	34.1	35.5	32.1	36.6	34.35	34.5	37.8	36.15
MRNNDST-2	58.3	42.7	50.5	60.7	56.1	58.4	51.2	51.2	51.2	59.5	50.0	54.75
MRNNDST-3	42.9	39.0	40.95	48.8	36.6	42.7	38.1	36.6	37.35	44.0	34.1	39.05
MRNNDST-4	42.9	34.1	38.5	47.6	43.9	45.75	34.5	39.0	36.75	42.9	36.6	39.75
RNNDST	29.8	32.9	31.35	36.9	37.8	37.35	33.3	35.4	34.35	33.3	30.5	31.9
k-NN	40.5	37.8	39.15	41.7	34.1	37.9	29.8	35.4	32.6	45.2	29.3	37.25
k-NNDST	36.9	35.4	36.15	33.3	37.8	35.55	35.7	35.4	35.55	36.9	28.0	32.45
m-MRNN	29.8	36.6	33.2	36.9	37.8	37.35	29.8	34.1	31.95	35.7	35.7	35.7

TABLE IV
COMPARISON OF DIFFERENT ALGORITHMS FOR X_7

Simulations	1			2			3			4		
Algorithms	P_1	P_2	A	P_1	P_2	A	P_1	P_2	A	P_1	P_2	A
MRNNDST-1	11.3	11.3	11.3	10.5	10.8	10.65	7.8	14.5	11.15	10.8	10.8	10.8
MRNNDST-2	13.3	9.5	11.4	14.3	11.0	12.65	19.5	20.0	19.75	14.3	10.8	12.55
MRNNDST-3	17.5	21.5	19.5	15.0	14.0	14.5	15.5	16.3	15.9	12.0	12.0	12.0
MRNNDST-4	10.3	7.3	8.8	9.3	8.3	8.8	6.8	11.8	9.3	9.3	8.0	8.65
RNNDST	13.75	12.5	13.12	10.25	11.0	10.62	13.75	11.75	12.75	12.75	12.0	12.37
k-NN	6.3	5.8	6.05	4.5	5.0	4.75	4.5	6.5	5.5	5.0	4.8	4.9
k-NNDST	6.0	5.8	5.9	4.8	4.3	4.55	4.5	7.3	5.9	5.3	4.5	4.9
m-MRNN	39.5	38.25	38.85	34.75	33.25	34.0	39.75	38.5	39.13	39.0	39.75	39.38

X_2 = a synthetic data set. This bivariate data set contains 200 points with each class having 100 points. It consists of two separated rectangular boxes in the first quadrant. For X_2 the domain of one feature is much bigger than the other.

Table II, represents the results for X_2. For this data set all of the proposed algorithms are found to show better performance than the three existing algorithms, k-NN, m-MRNN, k-NNDST. As explained earlier, for X_2 since one feature has a much larger domain than the other feature, the two distance-based classifiers, k-NN and k-NNDST exhibit poor performance and all of the proposed algorithms outperform them. All algorithms in the MRNNDST family show a remarkable improvement over the k-NN and k-NNDST.

X_3 = MANGO DATA. This is a 18-dimensional ($p = 18$) data containing 166 points [10]. This data set is generated from three different kinds of mango leaves, so $s = 3$. Here we consider only three features, one with a large domain and the remaining two with small domains. Table III displays the recognition scores for the four simulations of X_3. Out of the four simulations in three cases MRNNDST-1 and RNNDST outperform k-NN and k-NNDST, but in one case both k-NN and k-NNDST outperform MRNNDST-1 and RNNDST. All algorithms, MRNNDST-2, MRNNDST-3 and MRNNDST-4 and k-NN, which use distances, show very poor performance as expected.

X_7 = NORMAL. This is also a synthetically generated data set in 4-D ($p = 4$) with 800 points [11]. It has been generated by drawing 200 points each from four multivariate normal distributions with population mean $\mu_i = 3e_i$ and covariance $\Sigma_i = I_4, i = 1, 2, 3, 4$; e_i is the ith unit vector in R^4. Since each feature has more or less the same variance, and the clusters are reasonably separated, pure distance-based classifiers exhibit better performance than the proposed algorithms (Table IV). Here also all algorithms in the MRNNDST family, although do not perform better than k-NN or k-NNDST, exhibit significant improvement over the m-MRNN rule.

VI. CONCLUSION

We proposed five classification algorithms, which combine the features of the rank nearest neighbor classification rules and Dempster–Shafer theory of evidence in several interesting ways. These algorithms, particularly, are very useful when some features have very high values while others have low values. In such cases distance based classification rules like k-NN and k-NNDST may not work satisfactorily but the proposed schemes perform well. k-NNDST is usually found to show better performance than k-NN. In the present case, all algorithms that combine m-MRNN with D-S theory show a remarkable improvement over the m-MRNN rule. In this investigation we have used simple type BPAs. It will be more interesting to use belief functions with disjunctive clauses as focal elements so that it can deal with uncertainty about the class membership of the training data. In such cases we can use the pignistic probability distribution for final decision making. We leave it for future investigation.

REFERENCES

[1] E. Fix and J. L. Hodges, "Nonparametric discrimination: consistency properties," USAF School Aviation Medicine, Randolph Field, TX, Tech. Rep. 4, 1951.

[2] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inform. Theory*, vol. IT-13, pp. 21–26, 1967.

[3] S. A. Dudani, "The distance-weighted k-nearest neighbor rule," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-6, pp. 325–327, 1976.

[4] T. Denoeux, "A k-nearest neighbor classification rule based on Dempster–Shafer theory," *IEEE Trans. Syst., Man, Cybern.*, vol. 25, pp. 804–813, 1995.

[5] T. Anderson, "Some nonparametric multivariate procedures based on statistical equivalent blocks," in *Proc. 1st Int. Symp. Analysis*, New York, 1966.

[6] S. C. Bagui, "Nearest neighbor classification rules for multiple observations," Ph.D. thesis, Univ. Alberta, Edmonton, AB, Canada, 1989.

[7] ——, "Classification using first stage rank nearest neighbor for multiple classes," *Pattern Recognit. Lett.*, vol. 14, pp. 537–544, 1993.

[8] S. C. Bagui and N. R. Pal, "A multistage generalization of the rank nearest neighbor classification rule," *Pattern Recognit. Lett.*, vol. 16, pp. 601–614, 1995.

[9] G. Shafer, *The Mathematical Evidence with Dempster–Shafer Theory*. New York: Wiley, 1994.

[10] A. Bhattacharjee, "Some aspects of Mango (Mangifora indica L) leaf growth features in varietal recognition," M.S. thesis, Univ. Calcutta, Calcutta, India, 1986.

[11] N. R. Pal and J. C. Bezdek, "On cluster validity for the fuzzy C-means model," *IEEE Trans. Fuzzy Syst.*, vol. 3, pp. 370–379, 1995.

[12] R. Johnson and D. Wichern, *Applied Multivariate Statistical Analysis*. Englewood Cliffs, NJ: Prentice-Hall, 1988.

[13] P. Smets and R. Kennes, "The transferable belief model," *Artif. Intell.*, vol. 66, pp. 191–234, 1994.

[14] P. Smets, "The transferable belief model for quantified belief representation," in *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, D. M. Grabby and P. Smets, Eds. Norwell, MA: Kluwer, 1998, vol. 1, pp. 267–301.
[15] Z. Elouedi, K. Mellouli, and P. Smets, "Decision trees using the belief function theory," in *Proceedings 8th Int. Conf. IPMU*, vol. 1, Madrid, Spain, 2000, pp. 141–148.
[16] L. M. Zouhal and T. Denoeux, "An evidence-theoretic k-NN rule with parameter optimization," *IEEE Trans. Syst., Man Cybern.*, vol. 28, pp. 263–271, May 1998.

Existence and Construction of Weight-Set for Satisfying Preference Orders of Alternatives Based on Additive Multi-Attribute Value Model

Jian Ma, Zhiping Fan, and Quanling Wei

***Abstract*—Based on the additive multi-attribute value model for multiple attribute decision making (MADM) problems, this paper investigates how the set of attribute weights (or weight-set thereafter) is determined according to the preference orders of alternatives given by decision makers. The weight-set is a bounded convex polyhedron and can be written as a convex combination of the extreme points. We give the sufficient and necessary conditions for the weight-set to be not empty and present the structures of the weight-set for satisfying the preference orders of alternatives. A method is also proposed to determine the weight-set. The structure of the weight-set is used to determine the interval of weights for every attribute in the decision analysis and to judge whether there exists a positive weight in the weight-set. The research results are applied to several MADM problems such as the geometric additive multi-attribute value model and the MADM problem with cone structure.**

***Index Terms*—Decision analysis, extreme point, preference, weight-set.**

I. Introduction

Multiple attribute decision making (MADM) refers to making preference decisions (e.g., evaluation, prioritization, and selection) over the available alternatives that are characterized by multiple, usually conflicting, attributes. It is an important research topic with wide applications in management and engineering [1]–[5]. Current methods for MADM problems first determine the weights assigned to the attributes according to different preference information given by the decision maker. The mathematical models based on the determined weights are then used to rank the alternatives, where the most widely used model is the additive multi-attribute value model [2], [4], [5].

Sensitivity analysis is one of the hot research topics in MADM [4]. In sensitivity analysis, two or more alternatives being equal in overall utility and solution are found on the parameters (probabilities, payoffs, or weights) for which equality holds. Earlier work on sensitivity analysis can be found in Issacs [6], Fishburn *et al.* [7], Evans [8] and Schneller and Sphicas [9]. The majority of these works address the issue of sensitivity of decisions to probability estimation errors. Barron and Schmidt [10], Soofi [11], and Ringuest [12] investigate sensitivity analysis of additive multi-attribute value models. Given an initial set of weights and an outcome with maximum overall additive multi-attribute value, the proposed procedures generate new weights which equate or reverse by a prescribed amount of the overall additive multi-attribute value of the initially preferred outcome and any other nondominated outcome. Earlier work on sensitivity analysis in MADM [13]–[15] focused on probability estimates and estimates of attribute weights in an additive multi-attribute value model. However, little research has investigated the existence and structure of the weight-set while keeping the ranking orders on alternatives.

Based on the additive multi-attribute value model, this paper analyzes the structure of the weight-set and proposes a new method to determine the weight-set while keeping the ranking orders on alternatives according to linear programming theory. Given a set of ordering on alternatives, the proposed method can also tell if the attribute weights are feasible or not. Thus it provides the necessary and sufficient conditions for decision makers to adjust weights while still keeping the ranking orders. This paper also lists the type of MADM decision models where the proposed conditions are applicable. Examples are used to illustrate the application of the proposed method.

Section II of this paper introduces the additive multi-attribute value model. Section III defines the weight-set for satisfying one preference order of alternatives in a MADM problem. It also gives the sufficient and necessary conditions for the weight-set to be not empty and proposes a method to determine the structure of the weight-set. Section IV presents the weight-set for satisfying many preference orders of alternatives simultaneously. Section V investigates the application of the proposed method in MADM problems. Section VI provides remarks and Section VII summarizes the research outcomes and discusses the future work.

II. Additive Multi-Attribute Value Model

The following notations are frequently used in this paper:

$S = \{S_1, S_2, \cdots, S_m\}$: a discrete set of m possible alternatives.

$P = \{P_1, P_2, \cdots, P_n\}$: a set of n additively independent attributes.

$w = (w_1, w_2, \cdots, w_n)^T$: the vector of the relative importance or weights on the attributes, where $\sum_{k=1}^{n} w_k = 1$, $w_k \geq 0$, $k = 1, 2, \cdots, n$.

The additive multi-attribute value model is probably the simplest and still the most widely used MADM model [2], [15]. It can be expressed as

$$\varphi_r = \varphi(S_r) = \sum_{k=1}^{n} w_k \varphi_k(x_{rk}), \qquad r = 1, 2, \cdots, m$$

where $\varphi(S_r)$ is the value function of alternative S_r, and w_k and $\varphi_k(x_{rk})$ are weight and value functions of attribute P_k, respectively. Through the normalization process, each incommensurable attribute becomes a pseudovalue function, which allows direct addition among attributes. The overall value of alternative S_r can be rewritten as

$$\varphi_r = \sum_{k=1}^{n} w_k a_{rk}, \qquad r = 1, 2, \cdots, m$$

where a_{rk} is the comparable scale of x_{rk}, which can be obtained through normalization.

Manuscript received May 14, 1998; revised October 31, 2000. This work was supported in part by the Competitive Earmarked Research Grant (CERG) of Hong Kong under Project 9040375, the National Natural Science Foundation of China (NSFC) under Project 19471085, 79600006, and 70071004, the NSFC/Research Grant Council Joint Fund under Project 9050137, and the Strategic Research Grant of City University of Hong Kong under Project 7001017. This paper was recommended by Associate Editor A. Zomaya.

J. Ma is with the Department of Information Systems, City University of Hong Kong, Kowloon, Hong Kong.

Z. Fan is with the Department of Information and Decision Sciences, Northeastern University, Shengyang, China.

Q. Wei is with the Institute of Operations Research and Mathematical Economics, Renmin's University of China, Beijing, China.

Publisher Item Identifier S 1083-4427(01)01010-4.