Applied Intelligence 6, 75–86 (1996)

Knowledge Integration in a Multiple Classifier System

YI LU
Department of Electrical and Computer Engineering, The University of Michigan-Dearborn, Dearborn, MI 48128-1491
lu@umdsun2.umd.umich.edu

Received December 20, 1993; Revised August 9, 1994; Accepted October 7, 1994

Abstract. This paper introduces a knowledge integration framework based on Dempster-Shafer's mathematical theory of evidence for integrating classification results derived from multiple classifiers. This framework enables us to understand in which situations the classifiers give uncertain responses, to interpret classification evidence, and allows the classifiers to compensate for their individual deficiencies. Under this framework, we developed algorithms to model classification evidence and combine classification evidence from difference classifiers, we derived inference rules from evidential intervals for reasoning about classification results. The algorithms have been implemented and tested. Implementation issues, performance analysis and experimental results are presented.

Keywords: reasoning under uncertainty, pattern recognition, classification, evidential reasoning, handwritten digit recognition, multiple classifiers

1. Introduction

In the field of pattern recognition, it has been well recognized that a multiple classifier system is effective in solving complicated problems such as handwritten digit and word recognition [15]. The multiple classifiers in a system often use different feature sets and classification methods to achieve a high recognition rate. A multiple classifier system can be in either cascaded, parallel or hierarchical configurations (see Fig. 1). In a cascaded system, the classification results generated by a classifier are often used to direct the classification processes of the successive classifiers. The problem with this type of configuration is that since errors made by previous classifiers are not recoverable by the successive classifiers, the overall system error is the accumulation of the errors of individual classifiers in the system. In a parallel system, the classifiers generate results independently and then a decision process integrates the results from all the classifiers. If the decision process is well designed, the overall system may reach peak performance. In addition, classifiers in the parallel configuration can be implemented by parallel processors to achieve real-time performance. In a hierarchical system, the control strategy is a combination of cascaded and parallel processing.

In this paper, we address the problem of decision processes in a parallel classification system. The results generated by individual classifiers are considered knowledge from different sources, therefore the decision process is truly the knowledge integration. The classification results from individual classifiers may contain uncertain, imprecise and inaccurate information. The goal of any classification result integration (CRI) algorithms is to generate more certain, precise, and accurate system results. A good summary of existing techniques for combining classification results can be found in [7, 15].

The most widely-used approach in dealing with uncertainty is the Bayesian method. The Bayesian method is based on a well-understood technique from probability theory. However, the Bayesian approach has been widely criticized for requiring an agent to assign a subjective prior probability to every event. There

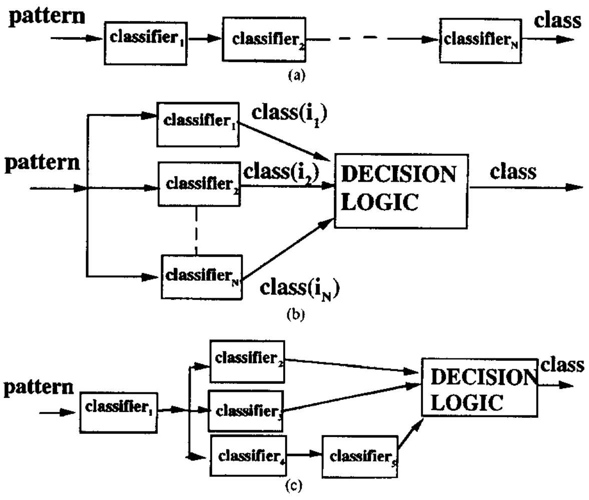


Figure 1. Configurations of multiple classifier systems. (a) Cascading, (b) parallel, (c) hierarchy.

is also the issue of whether it is reasonable to describe confidence by a single point rather than a range [3, 6]. In order to overcome the limitation of the Bayesian approach, many other approaches have been proposed, such as Dempster-Shafer theory [2, 12], Cohen's model of endorsements [1] and fuzzy logics [16]. The Dempster-Shafer approach recognizes the distinction between uncertainty and ignorance by creating belief functions, which is based on a relatively new body of mathematics commonly called the evidential reasoning. Under evidential reasoning, the fundamental measure of belief is represented as an interval bounding the probability of a proposition, thus allowing the representation of ignorance as well as uncertainty. Furthermore, the theory provides a mechanism for pooling multiple bodies of evidence. Dempster-Shafer's evidential reasoning theory has been investigated for sensor fusion [5] and combining classification results in pattern recognition by Xu et al. [15] at the Concordia University in Canada, and Mandler and Schuermann at the AEG Research Center Ulm [MaS88]. Xu et al.'s method was devised for classification results in the format of multiple outputs, i.e., the output from a classifier contains a list of possible classes [15]. The basic probability assignment function was defined based on the recognition rate and the substitution rate of classifiers on different classes. Mandler and Schuermann's method in [MaS88] transforms the distance measures of different classifiers into confidence values.

Our approach applies to the classification results that contain both class labels and the associated confidence values. We present algorithms for modeling and combining digit classification results, and rules for reasoning about more certain and accurate classification results based on combined evidence. We conducted experiments within the environment of handwritten digit recognition. The input to the proposed system were the classification results generated by three different handwritten digit recognition classifiers, neural net (NN), structural template matching (ST), and polynomial classifier (PL). The input data to these classifiers were digits automatically segmented from handwritten ZIP Codes on U.S. mail pieces. In order to have a meaningful comparison, we have implemented Bayesian method in our experiments. The results of the proposed evidential reasoning method are far superior than to those of the Bayesian method.

We begin with a brief overview of the Dempster-Shafer theory. Section 3 describes basic pattern recognition theories, algorithms for modeling classification results and evidential reasoning. Section 4 describes implementation, performance issues and experiment results, and Section 5 concludes our work.

2. Dempster-Shafer Representation

Dempster-Shafer theory can be defined by *basic probability assignments* across the propositions in Θ, where Θ is called the frame of discernment. A basic probability assignment, or mass assignment M on Θ satisfies:

1. $M(\phi) = 0$,
2. $\sum_{A \subset \Theta} M(A) = 1$.

The quantity $M(A)$ is called proposition A's basic probability number, and is understood to be the measure of the belief that is committed exactly to A. This representation allows one to specify one's belief at exactly the level of detail that one desires and at the same time ignores the propositions about which he doesn't have knowledge. From a basic probability assignment, we can fully describe the evidence for a proposition Q through a pair of functions, support function $\text{Spt}(Q)$ and plausible function $\text{Pls}(Q)$ defined as follows:

$$\text{Spt}(Q) = \sum_{A \subset Q} M(A), \quad \text{Pls}(Q) = 1 - \text{Spt}(\bar{Q}).$$

$\text{Spt}(Q)$ describes the degree of belief and $\text{Pls}(Q)$ describes the upper probability of Q. $[\text{Spt}(Q), \text{Pls}(Q)]$

is called an evidential interval and, $u(Q) = \mathrm{Pls}(Q) - \mathrm{Spt}(Q)$ is a measure of uncertainty.

Dempster's rule of combination pools multiple bodies of evidence represented by the basic probability assignments. Dempster's rule takes two basic probability assignments M_1 and M_2, and produces a new basic probability assignment M such that

$$M(Q) = \frac{1}{1-K} \sum_{A \cap B = Q} M_1(A) * M_2(B)$$
$$K = \sum_{A \cap B = \phi} M_1(A) * M_2(B).$$

M is called the orthogonal sum of M_1 and M_2 indicated by $M = M_1 \oplus M_2$, K is a measurement of-conflict between M_1 and M_2. As long as M_1 and M_2 are not completely contradictory, i.e., $K \neq 1$, M is well defined. Dempster's rule is both commutative and associative.

3. Evidence Modeling and Reasoning

Our intention is to treat classifiers as specialized knowledge sources. In this section we shall describe methods to model and combine such knowledge sources, and the subsequent reasoning process. The following discussion assumes we have q classifiers, $\mathrm{CL}_1, \mathrm{CL}_2, \ldots, \mathrm{CL}_q$, p classes in the pattern space, $\{C_0, C_1, \ldots, C_p\}$.

3.1. The Problem of Pattern Classification

The pattern classification problem can be defined formally by a triplet $\langle P, F, M \rangle$, where P is the pattern space consisting of p mutually exclusive sets, i.e., $P = C_0 \cup C_1 \cup \cdots \cup C_p$, C_i is often referred to as a class and $C_i \mid C_j = \emptyset$ for any $i \neq j, i \leq p$ and $j \leq p$; F is a N-dimensional feature vector space that represents the patterns in the pattern space; and M is a function or a classifier that maps feature vectors in F to the classes in pattern space P. The pattern classification problem is to find a mapping function M that defines a partition of the feature space such that different sets in the partition can be mapped to different classes in P. However in many applications, such a partition often does not exist in the feature space. Figure 2 illustrates this concept. The pattern space is on the left side and the feature space is on the right side. Feature vectors in the shaded area may be mapped to more than one classes. For example the feature vectors in the dark shaded area of feature sets F_0 and F_1 can be mapped to pattern classes in either C_0 or C_1; and the feature vectors in the gray shaded area can be mapped to pattern classes in either C_0, C_1 or C_p. It is also possible that some feature vectors have no corresponding classes. This demonstrates the uncertain and inaccurate situations a classifier system may need to solve.

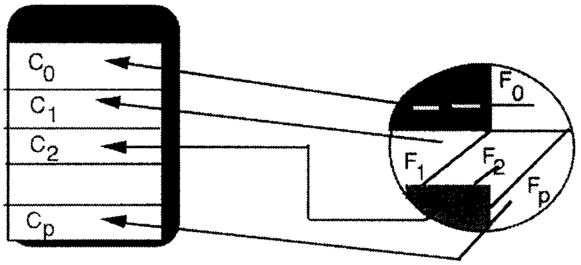


Figure 2. Mapping from feature space to pattern space.

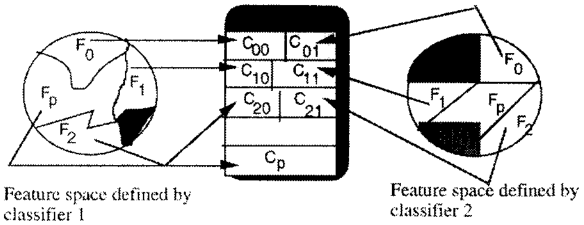


Figure 3. Illustration of orthogonal feature mapping.

Ideally, we design multiple classifiers to have orthogonal feature sets so that they can compensate each other. If we divide each class C_i in pattern space into a partition of subclasses, each classifier will map a number of subclasses to a partition of its subfeature space. Ideally, the union of the subclasses that each classifier can uniquely identify is equal to the entire pattern space. Figure 3 illustrates this argument. Classifier 1 provides a unique map from vectors in the unshaded area of F_0 to the patterns in subclass C_{01}, and from the vectors in F_1 to subclass C_{11}. Classifier 2 provides a unique map from vectors in the unshaded area of F_0, F_1 and F_p to the patterns in subclass C_{00}, C_{10} and C_p respectively. Note only classifier 2 has a map between the class C_p and the feature set F_p. After combining classifier 1 and classifier 2 all patterns can be uniquely identified.

Because of the ambiguity in the feature to pattern mapping found in many application problems it is well recognized that it is advantageous to generate more than one classification results [8, 9]. Specifically we need a technique that allows individual classifiers to

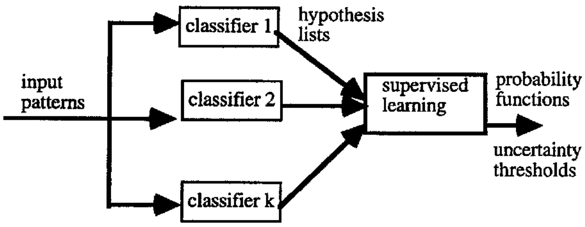


Figure 4. Modeling classification results.

represent "I don't know", or "I think the input may be class i, however my confidence is not high" or "the input is either class i or j but not k". A hypothesis scheme [9] was developed to represent such uncertain and imprecise results.

3.2. Modeling Classification Results

The modeling process is built upon a training data set containing classification results from individual classifiers. Figure 4 illustrates the modeling processes. We shall begin by discussing how the usual measurements performed during pattern to basic probability assignment.

3.2.1. Hypothesis Generation. The hypothesis scheme requires a classifier to generate for each input pattern a hypothesis list. A hypothesis list contains pattern classes that closely match the unknown input. Additionally, each class on the list is associated with a numeric value indicating the confidence of the classification. Classifiers generating hypothesis lists can be evaluated according to two measures: reliability and efficiency. Reliability is the frequency with which the correct class is included in the hypothesis list, and efficiency is the average length of the hypothesis list. Good classifiers are both efficient and reliable. An effective classifier will produce short hypothesis lists which are correct.

A pattern classifier is expected to capture the dissimilarity between different classes and the similarity between the patterns of the same class, as well as describe possible misclassifications and indicate "I don't know" for inputs for which it cannot find good classifications. The proposed hypothesis-generating strategy provides these capabilities. It reduces erroneous responses by allowing a classifier to produce multiple responses when a unique response cannot be determined. The confidence values permit identification of ambiguous patterns and situations like "I don't know". If the difference between the highest and the second highest confidence value is large, it indicates that the class with the highest confidence value is almost certainly the true class of the input; if the first $k(k > 1)$ classes on the hypothesis list have very close confidence values, it indicates that the input is ambiguous; if the highest confidence value on the hypothesis list is very low, it is an indication of "I don't know". Obviously, if it is desired, a unique response can be derived from a hypothesis list by applying a decision process to the hypothesized classes and their associated confidence values. In a system of multiple classifiers, it is possible either to pass a hypothesis list generated by a classifier on to other classifiers for further processing, or to apply a decision process to all the hypothesis lists generated by the classifiers in the system to achieve a reliable and unique classification.

Based on the above discussion, we can define a hypothesis-confidence list, $(C_0, \mathrm{Cf}_0; C_1, \mathrm{Cf}_1, \ldots, C_p, \mathrm{Cf}_p)$, where Cf_i is 0 if C_i is not on the hypothesis list otherwise Cf_i equals the confidence value attached to C_i on the hypothesis list. In the following discussion, we assume all classifiers under consideration produce hypothesis-confidence lists as output. Figure 5 shows an example of hypothesis-confidence list. The digit classifier generates three hypotheses for the input pattern, digit classes 3, 5, or 9 with corresponding confidence values 90, 50 and 70. The list above the output arrow is the hypothesis list, and the one below is the hypothesis-confidence list. Apparently the two lists, hypothesis list and the hypothesis-confidence list, are equivalent, namely, a hypothesis-confidence list can be derived from a hypothesis list and vice versa.

3.2.2. Computing Probability-Lists. To combine the classification results from different classifiers, a critical

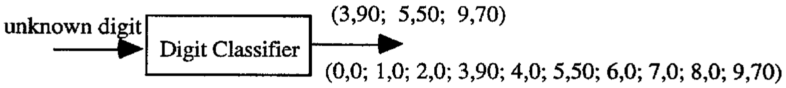


Figure 5. An example of hypothesis and hypothesis-confidence list.

issue we must address is the inconsistent confidence functions used by the different classifiers. Different classifiers can have different strategies to generate their confidence values. There are two problems involved. First the confidence values from all classifiers should have the same range. One classifiers may generate confidence values ranging from 0 to 1, and others may range from 1 to 100. The second problem is that the same confidence value from different classifiers may represent different degrees of support. For example, confidence value "60" generated by classifier 1 may represent the equivalent support to confidence value "80" generated by classifier 2. The first problem can be solved by mapping confidence values to a defined range. Therefore, without loss generality, we assume in the following description that the confidence values generated by all classifiers range from 0 to 100. The second problem is solved by using the following machine learning technique.

Let Γ_i^j be a set of hypothesis-confidence lists generated by classifier CL^j from a set of patterns belonging to class C_i, $R_i^j(\mathrm{Cf})$ be a reliability function, $P_i^j(\mathrm{Cf})$ a probability density function of confidence value Cf with classifier CL^j for class C_i,

- For all values of Cf's > 0, compute $R_i^j(\mathrm{Cf})$, the number of the hypothesis-confidence lists in Γ_i^j that contain the confidence value Cf for class C_i
- Let N_i^j be the number of hypothesis-confidence lists in Γ_i^j that contain a nonzero confidence for class C_i. The probability of the pattern with confidence value Cf for class C_i is $P_i^j(\mathrm{Cf}) = \frac{R_i^j(\mathrm{Cf})}{N_i^j}$.
- Find an uncertainty threshold UC_i^j such that for all $\mathrm{Cf} \geq \mathrm{UC}_i^j$, the $R_i^j(\mathrm{Cf})$ can be approximated by a monotonic curve.

The reliability function $R_i^j(\mathrm{Cf})$ represents the number of patterns in class C_i that are correctly identified with confidence value Cf. In theory, the higher confidence value, more reliable the result is, and therefore, the reliability function should be monotonic in theory. However in many applications, because of the noise in training data set and/or deficiencies in classification algorithms, the reliability function is not monotonic in particular when confidence values are low. The uncertain threshold UC_i^j is used to find the range of confidence values that are meaningful in the reliability function R_i^j. The confidence values above UC_i^j form a monotonic curve, therefore they are more reliable than those below UC_i^j. The uncertainty threshold will be used later in the reasoning process.

Now we want to show that $P_i^j(\mathrm{Cf})$ is indeed a probability density function of Cf defined on the training set Γ_i^j.

(1) We know by the definition that $0 \leq P_i^j(\mathrm{Cf}) \leq 1$.

(2) $$\int_{-\infty}^{\infty} P_i^j(\mathrm{Cf})\,d\mathrm{Cf} = \sum_{\mathrm{Cf}>0} P_i^j(\mathrm{Cf}) = \sum_{\mathrm{Cf}>0} \frac{R_i^j(\mathrm{Cf})}{N}$$

$$= \sum_{\mathrm{Cf}>0} \frac{\text{number of hc-lists containing Cf for } C_i}{N}$$

$$= \frac{\text{number of hc-lists containing a Cf} > 0 \text{ for } C_i}{N}$$

$$= \frac{N}{N} = 1$$

where hc-list represents the hypothesis-confidence list.

(3) Based on (2) we have

$$P_i^j(\bar{\mathrm{Cf}}) = \sum_{\mathrm{Cf}_k \neq \mathrm{Cf}} P_i^j(\mathrm{Cf}_k) = 1 - P_i^j(\mathrm{Cf}).$$

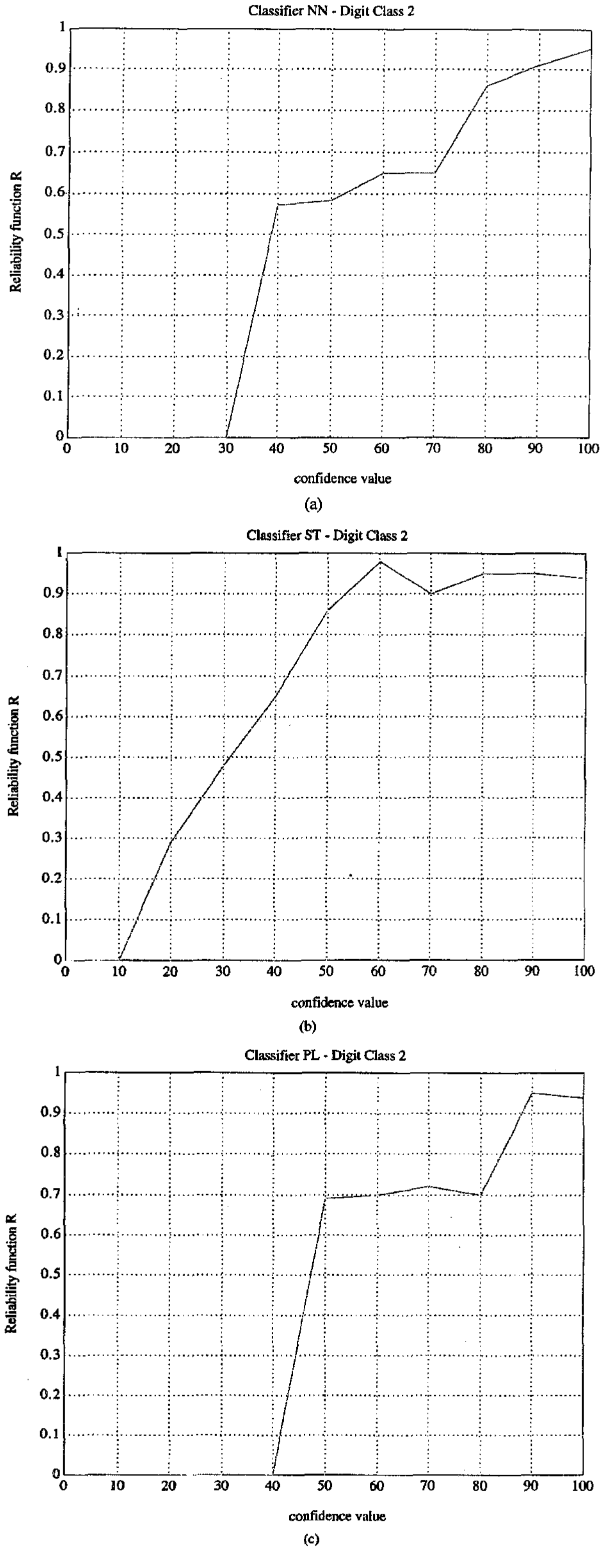
Once the probability density function (Cf) is obtained, it can be applied to any hypothesis-confidence list, $\mathrm{HC}^j = (C_0, \mathrm{Cf}_0; C_1, \mathrm{Cf}_1; \ldots, C_p, \mathrm{Cf}_p)$ generated by classifier CL^j, to achieve a hypothesis-probability list,

$$\mathrm{HP}^j = \left(C_0, P_0^j(\mathrm{Cf}_0); C_1, P_1^j(\mathrm{Cf}_1); \ldots, C_p, P_p^j(\mathrm{Cf}_p)\right).$$

The above steps need to be applied to every classifier and every pattern class in order to obtain R_i^j and P_i^j for all $0 \leq j \leq q$ and all $0 \leq i \leq p$. Figure 6 shows three R_i^j functions generated for handwritten digit class 2 from a training set containing hypotheses generated by three classifiers, a back propagation based Neural Net classifier, template matching and polynomial classifier. The detailed description of these algorithms can be found in [4, GaW90]. The uncertainty thresholds in Fig. 6(a) to (c) are 30, 40 and 10.

3.3. Belief Functions

Let the frame of discernment be $\Theta = \{C_0, C_1, \ldots, C_p, G\}$, where G represents any pattern



(a)

(b)

(c)

Figure 6. Reliability functions of three classifiers for digit class 2.

other than $C_0, C_1, \ldots, C_p$. We assume a hypothesis-probability list generated by classifier CL_i for an input pattern is

$$\mathrm{HP}^j = \left(C_0, P_0^j(\mathrm{Cf}_0^j); C_1, P_1^j(\mathrm{Cf}_1^j); \ldots; \right. \\ \left. C_p, P_p^j(\mathrm{Cf}_p^j)\right).$$

Let $\alpha = \sum_{k=0}^{p} P_k^j(\mathrm{Cf}_k^j)$. The mass function M^j for classifier CL^j can be computed as follows:

$$m^j(C_i) = \begin{cases} \dfrac{p_i^j(\mathrm{Cf}_i^j)}{\alpha^j} & \alpha^j \geq 1 \quad \& \ \mathrm{Cf}_i^j \geq \mathrm{UC}_i^j \\ p_i^j(\mathrm{Cf}_i^j) & \alpha^j < 1 \quad \& \ \mathrm{Cf}_i^j \geq \mathrm{UC}_i^j \\ 0 & \mathrm{Cf}_i^j < \mathrm{UC}_i^j \end{cases} \tag{1}$$

where $i = 0, \ldots, p$. Since $M^j(C_i) = \frac{p_i^j(\mathrm{Cf}_i^j)}{\alpha}$ only when $\alpha^j \geq 1$ and $\mathrm{Cf}_i^j \geq \mathrm{UC}_i^j$, it is obvious that $\sum_{i=0}^{p} M^j(C_i) \leq 1$.

The quantity $1 - \sum_{i=0}^{p} M^j(C_i) \geq 0$ indicates the uncertainty of the classifier for this input. This uncertainty can be interpreted as the mass function value for Θ, i.e.,

$$M^j(\Theta) = 1 - \sum_{i=0}^{p} M^j(C_i).$$

This interpretation is quite reasonable since Θ contains G, which represents anything other than the pattern classes. Thus $M^j(\Theta)$ represents the portion of the belief that could not be ascribed to any subset of Θ based on the evidence M^j has at present. This representation has several advantages. It allows the reasoning process identify and later the system reject the input that are not in the pattern space. In an OCR system, for example, text lines, words and characters are all extracted automatically by the processes preceded classification. This input to character classification algorithms may very likely contain partial or merged characters or sometimes even non-character patterns, and therefore a reasoning system using automatically processed data should be prepared to reject input. Another advantage of this representation is that it provides the opportunity for combining conflict results generated by individual classifiers. How this is accomplished will be made more clear during the evidence combination discussion.

Obviously,

$$M^j(\phi) = 0 \text{ and } \sum_{A \subset \Theta} M^j(A) = \sum_{i=0}^{p} M^j(C_i) + M^j(\Theta) = 1$$

Hence the function M^j satisfies two conditions of the basic probability assignment in Dempster-Shafer theory presented in Section 2.

3.4. Computing Composite Mass Functions

After obtaining the mass function M^j for each classifier, we form composite mass functions based on the Dempster's rule of combination described in Section 2. For example combining mass functions from M^j and M^l:

$$\begin{aligned} M^{j,l}(C_i) &= M^j(C_i) \oplus M^l(C_i) \\ &= \frac{1}{1-k}[M^j(C_i) * M^l(C_i) \\ &\quad + M^j(C_i) * M^l(\Theta) + M^j(\Theta) * M^l(C_i)] \end{aligned}$$

where $k = \sum_{i \neq j} M^j(C_i) * M^l(C_j)$.
The composition function $M^{j,l}(C_i)$ satisfies the two conditions of the basic probability function, and is ready to be combined with other mass functions, i.e.,

$$M = M^1 \oplus M^2 \oplus \cdots \oplus M^q$$

Clearly, this operation is commutative and associative.

The evidence modeling and pooling process described above has the following characteristics:

(1) If a pattern class receives support from all classifiers, its composite mass function has a higher value than the pattern classes that receive partial support.
(2) If a classifier gives a very strong support to a class, the composite mass function value for the class may still be significant even if the other classifier has no evidence to support that class.
(3) If the composite mass function is evenly distributed among a number of pattern classes, it is likely that the input is an ambiguous pattern.
(4) If the input pattern is not in the pattern class space, the composite mass function should give insignificant support to any pattern classes, and consequently $M(\Theta)$ is more significant than any othe $M(C_i)$.

3.5. Reasoning Process

For each input pattern, we generate an evidential interval $[\text{Spt}(C_i), \text{Pls}(C_i)]$ for each class C in the frame of discernment Θ from the composition mass function M. Based on the nature of our modeling process, the evidential interval can be computed by the following formula:

$$\text{Spt}(C_i) = M(C_i) \text{ and } \text{Pls}(C_i) = 1 - \sum_{j \neq i} M(C_j)$$

We have two special cases to prove. First we need to show Θ [1, 1], i.e.,

$$\text{Spt}(\Theta) = \text{Pls}(\Theta) = 1.$$
$$\text{Spt}(\Theta) = \sum_{A \subset \Theta} M(A) = \sum_{i} M(C_i) + M(\Theta) = 1$$

Obviously

$$\text{Pls}(\Theta) = 1 - \text{Spt}(\bar{\Theta}) = 1$$

The evidential interval may have one of the following forms:

(1) $C_i[0, 0]$: the input pattern is not C_i
(2) $C_i[0, 1]$: no evidence to support C_i. It implies that no evidence to support any other classes either, therefore the input is garbage.
(3) $C_i[1, 1]$: the input pattern is C_i, output C_i and stop the process
(4) $C_i[0.3, 1]$: evidence provides partial support for C_i
(5) $C_i[0, 0.7]$: evidence provides partial support for other classes
(6) $C_i[0.3, 0.8]$: evidence provides partial support for class C_i and other classes

After the knowledge is represented in the evidential intervals, a set of inference rules will be applied to the support and plausibility estimates for achieving desired classification results. The format of inference rules introduced in this paper has the format of "A ⇒ B", which can be interpreted as "statement B can be inferred from the statement A".

If our goal is to find a single classification result, the reasoning process needs to define an order on the evidential intervals. Often we want to find the class C with the strongest supporting evidence. Under this assumption, two steps of processes can be performed. First we may want to find the class that has the maximum Spt value. If there is a tie, we want find the class which has the maximum Pls value. If there is still a tie, the process will output all the propositions as the possible classes of the input pattern. Formally, we can define the following rules:

Rule 1:

$$\begin{array}{l} C_i[\mathrm{spt}(C_i), \mathrm{pls}(C_i)] \\ C_j[\mathrm{spt}(C_j), \mathrm{pls}(C_j)] \end{array} \mathrm{spt}(C_i) > \mathrm{spt}(C_j) \Rightarrow C_i > C_j$$

Rule 2:

$$\begin{array}{l} C_i[\mathrm{spt}(C_i), \mathrm{pls}(C_i)]\mathrm{spt}(C_i) \ \ = \mathrm{spt}(C_j) \\ C_j[\mathrm{spt}(C_j), \mathrm{pls}(C_j)]\mathrm{pls}(C_i) > \mathrm{pls}(C_j) \end{array} \Rightarrow C_i > C_j$$

Rule 3:

$$\begin{array}{l} C_i[\mathrm{spt}(C_i), \mathrm{pls}(C_i)] \\ C_j[\mathrm{spt}(C_j), \mathrm{pls}(C_j)] \end{array} \Rightarrow$$

$$\begin{array}{l} \qquad\qquad \mathrm{spt}(C_i \cup C_j) = \max\{\mathrm{spt}(C_i), \mathrm{spt}(C_j)\} \\ C_i \cup C_j[\mathrm{spt}(C_i \cup C_j), \mathrm{pls}(C_i \cup C_j)] \\ \qquad\qquad \mathrm{pls}(C_i \cup C_j) = \min\{1, \mathrm{pls}(C_i) + \mathrm{pls}(C_j)\} \end{array}$$

For example,

(1) If we have evidence intervals 4[0.2, 0.4], 7[0.2, 0.3], 9[0.4, 0.6], we have relation: 9 > 4 > 7. The single classification result will be digit class 9.
(2) If we have evidence intervals 4[0.2, 0.4], 7[0.2, 0.3], 9[0.1, 0.6], digit class 4 and 7 have a tie in spt value. However, digit class 4 has a larger Pls value, and therefore we have 4 > 7 > 9, and output is digit class 4.
(3) If we have 4[0.2, 0.6], 7[0.2, 0.6], 9[0.2, 0.6], then (4, 7, 9)[0.2, 1] and the output is (4, 7, 9). This result is ready to be combined with other available knowledge sources for further process.

Rules 1 and 2 are aimed at deriving single classification results, and Rule 3 is used to generate multiple classification results. If we have multiple knowledge sources represented in evidential intervals, we can integrate the knowledge directly from evidential intervals by using the following rules:

Rule 4:

$$\begin{array}{l} C_i[\mathrm{spt1}(C_i), \mathrm{pls1}(C_i)] \\ C_i[\mathrm{spt2}(C_i), \mathrm{pls2}(C_j)] \end{array} \Rightarrow$$

$$\begin{array}{l} \qquad\qquad \mathrm{spt}(C_i) = \max\{\mathrm{spt1}(C_i), \mathrm{spt2}(C_i)\} \\ C_i[\mathrm{spt}(C_i), \mathrm{pls}(C_i)], \\ \qquad\qquad \mathrm{pls}(C_i) = \min\{\mathrm{pls1}(C_i), \mathrm{pls2}(C_i)\} \end{array}$$

4. Algorithm Implementation, Performance Analysis and Experiments

The modeling and reasoning algorithms has been implemented in the programming language C on a SUN Sparc10workstation. The entire system has two major parts: classifier modeling and classification result reasoning. The classifier modeling module is designed to generate from a training set of data the reliability and the probability density function for every classifier and every pattern class. The output of this program are probability density functions in a tabular format and the uncertainty thresholds. Each classifier has a corresponding table of probability density function and uncertainty thresholds. Each table has m rows and n columns, where m is the number of pattern classes and n is the number of confidence values. The table entry at (i, j) represents the probability of pattern class j with the given confidence value i. The probability density functions can be stored in off-line files. The classification result reasoning process consists of the following steps:

- For every unknown from a classifier j, obtain mass function M^j from the probability density tables and the uncertainty thresholds during the modeling process.
- Obtain the combined mass function M by applying orthogonal sum to mass functions M^j.
- Generate evidential intervals from mass function M.
- Obtain the classification results by applying the inference rules to the evidential intervals.

We conducted experiments to test the algorithm within the environment of handwritten digit recognition. In the experiment, we used two data sets, training and test sets. Both data sets were obtained from handwritten ZIP Codes cropped from handwritten address blocks of U.S. mail pieces. The ZIP Codes were segmented by a computer program into individual digits which were in turn inputs to the three classifiers. Since the digit segmentation was automatic without manual cleaning up, the input to the classifiers contained a good portion of non-digit patterns. Figure 7 shows five examples of bad input in our experiment. The training set contained 388 handwritten ZIP Codes, approximately 2716 digits, and the test set contained 581 handwritten ZIP Codes, approximately 4067 digits.

The classification results, which were in the format of hypothesis-confidence lists, were generated by three

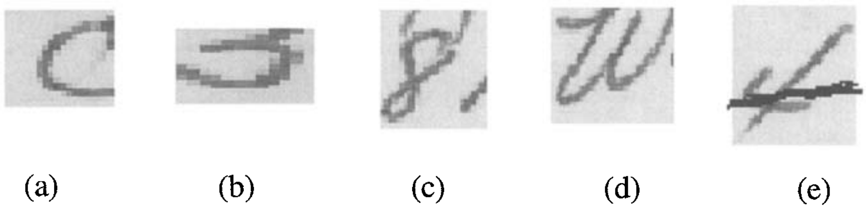


Figure 7. Examples of bad input in the data used in the experiments: (a) is an image of "0" being cut off from the right side, (b) is an image of "3" being cut off from the top, (c) is an image of "8" containing extraneous features, (d) is an image of "w", which does not belong to the digit domain, (e) is an image of "4" containing a cross line.

different classifiers, neural net (NN), structural template matching (ST), and polynomial classifier (PL). It is important to point out that the evidential reasoning algorithm we described above can be applied to any number of classifiers, although we use three classifiers in our experiments. Each classifier generated a hypothesis-confidence list for each input pattern in the format described in Section 3. In generating the reliability and probability functions, we scaled the confidence values to 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100. At the end of the modeling process, a table of probability density function was produced for each classifier. A table of probability density function for classifier CL_j has 10 rows and 10 columns. Each row represents P_i^j of $i = 0, \ldots, 9$, where P_i^j is the probability function of classifier j for digit class i. Tables 1, 2 and 3 show the probability functions generated during our experiment for classifiers NN, ST and PL respectively. These tables were stored in files to be used in the reasoning process. The modeling process was performed on the training data set.

The next issue needs to be discussed is how to evaluating a classification results integration (CRI) algorithm. We must realize that a CRI algorithm is not necessarily out perform an individual classifier as we experienced with the Bayesian method. Therefore one evaluation criterion is that a good CRI algorithm should at least perform better than any individual classifiers being combined. On the other hand, we must also realize that the performance of a CRI algorithm is bounded

Table 1. Probability function for NN classifier.

Cf/ class	10	20	30	40	50	60	70	80	90	100
0	0.00	0.00	0.00	0.00	1.00	0.44	0.67	0.73	0.88	0.97
1	0.00	0.00	0.00	0.00	0.22	0.29	0.57	0.83	0.85	0.91
2	0.00	0.00	0.00	0.57	0.56	0.65	0.65	0.86	0.91	0.95
3	0.00	0.00	0.00	0.00	0.10	0.00	0.30	0.55	0.81	0.96
4	0.00	0.00	0.00	0.12	0.00	0.23	0.20	0.08	0.56	0.89
5	0.00	00.0	00.0	0.33	0.25	0.00	0.33	0.40	0.83	0.90
6	0.00	0.00	0.00	0.00	0.10	0.00	0.11	0.27	0.64	0.92
7	0.00	0.00	0.00	0.00	0.18	0.33	0.25	0.47	0.86	1.00
8	0.00	0.00	0.00	0.00	0.00	0.57	0.20	0.21	0.89	0.85
9	0.00	0.00	0.00	0.00	0.00	0.50	0.60	0.50	0.93	0.90

Table 2. Probability function for ST.

Cf/ class	10	20	30	40	50	60	70	80	90	100
0	0.00	1.00	0.17	0.21	0.72	0.85	0.93	0.91	0.89	0.89
1	0.00	1.00	0.33	0.50	1.00	1.00	0.67	0.67	1.00	0.86
2	0.00	0.29	0.48	0.65	0.86	0.98	0.90	0.95	0.95	0.94
3	0.00	0.00	0.33	0.52	0.82	0.97	0.93	0.96	0.88	0.93
4	0.00	0.11	0.14	0.44	0.65	0.70	0.92	0.82	0.92	0.85
5	0.00	0.33	0.50	0.50	0.91	0.89	0.94	0.87	0.92	0.90
6	0.00	0.33	0.10	0.29	0.40	0.86	0.93	1.00	0.89	0.96
7	0.00	0.00	0.00	0.18	0.54	0.40	1.00	0.64	0.88	0.89
8	0.00	0.00	0.00	0.38	1.00	0.82	1.00	0.73	0.83	1.00
9	0.00	0.00	0.20	0.43	0.60	0.80	0.79	1.00	0.95	0.86

Table 3. Probability function for PL.

Cf/ class	10	20	30	40	50	60	70	80	90	100
0	0.00	0.00	0.00	0.50	0.38	0.54	0.60	0.62	0.88	0.90
1	0.00	0.00	0.00	0.50	0.00	0.33	0.44	0.63	0.89	0.86
2	0.00	0.00	0.00	0.00	0.69	0.70	0.72	0.70	0.95	0.94
3	0.00	0.00	0.00	0.33	0.00	0.27	0.85	0.81	0.91	0.92
4	0.00	0.00	0.00	0.00	0.21	0.41	0.25	0.61	0.85	0.86
5	0.00	00.0	00.0	0.00	0.14	0.40	0.64	0.89	0.89	0.88
6	0.00	0.00	0.00	0.00	0.14	0.00	0.54	0.72	0.83	1.00
7	0.00	0.00	0.00	0.00	0.00	0.22	0.46	0.65	0.82	0.91
8	0.00	0.00	0.00	0.00	0.18	0.21	0.50	0.74	0.90	1.00
9	0.00	0.00	0.00	0.00	0.00	0.38	0.70	0.79	0.90	0.96

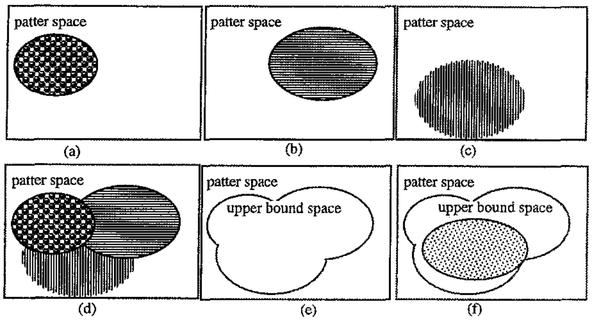


Figure 8. Illustration of upper bound performance of a CRI algorithm applied to three classifiers: (a), (b) and (c) show the recognition space of three individual classifiers, (d) shows superimposed three recognition spaces, (e) shows the upper bound space. The shaded area in (f) represents the possible recognition space of a CRI algorithm.

by the classifiers being combined. Let the recognition space of a classifier be the set that contain all the correctly recognized samples in the pattern space. The upper bound performance of a CRI algorithm can be measured by the union of the recognition spaces of the classifiers being combined. Figure 8 illustrates this notion. In this example, we assume three classifiers to be combined. The shaded areas in (a) and (b) and (c) represent the recognition spaces of the three classifiers respectively. The three recognition spaces are superimposed in (d). (e) shows the upper bound space, which is the union of the three recognition spaces. It is clear that the recognition space of a CRI algorithm (see Fig. 8(f)) can never exceed this upper bound space and it can be smaller than the recognition space of an individual classier. It is important to point out that the pattern space and recognition spaces are all data dependent. However a CRI algorithm should have its recognition space as close to the upper bound space as possible on any data set, and a CRI algorithm with a larger recognition space is considered better than the ones with smaller recognition spaces.

In order to have a better understanding of the performance of the evidential reasoning algorithm, we have implemented classic Bayesian Formalism [11, 15] and tested it on the same two data sets. In our implementation, the probability tables used in the evidential reasoning were used as the confusion matrix in the Bayesian algorithm. We conducted two experiments to compare the performance of the Bayesian method and the proposed evidential reasoning algorithm, one is on the training data set and another one is on a blind test set. The results of both experiments are shown in Table 4 along with the upper bound of recognition in each data set. In both experiments, the evidential reasoning consistently performed better than any individual classifiers and its performance is close to the upper bound on both data sets. Furthermore, the consistent gain of performance over different data sets shows that the knowledge integration framework under evidential reasoning is robust. In these two experiments, the performance of the Bayesian algorithm is worse than the lowest recognition score of the individual classifiers. The classic Bayesian method has been known for giving poor performance on noisy data [15], and therefore, the poor performance of the Bayesian algorithm could be caused by the noise (see Fig. 7) in both the training and the test set.

Table 4. Experiment results on both training and test sets.

Recognition rate	NN	ST	PL	Bayesian	Evidential reasoning	Upper bound
Training set 388 Zip codes 2716 digits	84%	86%	83%	81%	88%	88.9%
Test set 581 Zip codes 4067 digits	85.75%	86.1%	83.2%	80.9%	88%	89.8%

5. Conclusion

We have presented a knowledge integration framework for a multiple classifier system, described evidence modeling and reasoning algorithms under the framework, discussed about the performance of CRI algorithms, and presented our experimental results.

The modeling process, which converts hypothesis lists to probability lists, is implemented as a machine learning process that generates probability density functions and uncertainty thresholds for each classifier from a training data set containing classification results generated by the classifiers. The reasoning process produces mass function for each individual classifier, generates the composite mass function, and the evidential intervals. Inference rules for generating single result or multiple results are presented. Inference rules for polling evidence directly from evidential intervals are described.

Our framework of evidential reasoning has the following advantages:

(1) The evidence reasoning has no subjective tuning parameters. The only parameter used in the

reasoning process is the uncertainty threshold which is determined automatically during the learning process.

(2) The evidential reasoning process can be quickly adapted to any particular type of data set since the learning process is automatic and requires no manual adjustment.

(3) The evidential reasoning framework allows individual classifiers to

- have their own measurement of confidence values
- represent uncertain, imprecise and inaccurate results including "I don't know".

(4) We argue that the evidential reasoning system produces more certain, accurate and precise results based on the following two facts:

- the evidential reasoning system generates one hypothesis list from **m** input hypothesis lists, where $\mathbf{m} > \mathbf{1}$ is the number of classifiers in the system. Further more the m input hypothesis lists can contain conflict information, and individual hypothesis lists may contain erroneous and less precise information about the input pattern.
- The evidential reasoning system produces better results than any individual classifier on both the training and the test data set.

(5) The implementation of the system is efficient. Since the probability functions are generated offline and stored in tabular format, the reasoning process can use table-lookup to generate mass functions. Furthermore, it can be implemented directly on parallel processors since the combination of mass functions is communicative and associative.

Overall, we have presented a knowledge integration framework based on evidential reasoning, and demonstrated that it is an extremely promising technique in solving uncertainty problems in pattern recognition.

In order to coherently represent beliefs of classification results generated by different classifiers, we obtain a reliability function and the subsequent probability density function through the following supervised learning process.

References

1. P.R. Cohen, "Heuristic reasoning about uncertainty: An artificial intelligence approach," Pitman, 1985.
2. A.P. Dempster, "A generalization of Bayesian inference," *Journal of the Royal Statistical Society*, Series B 30, pp. 205–247, 1968.
3. R. Fagin and J.Y. Halpern, "Uncertainty, Belief, and Probability," *IJCAI*, pp. 1161–1167, 1989.
4. P. Gader, B. Forester, and M. Ganzberger, "Recognition of handwritten digits using template and model matching," *Pattern Recognition*, vol. 24, no. 5, pp. 421–432, 1991.
5. T.D. Garvey, J.D. Lowrance, and M.A. Fischler, "An inference technique for integrating knowledge from disparate sources," *IJCAI*, pp. 319–325, 1981.
6. J. Gordon and E.H. Shortliffe, "A method of managing evidential reasoning in a hierarchical hypothesis space," *Artificial Intelligence*, vol. 26, pp. 3232–3357, 1985.
7. T.K. Ho, J.J. Hull, and S.N. Srihari, "Decision combination in multiple classifier systems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 1, pp. 66–75, 1994.
8. S. Kahan and T. Pavlidis, "On the recognition of printed characters of any font and size," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-9, pp. 274–287, March 1987.
9. Y. Lu, S. Sehlosser, and M. Janeczko, "Fourier descriptors and handwritten digit recognition," *Machine Vision and Applications*, vol. 6, no. 1, pp. 25–34, 1993
10. Y. Lu, "Evidential reasoning in a multiple classification system," The Sixth International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, pp. 476–479, 1993
11. J. Pearl, *Probabilistic reasoning in intelligent systems: networks of plausible inference*, Morgan Kaufmann Publishers: San Mateo, California, 1988.
12. G. Shafer, *A Mathematical Theory of Evidence*, Princeton University Press, 1976.
13. T.M. Strat, "Continuous belief functions for evidential reasoning," *AAAI*, pp. 308–313, 1984.
14. T.R. Thompson, "Parallel formulation of evidential-reasoning theories," *IJCAI*, pp. 321–327, 1985.
15. L. Xu, A. Krzyzak, and C.Y. Suen, "Methods of combining multiple classifiers and their applications to handwriting recognition," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 22, no. 3, pp. 418–435, 1992.
16. L.A. Zadeh, "Fuzzy logics and approximate reasoning," *Synthese*, vol. 30, pp. 407–428, 1975.

Yi Lu graduated with a diploma majored in mathematics from Shanghai Teacher's College, Shanghai, China, in 1980. She received

an M.S. degree in Computer Science from Wayne State University, Detroit, Michigan in 1983 and a Ph.D. degree in Computer, Information, Control Engineering from the University of Michigan, Ann Arbor, Michigan in 1989.

From 1989 to 1992, she was a research scientist at the Environmental Research Institute of Michigan, Ann Arbor, Michigan. Currently she is an assistant professor at the Department of Electrical and Computer Engineering at the University of Michigan-Dearborn. Her research interests include computer vision, pattern recognition and artificial intelligence. She has publications in the areas of low level vision processes, knowledge-based computer vision processes, handwritten digit recognition, machine-printed character segmentation and recognition, medical image analysis, and knowledge integration. Currently, she is an associate editor for Pattern Recognition.