A Survey of Decision Fusion and Feature Fusion Strategies for Pattern Classification

Utthara Gosa Mangai, Suranjana Samanta, Sukhendu Das & Pinaki Roy Chowdhury

To cite this article: Utthara Gosa Mangai, Suranjana Samanta, Sukhendu Das & Pinaki Roy Chowdhury (2010) A Survey of Decision Fusion and Feature Fusion Strategies for Pattern Classification, IETE Technical Review, 27:4, 293-307

To link to this article: https://doi.org/10.4103/0256-4602.64604

Published online: 01 Sep 2014.

Submit your article to this journal

Article views: 697

Citing articles: 6 View citing articles

A Survey of Decision Fusion and Feature Fusion Strategies for Pattern Classification

Utthara Gosa Mangai, Suranjana Samanta, Sukhendu Das and Pinaki Roy Chowdhury[1]

VP Lab, Department of Computer Science and Engineering, Indian Institute of Technology Madras, Chennai – 600 036,
[1]Defence Terrain Research Laboratory, DRDO, Metcalfe House, New Delhi – 110 054, India

Abstract

For any pattern classification task, an increase in data size, number of classes, dimension of the feature space, and interclass separability affect the performance of any classifier. A single classifier is generally unable to handle the wide variability and scalability of the data in any problem domain. Most modern techniques of pattern classification use a combination of classifiers and fuse the decisions provided by the same, often using only a selected set of appropriate features for the task. The problem of selection of a useful set of features and discarding the ones which do not provide class separability are addressed in feature selection and fusion tasks. This paper presents a review of the different techniques and algorithms used in decision fusion and feature fusion strategies, for the task of pattern classification. A survey of the prominent techniques used for decision fusion, feature selection, and fusion techniques has been discussed separately. The different techniques used for fusion have been categorized based on the applicability and methodology adopted for classification. A novel framework has been proposed by us, combining both the concepts of decision fusion and feature fusion to increase the performance of classification. Experiments have been done on three benchmark datasets to prove the robustness of combining feature fusion and decision fusion techniques.

Keywords

Accuracy, Classifier selection, Combining classifiers, Decision fusion, Feature fusion, Feature ranking, Feature selection.

1. Introduction

The ultimate goal of designing pattern recognition systems is to achieve the best possible classification performance for a given problem domain. This objective traditionally led to the development of different classification schemes for any pattern recognition problem to be solved. Studies have been done in the recent past to obtain optimal feature set and classifier set. There are mainly three types of fusion strategies [1], namely, information/data fusion (low-level fusion), feature fusion (intermediate-level fusion), and decision fusion (high-level fusion). Data fusion combines several sources of raw data to produce new raw data that is expected to be more informative and synthetic than the inputs [1]. Decision fusion uses a set of classifiers to provide a better and unbiased result. The classifiers can be of same or different type and can also have same or different feature sets [2]. There are different classifiers such as SVM with various kernels [3], ANN [4], KNN [5], GMM [6], etc., and a single classifier may not be well suited for a particular application. Hence a set of classifiers is used and finally the outputs of all the classifiers are merged together by various methods to obtain the final output. In recent years, decision fusion techniques have been investigated [2,3,7-11] and their application on classification domain has been widely tested. Various classifier combination schemes have been devised and it has been experimentally demonstrated that some of them consistently outperform a single best classifier. However, there is presently inadequate understanding why some combination schemes are better than others and in what circumstances. Most of decision fusion techniques discussed in this paper are taken from Kuncheva's work [2,11] and classifier ranking has been obtained from Ho *et al.* [3].

Methods of feature fusion [12-17] deal with the selection and combination of features to remove redundant and irrelevant features. If two features have similar or nearly similar distribution, one of them is redundant. A feature is said to be irrelevant if it correlates poorly with the class information. The final set of features is fused together to obtain a better feature set, which is given to a classifier to obtain the final result. From the definition itself, it is clear that feature fusion is an advancement of information fusion. Hence we concentrate on decision fusion and feature fusion for better classification accuracy. Areas of application have a wide-range remote sensing [18-21], medical image processing [22], target and object recognition [23], face recognition [24-26], speech processing and video classification and retrieval [27], and gene detection in DNA sequences [28].

In this paper, we first give a detailed survey of decision fusion and feature fusion techniques. Most of these techniques use the databases from the UCI repository [29], Vistex texture [30], speech [31], and medical image [32], for exhibiting their performance. In this paper we have not discussed the results of prior work in detail, as the set of classifiers and databases used widely vary across different methods published in the literature. We then propose a framework which uses both the techniques for a better classification result. We first perform feature fusion on datasets, and finally use the different feature sets for decision fusion using separate classifiers. Results are presented using three benchmark datasets selected from the UCI repository [29].

The rest of the paper is organized as follows. Section 2 gives the justification of decision fusion of multiple classifiers. Section 3 describes various techniques and stages of decision fusion. Section 4 justifies and describes feature fusion techniques. Section 5 describes a novel method of combining decision fusion and feature fusion strategies for improving the accuracy of classification. Section 6 gives the conclusion of the paper.

2. Need for Classifier Fusion

There are several reasons for preferring a multiclassifier system over a single classifier. It is mainly done to improve the accuracy and efficiency of the classification system. Three most important reasons are given below: (a) in certain applications, the volume of data to be analyzed is too large to be handled by a single classifier. Training a classifier with such a vast amount of data is usually not practical. A multiclassifier system will be an efficient approach, where data is partitioned into smaller subsets, trained with different classifiers for different subsets and the outputs are combined. (b) A single classifier cannot perform well when the nature of features is different. Using multiple classifiers with a subset of features may provide a better performance. (c) Another reason for combining classifiers is to improve the generalization performance: a classifier may not perform well for a certain input when it is trained with a limited dataset. Finding a single classifier to work well for all test data is difficult. Instead multiple classifiers can be combined to give a better output than a single classifier. It may not necessarily out-perform a single best classifier, but the accuracy will be on an average better than all the classifiers.

2.1 Different Multiple Classifier Systems

Classifier combinations are results of two parallel lines of study [3]: (i) decision optimization method in which an attempt is made to find an optimal combination of classifiers' decision, given a fixed set of carefully designed and highly specialized classifiers; (ii) coverage op-timization method – generate a set of mutually complementary, generic classifiers that can be combined to achieve optimal accuracy assuming a fixed combination rule. The multiple classifier system can be achieved in one of the following ways:

Variation of initial parameters of the classifiers: A set of classifiers can be created by varying the initial parameters, using which each classifier is trained with the same training data.

Variations of the training dataset of the classifiers: Multiclassifier systems can be built by training the same classifier with different training datasets.

Variations in the number of individual classifiers used: Training different types of classifiers like SVM, ANN, etc., with the same training dataset.

The type of training in the two level scheme (i) training the individual classifier and applying fusion; (ii) training the individual classifier followed by training the fusion.

The hierarchy of "decision fusion" strategies has been suitably categorized by us and shown in Figure 1. There are two types of classifier combination strategies: classifier fusion [33] and classifier selection [34]. In classifier fusion, every classifier is provided with complete information on the feature space, and the outputs from different classifiers are combined. Every classifier contributes to make a final decision whereas in classifier selection methods, every classifier is an expert in a specific domain of the feature space and the local expert alone decides the output of the ensemble. Classifier fusion in Figure 1 is further categorized based on the output of classifiers and classifier selection is classified into dynamic and static classifier selection.

The results of an experimental assessment of the different designs would then be the basis for choosing one of the

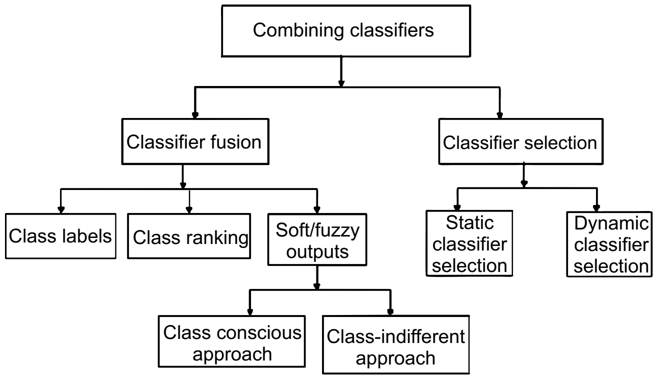

Figure 1: A hierarchy of methods used in combining classifiers, proposed for discussion in this paper.

classifiers as a final solution to the problem. It had been observed in such design studies, that although one of the designs would yield the best performance, the sets of patterns misclassified by the different classifiers would not necessarily overlap. This suggested that different classifier designs potentially offered complementary information about the patterns to be classified which could be harnessed to improve the performance of the selected classifier. These observations have motivated the recent interest in combining classifiers. The idea is not to rely on a single decision-making scheme. Instead, all the designs, or their subset, are used for decision making by combining their individual opinions to derive a consensus decision. We provide a discussion of various methods of classifier combination as given in Figure 1. We believe this categorization is easier for the reader to follow rather than a chronological development of techniques as published in the literature. In the next section, we discuss the different categories of methods used for classifier fusion and selection.

3. Methods of Classifier Fusion and Selection

The possible ways of combining the outputs of the L classifiers in an ensemble depends on the information obtained from the individual classifiers. In the work by Xu *et al.* [7] and Kuncheva *et al.* [33], "information fusion" at the decision level is divided into three categories based on the type of classifier outputs:

Type 1 (the abstract level): each classifier D_i produces a class label $s_{i,i}=1,...,L$. Thus, for any object $x \in \Re^n$ to be classified, the L classifier outputs define a vector $s=[s_1,...,s_L]^T € \Omega^L$ where $\Omega = \{\omega_1,\omega_2,...,\omega_C\}$ is the set of class labels.

Type 2 (the rank level): the output of each D_i is a subset of Ω, with the alternatives ranked in order of plausibility of being the correct label.

Type 3 (the measurement level): each classifier D produces a c-dimensional vector $[d_{i,1},...,d_{i,c}]^T$; the value $d_{i,j}$ represents the support for the hypothesis that vector x submitted for classification comes from class ω_j.

The remaining part of this section consists of three parts, based on the categorization of "classifier fusion" techniques provided by us in Figure 1. The output of a classifier may be a crisp label or probabilistic confidences or ranking of classes in a decreasing order. Subsection 3.1 describes the combination techniques used when the output of classifiers are crisp labels (abstract level). Subsection 3.2 describes the combination techniques used when the outputs of classifiers are a subset of possible matches sorted in a decreasing order of confidences (rank level). Subsection 3.3 describes the combination techniques used when the outputs of classifiers are probabilistic confidence measures obtained over each class (measurement level).

3.1 Combining Class Labels (Crisp Outputs)

The class labels are available from classifier outputs. The decision of the i^{th} classifier is defined as $d_{i,j}$ € 0, 1,i =1, 2,...,L and j =1, 2,...,C (C = number of classes). If the i^{th} classifier chooses class ω_j, then $d_{i,j}$=1, and 0 otherwise. In the following, we describe the four different ways of combining the crisp labels given by different classifiers. Subsection 3.1.1 describes the majority voting method. Subsection 3.1.2 describes the weighted majority voting method by assigning weights to the degree of support given by the classifiers. Subsection 3.1.3 describes the behavior knowledge space (BKS) method and subsection 3.1.4 describes the Naive–Bayes combination.

3.1.1 Majority Voting

In majority voting [8], the ensemble chooses a class when (any one of the situations are considered): (i) all classifiers agree on the specific class (unanimous voting), (ii) predicted by at least one more than half the number of classifiers (simple majority), or (iii) it receives the highest number of votes, whether or not the sum of those votes exceeds 50% (majority voting or plurality voting). The ensemble decision for the majority voting can be described as follows: choose class ω_k,

$$\text{if} \quad \sum_{i=1}^{L} d_{i,k} = \max_{j=1}^{c} \sum_{i=1}^{L} d_{i,j} \tag{1}$$

Majority voting is an optimal combination rule under the minor assumptions: (1) the number of classifiers, L, is odd; (2) the probability for each classifier to give the correct class label is p for any instance x; and (3) the classifier outputs are independent. Plurality voting and simple majority are identical and the ensemble makes the correct decision if at least $[L/2]+1$ classifiers choose the correct label. The probability of ensemble success is

$$P_{majvot} = \sum_{m=\lfloor L/2 \rfloor+1}^{L} \binom{L}{m} p^m (1-p)^{L-m} \tag{2}$$

If p is higher than 0.5, there is a possibility of ensemble to give a better accuracy over the individual accuracy p.

3.1.2 Weighted Majority Voting

If the classifiers in the ensemble do not have identical performance (in general), then more competent classifiers are given greater power to make the final decision [35]. The label (crisp) outputs can be represented as degrees of support for the classes in the following way:

$$d_{ij} = \begin{cases} 1, \text{if } D_i \text{ labels } x \text{ in } w_j \\ 0, \text{otherwise} \end{cases} \tag{3}$$

The discriminant function for class *j* obtained through weighted voting is

$$g_j(x) = \sum_{i=1}^{L} b_i d_{i,j} \tag{4}$$

where b_j is the coefficient (or weight) of classifier D_i. Thus the value of the discriminant function will be the sum of the coefficients for these members of the ensemble whose output for input *x* is class *j*. The classifiers whose decisions are combined through weighted majority voting will choose class ω_k *if*

$$\sum_{t=1}^{L} b_t d_{t,k} = \max_{j=1}^{c} \sum_{t=1}^{L} b_t d_{t,j} \tag{5}$$

Then the accuracy of the ensemble P_{wmaj} is maximized, if voting weights satisfy the proportionality

$$b_t \,\alpha \log \frac{p_t}{1-p_t} \tag{6}$$

where p_t is the accuracy of classifier *t*.

3.1.3 Behavior Knowledge Space

BKS [36] uses a lookup table constructed based on the classification of training data, that keeps information of how often each labeling combination is produced by the classifiers. The label vector *s* gives an index to a cell in a lookup table (*Z*). Each $z_j \in Z$ is placed in the cell indexed by *s* for that object. The numbers of elements in each cell are tallied and the most representative class label is selected for that cell. The highest score corresponds to the highest estimated posterior probability *P* (*k* | *s*). Ties are resolved arbitrarily. The empty cells are labeled in some appropriate way.

3.1.4 Naive–Bayes Combination

This method assumes that the classifiers are mutually independent, given a class label. Let *P* (s_j) be the probability that classifier D_j classifies a sample *x* in class s_j. The conditional independence allows for the following representation [33]:

$$P(s|w_k) = P(s_1, s_2, ..., s_L|w_k) \prod_{i}^{L} P(s_i|w_k) \tag{7}$$

The posterior probability needed to label sample *x* is

$$P(w_k|S) = \frac{P(w_k)P(S|w_k)}{P(S)} \tag{8}$$

Support for class ω_k is calculated as

$$\mu_k(x) \propto P(w_k) \prod_{i}^{L} P(s_i|w_k) \tag{9}$$

For each classifier D_i, a *C**C confusion matrix CM^i is calculated by applying D_i to the training dataset. The (*k,s*) th entry of this matrix $CM^i_{k,s}$ is the number of elements of the dataset whose true class label was ω_s and was assigned by D_i to class ω_s. ˜refequation: support can also be represented as

$$\mu_k(x) \propto (1/N_k^{L-1}) \prod_{i=1}^{L} CM^i_{k,s_i} \tag{10}$$

where $CM^i_{k,s_i} / N_k$ is an estimate of the probability $P(s_i|w_k)$. The total number of sample in a dataset belonging to class ω_s is denoted by N_s. The Naive–Bayes classifier [33] has been found to be surprisingly accurate and efficient in many experimental studies, despite the fact that the entities being combined are seldom independent.

3.2 Class Ranking

The output of each classifier is a subset of possible matches sorted in the decreasing order of confidence. Decisions by the classifiers are represented as rankings of classes so that they can be compared across different types of classifiers and different instances of a problem. There are two main approaches when fusion is applied to the class ranking outputs from multiple classifiers: (i) class set reduction and (ii) class set reordering. Ho *et al.* [3] described three methods to combine the ranks assigned by different matchers. In the highest rank method, each possible match is assigned the highest (minimum) rank as computed by different matchers for class reranking or class set reordering. The Borda count method uses the sum of the ranks assigned by the individual matchers to calculate the combined ranks. The logistic regression method is a generalization of the Borda count method, where the weighted sum of the individual ranks is calculated and the weights are determined by logistic regression. In the following sections, the above two methods are discussed in detail.

3.2.1 Class Set Reduction Methods

The objective of this method is to consider a reduced set of classes by reducing the number of classes as small as possible, but ensuring that the correct class is still represented in the reduced set. The class set reduction methods [3] try to find the trade-off between minimizing the class set and maximizing the probability of inclusion of a true class. This can be done in two ways:

(i) Intersection of neighborhoods: firstly the neighborhoods of all classifiers are determined by the ranks of true classes for the worst case in the training dataset.

The lowest rank given by any classifier is taken as the threshold and only the classes that are ranked above are used for further processing. This method also recognizes the redundant classifiers as the ones for which the thresholds are equal to the class set. This method is applied to classifiers with moderate worst-case performance.

(ii) Union of neighborhoods: the threshold for each classifier is calculated as the maximum (worst) of the minimum (best) ranks of true classes over the training dataset. The redundant classifier is determined easily, since its threshold will be equal to zero which means that its output is incorrect. This method is suitable for classifiers with different type of inputs.

3.2.2 *Class Set Reordering Methods*

The class set reordering method [3] tries to improve the overall rank of the true class. This method is considered to be successful if it ranks the true class higher than any individual classifier. There are three commonly used techniques: highest rank method, Borda count method, and logistic regression.

1. Highest rank method [3]: assume that for each input pattern, *m* classifiers are applied to rank a given set of classes. The minimum of these *m* ranks is assigned to each class and then the classes are sorted according to the new ranks to derive a combined ranking for that input. To determine the individual class as a final decision, the class which is at the top of the reordering ranking is chosen. If there exists a tie in the combined ranking, it may be broken arbitrarily to achieve a strict linear ordering. This method is used when there are few classifiers and a large number of classes. The advantage of this method is that it utilizes the strength of every individual classifier. A major disadvantage is that there may be a possibility of occurrence of ties when combined ranking is done. The number of classes sharing the tie depends on the number of classifiers used. Hence this method is useful only when there is a small number of classifiers.

2. Borda count method [3]: it is defined as a mapping from a set of individual rankings to a combined ranking leading to the most relevant decision. Borda count *Bj* for a particular class *j* is the sum of the number of classes ranked below class *j* by each classifier. This is defined as

$$B_j = \sum_{i=1}^{L} B_i(j) \tag{11}$$

where $B_i(j)$ is the number of classes ranked below class *j* by classifier *i* and *L* is the number of classifiers. The idea behind the Borda count method is based on the assumption of additive independence among the contributing classifiers. The disadvantage of this method is that it treats all classifiers equally and does not take into account the individual classifier capability.

3. Logistic regression [3]: the improvement to Borda count method can be done by assigning the weights to each classifier reflecting their importance in a multiple decision system. Assuming the responses $(x_1, x_2, \ldots, x_m)$ from *m* ($m<l$) classifiers to be the highest for the classes ranked at the top of the ordered list, the logistic response function in [3] is

$$\pi(x) = \frac{\exp(\alpha + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_m x_m)}{1 + \exp(\alpha + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_m x_m)} \tag{12}$$

where, $\alpha, \beta_1, \beta_2, \ldots, \beta_m$ are constants. The output of the transformation

$$L(x) = \log \frac{\pi(x)}{(1-\pi(x))} = (\alpha + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_m x_m) \tag{13}$$

is referred to as "logit." This is treated as a confidence measure and it is possible to determine a threshold value so that classes with a confidence value below it are rejected.

3.3 Combining Probabilistic (Soft) Outputs

The continuous output provided by a classifier for a given class is interpreted as the degree of support given to that class, and it is usually accepted as an estimate of the posterior probability for that class. In the work by [9], a decision profile matrix *DP* (*x*) is used for decision combination. This is described in the following text.

In a traditional multiple classifier system, a feature vector *x* is classified into one of the *C* classes using *L* classifiers, where $\Omega = \omega_1, \omega_2, \ldots, \omega_c$ is the set of class labels. Each classifier D_i in the ensemble outputs *c* degrees in the interval [0,1]. The support that classifier D_i provides for the class label ω_j is $d_{i,j}(x)$. The larger the support, the more the likelihood of the sample getting assigned to class label ω_j. The output of *L* classifiers for a particular input *x* is organized as a decision profile (*DP* (*x*)), in the form of a matrix as follows:

$$\begin{bmatrix} d_{1,1}(x) \ldots d_{1,j}(x) \ldots d_{1,c}(x) \\ d_{i,1}(x) \ldots d_{i,j}(x) \ldots d_{i,c}(x) \\ d_{L,1}(x) \ldots d_{L,j}(x) \ldots d_{L,c}(x) \end{bmatrix} \tag{14}$$

Some fusion techniques known as class-conscious [9] techniques do column-wise class-by-class operation on the *DP* (*x*) matrix to obtain $\mu(x)$. Examples of these types of fusion techniques are sum, product, minimum, maximum, etc. [37]. Another fusion approach known as class indifferent [9] uses entire *DP* (*x*) to calculate $\mu(x)$.

3.3.1 Class-conscious Methods

Given $DP\,(x)$, class-conscious methods [9] operate class-wise on each column of $DP\,(x)$. The rules are described as follows:
Sum rule: It computes the soft class label vectors using

$$\mu_j(x) = \sum_{i=1}^{L} d_{i,j,} j = 1,2,\ldots,C \qquad (15)$$

Product rule: It computes the soft class label vectors as

$$\mu_j(x) = \prod_{i=1}^{L} d_{i,j,} j = 1,2,\ldots,C \qquad (16)$$

Min. rule: It computes the soft class label vectors using

$$\mu_j(x) = \min_{i=1}^{L} d_{i,j}, j = 1,2,\ldots,C \qquad (17)$$

Max. rule: It computes the soft class label vectors using

$$\mu_j(x) = \min_{i=1}^{L} d_{i,j}, j = 1,2,\ldots,C \qquad (18)$$

3.3.2 Class-Indifferent Methods

Given $DP(x)$, class-indifferent approach uses whole of the DP disregarding the classes. There are two different methods: decision templates and Dempster–Shafer combination.

1. Decision templates

The idea of decision template (DT) combiner [2] is to remember the most typical decision profile for each class ω_j called the decision template, DT_j, which is then compared with the decision profile $DP\,(x)$, using a similarity measure. The closest match will label x. Two measures of similarity used are as follows:

- Squared Euclidean distance (DT(E)): the ensemble support for ω_j is

$$\mu_j(x) = 1 - \frac{1}{L*C}\sum_{i=1}^{L}\sum_{k=1}^{C}[DT_j(i,k) - d_{i,k}(x)]^2 \qquad (19)$$

where $DT_j(i,k)$ is the (i,k)th entry in decision template DT_j.

- Symmetric difference ($DT(S)$): the support for ω_j is

$$\mu_j(x) = 1 - \frac{1}{L*C}\sum_{i=1}^{L}\sum_{k=1}^{C} max[min\{DT_j(i,k),(1-d_{i,k}(x))\}, \\ min\{(1-d_{ij}(i,k)), d_{i,k}\}]$$

2. Dempster–shafer combination [33,38] is described as follows:

- Let $DT^{\,i}$ denote the ith row of decision template DTj. $D_i(x)$ represents the output of D_i. Proximity Φ between $DT_j^{\,i}$ and the output of classifier D_i for the input x is defined as

$$\Phi_{j,i}(x) = \frac{(1+\left\|DT_j^i - D_i(x)\right\|^2)^{-1}}{\sum_{k=1}^{c}(1+\left\|DT_j^i - D_i\right\|^2)^{-1}} \qquad (20)$$

where $\|\,\|$ is any matrix norm. For each decision template, there are L proximities.

- Belief degrees is calculated for every class j=1,...,C and for every classifier i =1, 2,...,L, as

$$b_j(D_i(x)) = \frac{\Phi_{j,i}(x)\prod_{k\neq j}(1-\Phi_{k,i}(x))}{1-\Phi_{j,i}(x)[1-\prod_{k\neq j}(1-\Phi_{k,i}(x))]} \qquad (21)$$

The final degree of support is

$$\mu_j(x) = K\prod_{i=1}^{L} b_j(D_i(x)) j = 1,\ 2,\ \ldots,\ c \qquad (22)$$

where K is a normalizing constant.

3.4 Classifier Selection

In the process of selection-based method, a single classifier is used to correctly classify the input pattern. Exactly one classifier is nominated to make the decision. There are two classification selection methods based on the phase in which the label is assigned to each classifier. As in classifier fusion, these methods do not assume that the classifiers are independent. They are static classifier selection and dynamic classifier selection method. In the static classifier selection method, during the training phase, the selection regions are specified prior to classifying the unlabeled vector x. In the operation phase, the region of x is first found, for example, R_j, and processed further by the respective classifier $D_{i(j)}$, responsible for region R_j. The training phase can be done in two ways: (i) specify the region and then assign a responsible classifier for each region [3], or (ii) given the set of classifiers, find a region where each classifier is the best one. Second approach is more efficient, but difficult to implement. On the other hand, in a dynamic classifier selection method, the choice of the classifier to label x is made during the operation phase. This choice is typically based on the certainty of the current decision. Preference is given to more certain classifiers. The competence of each classifier is estimated in the vicinity of x as the classifier's accuracy. The classifier with the highest competence is authorized to label x. Kuncheva *et al.* [34] presented a combination of classifier selection and fusion by using statistical inference to switch between the two by applying classifier selection in those regions of the feature space where one classifier strongly dominates the others from the pool and fusion is applied in the remaining regions.

4. Methods of Feature Fusion

In spite of the power demonstrated by several decision fusion strategies, features are the most vital and important cues (information) for any successful classification or clustering task. Features which are noisy, overlapping (based on class-wise distribution), distorted, and confusing produce inhibited performance for any classifier. Hence the selection of a proper set of features is vital for any successful operation of a single or multiple classifier(s) system. It is necessary to identify the rich set of valuable features and filter the undesired ones which confuse the classifier and produce worse performance.

Features (in pattern recognition, features are the individual measurable heuristic properties of the phenomena being observed. Choosing discriminating and independent features is key to any pattern recognition algorithm to be successful in classification (Source: Wikipedia '09)) play a very important role for any pattern classification task. A set of bad features can deteriorate the performance of a good classifier. With the increase in noise and dimensionality, feature selection and fusion becomes an essential step. A feature that is having too much of confusing (contradictory) information than the rest of the set should be avoided as these features confuse the classifiers. To reduce the noise in the data, features which are weakly correlated to the class information should be removed. When class information is missing, generally a clustering algorithm is used, where scatter or compactness of the cluster formed with and without a particular feature are compared to decide if the particular feature is redundant or not.

"Curse of dimensionality" is also a big motivation for feature selection. Too many features increase the computational time without any significant change in the performance during the testing phase. Over the past few decades, a lot of work has been done on feature selection though most of these algorithms assume that the selection criterion is monotonic in nature. This is a serious drawback, as in reality gradual addition or deletion of a feature from a set does not always change the criterion function value monotonically. The fusion of different features is a step to generate a new feature set from the selected set of features. It is still not clear how to combine the features in the best possible manner, and how to use the unselected features so that they can provide better classification results together.

The main aim of feature fusion is feature reduction and removal of noisy features. Also it can merge two or more features of different domains and reduce the "curse of dimensionality" problem. The main essence of feature fusion lies in the feature selection process. A lot of work has been done on feature selection [12,13], but only a few have been reported for the task of feature fusion. Feature fusion [16,17] can be divided into the following methods, namely, feature ranking, selection, feature extraction, and feature combination. Ranking of features is a generalization of the feature selection. In feature selection, we obtain a rank-ordered set having less number of features, while in feature ranking we obtain a rank-ordered set of all the features present in the original dataset. The flow-chart given in Figure 2 presents a hierarchical organization of feature fusion methods. The following subsections present a detailed review of these fusion techniques based on this flow-chart [Figure 2].

4.1 Feature Selection

Feature selection is the main essence of feature fusion. Feature selection is a well-known problem and has been much explored specially in the areas of data min-

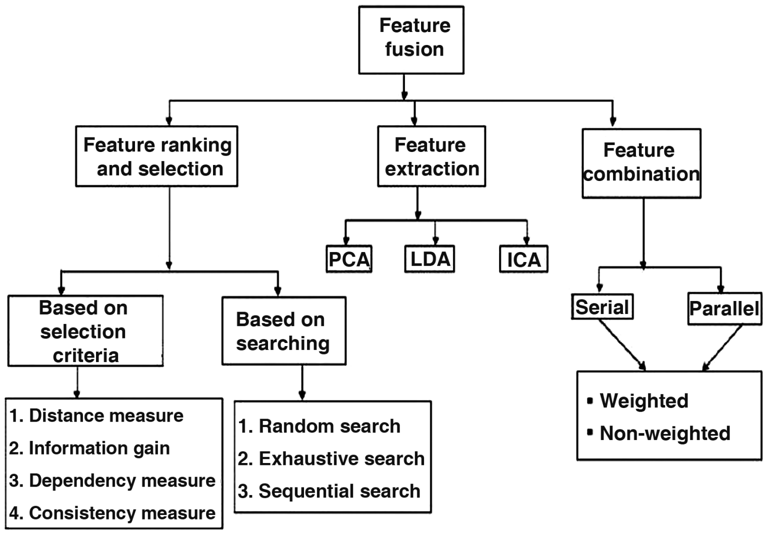

Figure 2: A hierarchical categorization of the different methods for feature fusion, built using those given in [12,13,17,39].

ing and CBIR (content-based image retrieval). The problem deals with finding an appropriate (suitable or discriminatory) subset of features from a given set, for some particular application domain, to improve the accuracy. This involves finding a minimal subset that represents the whole set, or to rank the features based on their importance, from the overall set. Feature selection algorithm can be divided into three types, namely, filter method [12,13], wrapper method [12,13], and hybrid method [13]. In the filter approach, the feature set is evaluated at once which is independent of any clustering algorithm or classifier. On the other hand, the wrapper method calls the clustering algorithm or the classifier for each subset evaluation to find the final subset. While the filter method is unbiased and fast, the wrapper method gives better results for a particular clustering algorithm or classifier. Hybrid method is a fusion of both filter and wrapper methods.

A number of feature selection methods have been proposed over the last few decades. The selection method mainly depends on the type of learning (supervised or unsupervised) and the algorithm used. Most of the feature selection (evaluation function) methods can be used in all types of algorithms (filter or wrapper), while there are a few which are exclusively for wrapper method or filter method. Four steps describe the general feature selection framework, namely, subset generation, subset evaluation, stopping criteria, and result validation. In the case of supervised learning, one can use a validation set to check the performance or validity of the final set obtained. The framework for feature selection (filter and wrapper method) has been illustrated in Figures 3 and 4. In the following sections, we give detailed descriptions of the different steps of feature selection, as given in Figures 3 and 4.

4.1.1 Subset Generation

Subset generation is essentially a search algorithm used for selecting a subset of features from the original set of features. In an original feature set on "*N*" dimension, there are 2^N subsets. Exhaustive searching of all the subsets is a time-consuming process. Hence, people have come up with various optimal and sub-optimal search techniques. The different algorithms used for subset generation [13] are described in the following:

4.1.1.1 Complete search

The simplest complete search technique is a method of exhaustive searching on the entire feature space. Schlimmer [40] says that complete search technique is not exhaustive all the time. Back-tracking can be used in various algorithms like branch and bound, beam search, and best first search techniques [41].

4.1.1.2 Sequential search

This is a very popular search technique mainly because of its simplicity and low time complexity. For a feature set of *N* dimension, the time complexity is $O(N^2)$. Greedy approaches like sequential forward search (SFS), sequential backward search (SBS), sequential floating forward search (SFFS), and sequential floating backward search (SFBS) have been widely used. Alternative versions of these algorithms such as take out *p* features at a time or add *q* features at a time or PTA(*p*,*q*) (plus *p*, take away *q* at a time) [41] have also been used.

4.1.1.3 Random

This type of algorithm imparts randomness into the existing algorithms. One way is to start with a random set and follow greedy approaches like SFS or SBS. Another approach is to find the next subset again in a random manner. More the number of iterations, better the final result. Some examples of random algorithms are simulated annealing, random start hill climbing, and Las Vegas algorithm [42].

4.1.2 Feature Selection Evaluation Function

The subset evaluation criteria depend on whether we

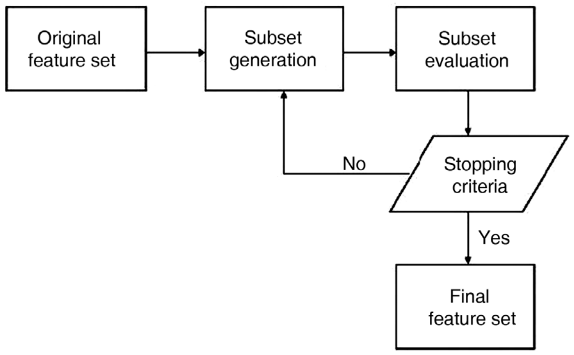

Figure 3: The filter method for feature selection [12].

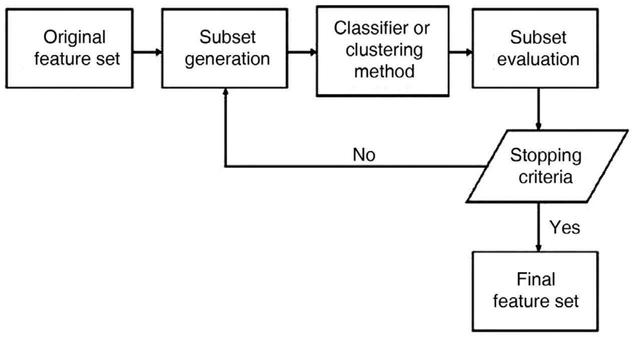

Figure 4: The wrapper method for feature selection [12].

are dealing with a supervised or unsupervised learning or whether we are using a filter or wrapper method. Liu *et al*. [13] have broadly classified all the subset evaluation criteria into four types, namely, distance measure, information gain measure, dependency measure, and consistency measure. In the following section, we give a brief introduction of each of these different techniques.

4.1.2.1 Distance measure

A lot of distance measures like Euclidean distance or Bhattacharya distance [4] have been used for feature selection problems. Bhattacharya distance provides an upper bound of the Bayesian probability error. Less the Bhattacharya distance for a feature, more it is preferred. Another popular distance measure criterion is a linear function of interclass and intraclass distances. Distance measure criteria are very easy to implement for supervised learning, while for unsupervised learning we have to find out the number of most suitable cluster for each subset. A distance measure criterion using "within class scatter" matrix and "between class scatter" matrix has been used widely. Different distance measure criteria for feature selection have been described in [43-48].

4.1.2.2 Information gain

The main idea of information gain criteria is to find out how much unique information is contributed by a feature to the entire feature set. Information gain of a feature f can be calculated as $F(S \cup f) - F(S)$, where $F(.)$ is the evaluation criterion and S is the already selected subset of features. Feature f_1 is preferred to feature f_2 if the information gain of f_1 is greater than that of f_2. Bayes error rate, conditional probability, and information gain are some of the information gain criteria. Information gain criteria have been used in [49, 50, 51].

4.1.2.3 Dependency measure

This type of criteria measures the dependency between two pairs of feature subsets. Mostly, similarity measurement functions like entropy, correlation, KL divergence, and Cohen's Kappa [52] are used. In the case of supervised learning, a feature f_1 is preferred to f_2 if the dependency between f_1 and class label C is greater than that of f_2 and C. In the case of unsupervised learning, if f is already a selected feature then f_1 is preferred to f_2 if the dependency between f_1 and f is less than that of f_2 and f, since f_1 is more redundant than f_2 for the already selected feature subset. Some of the feature selection algorithms using dependency measures have been mentioned in [52–56].

4.1.2.4 Consistency measure

The consistency measure aims to find the minimum number of features that can describe the whole feature set. It is purely a supervised learning. A pair of instances having same value but different class labels are said to be inconsistent. Let c_i be the number of samples in the ith class, where i varies from 1 to c. If the mth class has the maximum number of instances, i.e., c_m, then the inconsistency count is given by $(N - c_m)$, for which i varies from 1 to c. The inconsistency rate is the sum of all inconsistency counts divided by the total number of instances. If the inconsistency rate is less than a predefined threshold, the subset is selected, or else further processing is done to find the minimal set of features. Liu *et al*. [42] described about the Las Vegas algorithm that uses the consistency measure along with random search.

4.1.3 Stopping Criteria

A proper stopping criterion is needed for the algorithm to determine when the optimal solution is reached. Typically, a stopping criterion varies with the kind of output needed.

Some of the stopping criteria described in [13] are
1. The search is complete (applicable for a complete searching algorithm).
2. The best M features are obtained where M is a given number.
3. Addition or deletion of any more features does not improve the result or bring down the accuracy of the feature set.
4. The accuracy (or error in output) obtained by the selected feature set is below (or more than) a threshold value, which is predetermined.

4.1.4 Result Validation

This step is mainly applicable for supervised learning. Once the best feature set is obtained using a selection criterion (function), an accuracy measure can be used to validate the feature set. In some cases, choosing a wrong selection criterion function can lead to a false validation result. For example, the minimization of the error rate is a better selection criterion than measuring the cluster spread with entropy, in the case of obtaining a feature set for a classification problem. Hence, the validation step not only ensures that the selected feature gives better performance than the original set, but it also can find the most suitable selection criterion function for a particular feature set.

4.1.5 Examples of Result Validation

In this section, we discuss some existing technologies for better understanding of the above-mentioned techniques. Mark Hall [57] has used a supervised learning technique for feature subset selection. Pearson's correlation has been used for feature selection, which is given by

$$Merit_s = \frac{k \times \overline{r_{cf}}}{\sqrt{k + k(k-1)\overline{r_{ff}}}} \tag{23}$$

where *Merits* is the merit of a feature subset containing *k* features, *rcf* is the average feature class correlation, and $\overline{r_{ff}}$ is called the feature–feature correlation. For finding the correlation, Hall [57] used two equations based on the type of data. For discrete class data, the correlation is calculated by the symmetric uncertainty (SU) factor, which is given by

$$SU = 2.0 \times \left[\frac{H(X)+H(Y)-H(X,Y)}{H(X)+H(Y)} \right] \tag{24}$$

where $H(X)$ is the entropy of the feature X and $H(X,Y)$ is the cross-entropy of two discrete features X and Y. For continuous class data, the correlation is calculated by

$$r_{XY} = \frac{\sum xy}{n\sigma_x\sigma_y} \tag{25}$$

where x and y are the two continuous variables and n is the number of instances in X and Y. σ_x and σ_y are the standard deviations of the variables x and y, respectively. Finally, a best search algorithm has been used for searching. If a new expanded subset does not improve the result, the old subset is restored. Failing to improve the result by expansion for five consecutive times is the stopping criterion.

Mitra and Pal [58] used an unsupervised clustering algorithm for feature selection. Maximal information compression index has been used for feature evaluation, which is given by

$$\lambda_2(x,y) = \frac{var(x)+var(y)-\sqrt{\dfrac{(var(x)+var(y))^2-4var(x)}{var(y)(1-\rho(x,y)^2)}}}{2}$$

where $var(x)$ and $var(y)$ are the variances of x and y, respectively. Higher the value of $\lambda_2(x,y)$, more dissimilar are the features. The final number (k) of features that are required is provided by the user. The feature which is most similar to its k neighbors is considered and those k neighbors are then discarded.

Guo *et al.* [59] used a supervised algorithm for feature selection. The first feature is selected as

$$x_1^0 = max_i I(x_i, y) \tag{26}$$

where y is the class label, x_i is the ith feature vector, and x_k^0 is the result of maximization at step k. $I(X,Y)$ is the mutual information which is given by

$$I(X,Y) = H(X) - H(X/Y) \tag{27}$$

where $H(X)$ and $H(Y)$ are the entropy of X and Y feature spaces, respectively, and $H(X/Y)$ is the conditional entropy of feature X given feature Y. The second feature is selected as

$$x_n^0 = max_{i\neq 1}\left[I(x_i,y) - \sum_{i\neq 1} I(x_i, x_1^0) + \sum_{i\neq 1} I(x_i, x_1^0 \,|y) \right] \tag{28}$$

The rest of the features are evaluated as

$$x_n^0 = max_{i\neq j}\left[I(x_i,y) - \sum_{j}\sum_{i\neq j} \beta I(xi, x_j^0) \right] \tag{29}$$

where $0 \le \beta \le 1$. A greedy approach has been used for searching the appropriate set of features. Finally, the algorithm stops when a predefined number of features get selected.

4.2 Feature Extraction

In this case, the whole feature space is projected into another dimensional space for a better analysis of the features [60,61]. In the reduced dimension space, the scatter of the clusters, with or without the class information, is observed to rank the reduced number of features. The number of dimensions can be reduced by principal component analysis (PCA) [48,17], kernel PCA (KPCA) [62], linear discriminant analysis (LDA), and independent component analysis (IDA) [63] or can be increased as in CCA (canonical correlation analysis (CCA) [64, 65]. The projection has to be in the direction where the maximum class separability is obtained. In the case of unsupervised learning (PCA), the aim is to maximize the cluster separability. These methods are traditionally used not only for dimensionality reduction [48, 17], but also for feature selection when computational cost is an important issue. In the case of supervised analysis (LDA), a criterion (Fisher [39]) maximizes the interclass separability, while the intraclass spread is minimized. In ICA, the condition of orthogonality is relaxed to obtain a feature set of nonprincipal components, for projection onto a new dimension. Application of PCA, LDA, ICA, etc., span over face recognition, object recognition, content-based image and video retrieval and image (texture) segmentation. Success mainly depends on the dataset, criteria, and classifier used.

4.3 Feature Combination

Once the best feature subset is obtained from the original set, we can use the derived set or can derive a new feature based on two or more of the selected features for the task of classification. Based on this concept, there are two existing techniques of feature combination: serial and parallel combination [17]. The following sections will describe these combination techniques and the preprocessing step for combination.

4.3.1 Preprocessing – Calculation of Weights

Not all the features in the selected feature set are useful or have same importance. Hence, weights can be used to increase the influence of a good feature and to decrease the influence of relatively bad features. One way to calculate the weights is to set them equal to normalize the features so that each of them has equal contribution to the final set [61]. Weights can be calculated from the rank of the features which is set equal to the order in which each of them is retrieved from the original dataset during the selection process. A distance-based measure between a feature and cluster center can also be used for calculating the weightage [58]. In other cases, the ratio of number of features in two feature sets can be used for weightage calculation [17]. In most of the cases, weightage is adjusted manually or all the features are given same weightage.

4.3.2 Serial and Parallel Combination

This is the general technique where two features α and β are concatenated together [17]. If m and n are the weights of α and β, respectively, then according to the serial fusion the combined feature is $[m \times \alpha; n \times \beta]$. In parallel combination, a complex variable is used to combine the two features into a complex feature. The absolute value of the complex feature is taken as the final feature. Hence, if m and n are the weights of α and β, respectively, then the combined feature is set as $\|(m\alpha)+ i(n\beta)\|$. In both these cases, the weights m and n can be decimal or binary values. In the latter case, 0 as a weight for a particular feature denotes that the corresponding feature is discarded, while 1 denotes that the corresponding feature is selected for the final subset of features. It has been shown by experiments in [17] that a parallel combination produces better results than serial combination of features.

We have so far presented methods of feature selection, ranking, and fusion in this section. Most of these works in the published literature (including those for decision fusion) use benchmark dataset, from the UCI repository [29], or real-world datasets (texture images [66], speech [31], seismograph, medical image [32], etc.) for performance analysis.

In the following section we present a novel method which combines the advantages of both the fusion techniques (decision fusion and feature fusion) for pattern classification, and prove the same using experimental results.

5. Proposed Framework and Results

Motivated by the methods and performance of various decision fusion and feature fusion strategies, presented in the two previous sections, we design a framework which integrates these two methodologies. We believe that an integration of both these fusion techniques should provide, in general, a powerful mechanism for data classification. This method is novel in the sense that we have not been able to identify any prior work which combines these mechanisms of fusion for building an integrated framework to exploit the advantages of this pair of strategies.

We propose a framework for combining decision and feature fusion, as shown in Figure 5. Features of a sample are rank ordered and the feature set which provides

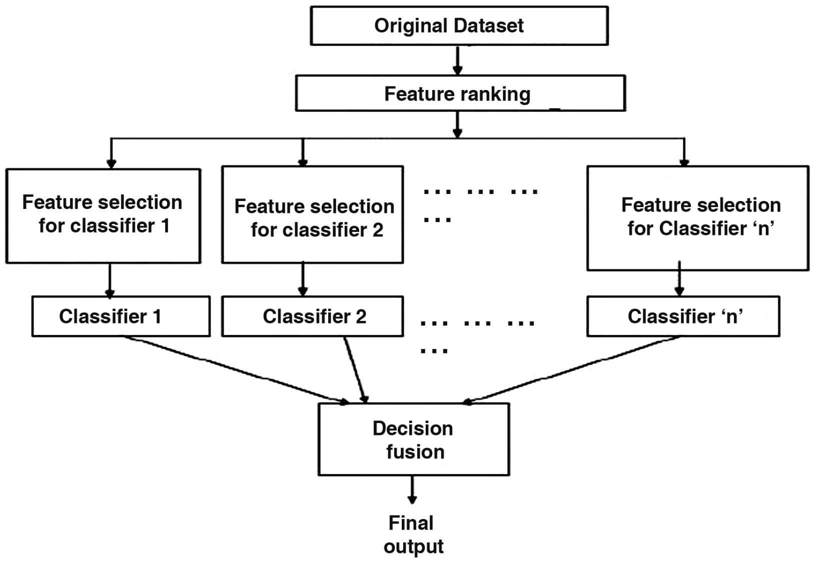
Original Dataset
Feature ranking
Feature selection for classifier 1
Feature selection for classifier 2
Feature selection for Classifier 'n'
Classifier 1
Classifier 2
Classifier 'n'
Decision fusion
Final output

Figure 5: Proposed framework for combining feature fusion and decision fusion (n = 3, in our case).

the best classification accuracy is selected. This is done separately for each classifier and the selected features are used for training the corresponding classifiers. Each classifier gives a confidence measure which is then combined using a sum rule [9]. Results are verified using three benchmark datasets from the UCI repository [29].

One-third of the total dataset is used as the training data and the rest two-thirds have been used for testing. At first, we rank the features using the feature ranking method described in [59]. For this purpose we again divide the training dataset into two parts, where half of the training data is used for feature ranking and the other half (of the training data) has been used as a validation set. Once the ranking has been done, we classify the validation dataset with the increasing number of the rank-ordered features. Three classifiers, SVM with a linear kernel (SVM-L), SVM with a polynomial kernel of degree 2 (SVM-P), and fuzzy K nearest neighbor (F-KNN), have been used for the classification purpose. We observe the change in accuracy with the increasing number of rank-ordered features. We obtain an optimal number of rank-ordered features, say *M*, for which the accuracy observed is maximum. We select the first top *M* rank-ordered features as the final set of features. The value of *M* is not identical for all classifiers (also depends on the dataset used). We hence obtain nine (9) different combinations (sets) of *M* features, for each different combination of classifier (3) and dataset (3) used for experimentation. Each respective value of *M* for a classifier–dataset combination is obtained after averaging over 25 trials of the feature ranking technique [59]. Each dataset is then reorganized to form three selected subsets of *M* rank-ordered features and then given to each of the respective classifiers (3). The final decision is obtained using decision fusion (sum rule). Experimentation is done over different datasets available in the UCI repository [29]. Experiments are realized in four different ways: (i) classifying the dataset with three classifiers; (ii) combining the confidence given by classifiers using decision fusion technique; (iii) using feature selection method to select the features and then classifying it using the three classifiers; and (iv) finally [Figure 5], combining the confidences given by classifiers trained with the features, which were selected using the feature selection method.

Table 1: Description of the datasets (number of classes = 2, in each case)

Dataset	Dimension	Number of samples	
Heart	13	N1	150
		N2	120
Ionosphere	34	N1	225
		N2	126
Lymphography	18	N1	81
		N2	61

First, we have shown the increase in the accuracy using decision fusion and feature fusion techniques separately. We have taken the heart, ionosphere, and lymphography datasets from the UCI repository [29]. The original lymphography dataset contains four classes where the numbers of instances in each of the classes are 81, 61, 4, and 2. We neglect the last two classes, as the number of instances is very small and convert the dataset into a two-class problem. SVM-L, SVM-P, and F-KNN have been used as classifiers for this purpose.

For decision fusion we have used the sum rule [9] for fusion. For feature selection and fusion, we have used the algorithm proposed by Guo *et al*. [59]. Table 1 describes the datasets used, number of features, classes, and number of instances under each class. N1 and N2 are the number of samples available for class 1 and class 2. Table 2 shows the classification accuracy obtained before and after decision fusion. The contribution of each classifier is verified by combining the remaining two classifiers. Results show that generally the F-KNN classifier along with SVM-L performs better than any (dual) other combination. The complementary nature of these two classifiers has improved the classification accuracy. It is interesting to note that SVM-L gives better accuracy, but performance does not improve when combined with SVM-P. This is because SVM-P and SVM-L have similar characteristics. They are noncomplementary in nature and were observed to fail with the same/similar set of examples. Results also show that a combination of only two classifiers does not always perform better, while in some cases the performance of fusion of all three is less than the combination of two classifiers. It (see the last column of Table 2) however gives better results on an average, if not always the best. In-creasing the number of classifiers should improve the result of decision fusion strategy (sum rule).

Table 3 shows the classification accuracy obtained before

Table 2: Accuracy before and after decision fusion (maximum accuracy in each row, is identified in bold)

Dataset	Accuracy %						
	Before decision fusion			After decision fusion (sum rule)			
	SVM-P	SVM-L	F-KNN	SVM-P + SVM-L	SVM-L + F-KNN	SVM-P + F-KNN	All three
Heart	73.32	82.75	63.35	72.05	82.22	80.11	80.55
Ionosphere	86.72	84.97	81.30	86.83	84.57	86.68	86.95
Lymphography	77.76	81.05	77.29	79.26	81.78	79.47	81.68

Table 3: Accuracy before and after feature fusion (maximum accuracy in each row is identified in bold)

Dataset	Accuracy %					
	Before feature fusion			After feature fusion		
	SVM-P	SVM-L	F-KNN	SVM-P	SVM-L	F-KNN
Heart	73.32	82.75	63.35	75.39	83	62.61
Ionosphere	86.72	84.97	81.30	90.65	85.04	82.61
Lymphography	77.76	81.05	77.29	77.66	82.34	78.94

Table 4: Improvement in accuracy (%) using our proposed framework [Figure 5]

Dataset	Improvement in accuracy (%)		
	Decision fusion only	Feature fusion only	Our proposed framework
Heart	7.41	0.52	7.87
Ionosphere	2.62	1.77	3.47
Lymphography	2.98	0.98	5.18

and after feature fusion [59]. The improvement in the accuracy is not as significant as in decision fusion and also depends on the dataset–classifier combination used for classification. We also observed that removing any one or two of the features from the dataset does not affect the performance, in general, as the dimension of the feature space (*K*) is high (compared to the number of classifiers, *C*) in the databases used by us, as well as those commonly used in the literature. In all the three datasets, we have considered for analysis, the dimension is high (13–34). None of them contain any superior feature which is predominantly more significant than the others. This is the reason why we do not observe any significant change in the accuracy of classification with the remaining set of features.

Table 4 shows the improvement in the classification accuracy of our proposed method by combining decision fusion and feature fusion, as shown in Figure 5. All results are obtained by averaging over a 25-fold iteration, where one-third of the dataset was for training and the rest for testing. The amount of improvement in the accuracy for decision fusion [Table 4] is calculated from Table 2, in which the average of accuracies obtained by three individual classifiers is subtracted from the accuracy obtained by fusing all three classifiers. The amount of improvement in the case of feature fusion is obtained from Table 3, by subtracting the mean of accuracies obtained from three individual classifiers from the mean of accuracies obtained after feature fusion. The last column in Table 4 is obtained by applying decision fusion over the classifier outputs obtained using the features selected by the feature fusion method [Figure 5]. Results in Table 4 show that the proposed framework performs always better than that using only decision fusion or feature fusion alone. This exhibits the power of our proposed method of integration of both these methods of fusion.

6. Conclusion

We have categorically presented a detailed survey of the prominent methods of decision fusion and feature fusion published in the literature. In the case of decision fusion, various techniques have been discussed to combine decisions provided by multiple classifiers (hetero- or homogeneous). Classifier selection, weighted combination, and class-indifferent methods have also been discussed. In the case of feature fusion, methods of feature selection, ranking, and feature combination have been presented. In both the cases of fusion, it was difficult to identify the best or the best set of methods for a particular task. The performance depends on the dataset and the application domain. We hence presented a unified framework to combine the advantages of both these fusion techniques, which has a rich potential and scope of application.

Decision fusion and feature fusion is an ever-growing field, with a wide scope of interdisciplinary research over the fields of computer science, mathematics, statistics, and machine learning. In the future, one can expect rich concepts from widely varying areas such as information theory, optimization theory, rough-fuzzy sets, soft computing, evolutionary computation, etc., to contribute and enrich this problem domain in the field of pattern recognition.

References

1. B. V. Dasarathy, *Decision Fusion*, Computer Society Press, 1994.
2. L. I. Kuncheva, J. C. Bezdek, and R. P. W. Duin, "Decision templates for multiple classifier fusion: an experimental comparison," Pattern Recognition, vol. 34, pp. 299- 314, February 2001.
3. T. K. Ho, J. J. Hull, and S. N. Srihari, "Decision in multiple classifier systems," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 16, pp. 66-75, January 1994.
4. C. M. Bishop, "Neural Networks for Pattern Recognition", Oxford University Press, 1995.
5. B. V. Dasarathy, "Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques", IEEE Computer Society Press, 1995.
6. A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," Journal of the Royal Statistical Society, vol. B 39, pp. 1-38, January 1977.
7. 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, pp. 418-435, March 1992.
8. J. V. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, "On combining classifiers," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 20, pp. 226-239, March 1998.
9. L. Lam and C. Y. Suen, "Optimal combinations of pattern classifiers," Pattern Recog-nition Letters, vol. 16, pp. 945-954, September 1995.
10. K. Woods, W. P. Kegelmeyer, and K. Bowyer, "Combination of multiple classifiers using local accuracy estimates," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 19, pp. 405-410, April 1997.

11. L. I. Kuncheva and C. J. Whitaker, "Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy," Machine Learning, vol. 51, pp. 181-207, May 2003.
12. M. Dash and H. Liu, "Feature selection for classification," Intelligent Data Analysis, vol. 1, pp. 131-156, August 1997.
13. H. Liu and L. Yu, "Toward integrating feature selection algorithms for classification and clustering," IEEE Trans. Knowledge and Data Engineering, vol. 17, pp. 491-502, April 2005.
14. S. Samanta and S. Das, "A Fast Supervised Method of Feature Ranking and Selection for Pattern classification," in Accepted in Third International Conference on Pattern Recognition and Machine Intelligence (PReMi'09), LNCS Publication, Delhi, Dec. 2009.
15. S. Samanta and S. Das, "Unsupervised Texture Segmentation using Feature Selection and Fusion," in *Accepted in IEEE International Conference on Image Processing (ICIP'09)*, Cairo, Nov. 2009.
16. Q. S. Sun, S. G. Zeng, Y. Liu, P. A. Heng, and D. S. Xia, "A new method of feature fusion and its application in image recognition," Pattern Recognition, vol. 38, pp. 2437- 48, December 2005.
17. J. Yang, J. Y. Yang, D. Zhang, and J. Lu, "Feature fusion: parallel strategy vs. serial strategy," Pattern Recognition, vol. 36, pp. 1369-81, June 2003.
18. G. J. Briem, J. A. Benediktsson, and J. R. Sveinsson, "Multiple classifiers applied to multisource remote sensing data," IEEE Trans. Geoscience and Remote Sensing, vol. 40, pp. 2291-99, October 2002.
19. P. C. Smits, "Multiple classifier systems for supervised remote sensing image classification based on dynamic classifier selection," IEEE Trans. Geoscience and Remote Sensing, vol. 40, pp. 801-823, April 2002.
20. V. D. Linden and B. S. Waske, "Classifying multilevel imagery from SAR and optical sensors by decision fusion," IEEE Trans. Geoscience and Remote Sensing, vol. 46, pp. 1457-66, May 2008.
21. S. Prasad and L. M. Bruce, "Decision fusion with confidence-based weight assignment for hyperspectral target recognition," IEEE Trans. Geoscience and Remote Sensing, vol. 46, pp. 1448-56, May 2008.
22. A. S. Constantinidis, M. C. Fairhurst, and A. F. R. Rahman, "A new multi-expert decision combination algorithm and its application to the detection of circumscribed masses in digital mammograms," Pattern Recognition, vol. 34, pp. 1527-37, August 2001.
23. R. R. Brooks, P. Ramanathan, and A. M. Sayeed, "Distributed target classification and tracking in sensor networks," Proceedings of the IEEE, vol. 38, pp. 1163-71, August 2003.
24. Q. Tao and R. Veldius, "Threshold-optimized decision-level fusion and its application to biometrics," Pattern Recognition, vol. 19, pp. 823-836, May 2009.
25. N. E. Mitrakis, J. B. Thoecharis, and V. Petridis, "Amultilayered neuro-fuzzy classifier with self-organizing properties," Fuzzy Sets and Systems, vol. 159, pp. 3132-59, December 2008.
26. A. R. Mirhosseini, H. Y. Lam, and T. K. M. Pham, "Human face image recognition: An evidence aggregation approach," Computer Vision and Image Understanding, vol. 23, pp. 213-230, August 1998.
27. M. A. Herraez, J. Domingo, and F. J. Ferri, "Combining similarity measures in content based image retrieval," Pattern Recognition Letters, vol. 29, pp. 2174-81, December 2008.
28. X. Chen, Y. Zhao, Y. Zhang, and R. Harrison, "Combining SVM classifiers using genetic fuzzy systems based on auc for gene expression data analysis," in International Symposium on Bioinformatics Research and Applications, Atlanta, May 2007, vol. 4463, pp. 496-505.
29. C. L. Blake and C. J. Merz, "UCI repository of machine learning databases," http://archive.ics.uci.edu/ml/.
30. "VisTex Database," http://vismod.media.mit.edu/vismod/imagery/VisionTexture/vistex .html.
31. T. Drugman, M. Gurban, and J. P. Thiran, "Relevant Feature Selection for Audio-Visual Speech Recognition," in IEEE 9th Workshop on Multimedia Signal Processing, Crete, Oct. 2007, pp. 179-182.
32. S. Manikandan and V. Rajamani, "A Mathematical Approach for Feature Selection and Image Retrieval of Ultra Sound Kidney Image Databases," European Journal of Scientific Research, vol. 24, pp. 163-171, December 2008.
33. L. I. Kuncheva, "Combining Pattern Classifiers: Methods and Algorithms," Wiley Inter-science, 2004.
34. L. I. Kuncheva, "Switching between selection and fusion in combining classifiers: An experiment," IEEE Trans. Systems, Man and Cybernetics, vol. 32, pp. 146-156, April 2002.
35. N. Littlestone and M. Warmth, "Weighted majority algorithm," Information and Computation, vol. 108, pp. 212-261, February 1994.36
36. Y. S. Huang and C. Y. Suen, "A method of combining multiple experts for the recog-nition of unconstrained handwritten numerals," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 17, pp. 90-94, January 1995.
37. P. D. Gader, M. A. Mohamed, and J. M. Keller, "Fusion of handwritten word classifiers," Pattern Recognition Letters, vol. 17, pp. 577-584, May 1996.
38. G. Shafer, A Mathematical Theory of Evidence, Princeton University Press, 1976.
39. K. Fukunaga, "Introduction to Statistical Pattern Recognition, Second Edition (Computer Science and Scientific Computing Series)," Academic Press, 1990.
40. J. C. Schlimmer, "Efficiently inducing determinations: Acomplete and systematic search algorithm that uses optimal pruning," in International Conference on Machine Learning (Morgan Kaufmann Publisher), Amherst, Jun. 1993, pp. 284-290.
41. M. Kudo and J. Sklansky, "Comparison of algorithms that select features for pattern classifiers," Pattern Recognition, vol. 33, pp. 25-41, January 2000.
42. H. Liu and R. Setiono, "Aprobabilistic approach to feature selection -a filter solution," in International Conference on Machine Learning (Morgan Kaufmann Publisher), Bari, Jun. 1996, pp. 319-327.
43. J. G. Dy and C. E. Brodley, "Feature subset selection and order identification for unsupervised learning," in International Conference on Machine Learning (Morgan Kaufmann Publisher), Stanford, Jun. 2000, pp. 247-254.
44. Y. Li, M. Dong, and J. Hua, "Localised feature selection for clustering," Pattern Recognition Letters, vol. 29, pp. 10-18, January 2008.
45. A. Frank, D. Geiger, and Z. Yakhini, "Adistance-based branch and bound feature selection algorithm," in Conference on Uncertainty in AI, Mexico, Aug. 2003, pp. 241-248.
46. X. Guorong, C. Peiqi, and W. Minhui, "Bhattacharyya distance based feature selection," in International Conference on Pattern Recognition, Vienna, Aug. 1996, pp. 195-199.
47. J. Li, S. Chu, J. H. Chang, and J. Pan, "Discriminant feature fusion strategy for supervised learning," in International Conference on Intelligent Information Hiding and Multimedia Signal, California, Dec. 2006, pp. 301-304.
48. A. Patra and S. Das, "Dual space based face recognition using feature fusion," in International Conference on Visual Information Engineering, Bangalore, Sept. 2006, pp. 155-160.
49. O. A. Shiba, W. Saeed, M. N. Sulaiman, F. Ahmad, and A. Mamat, "Toward optimal feature subset selection," in IEEE Student Conference on Research and Development, Aug. 2003, pp. 376-380.
50. D. W. Liang, Q. M. Huang, S. Q. A. Jiang, H. X. Yao, and W. Gao, "Mean-shift blob tracking with adaptive feature selection and scale adaptation," in International Conference on Image Processing, San Antonio, Sept. 2007, pp. 369-372.
51. N. S. Madsen, C. Thomsen, and J. M. Pena, "Unsupervised feature subset selection," in Proc. Workshop on Probabilistic Graphical Models for Classification (PKDD'03), Croatia, Sept. 2003, pp. 71-82.

52. Y. Zhao, L. Zhang, and P. Li, "Texture feature fusion for high resolution satellite image classification," in International Conference on Computer Graphics, Imaging and Vision, Beijing, Jul. 2005, pp. 19-23.
53. K. Schneider, "A new feature selection score for multinomial Naive Bayes text clas-sification based on KL-divergence," in 42nd Annual Meeting of the Association for Computational Linguistics, Barcelona, Jul. 2004, pp. 186-189.
54. M. Dash, K. Choi, P. Scheuermann, and H. Liu, "Feature selection for clustering a filter solution," in International Conference on Data Mining, Maebashi city, Dec. 2002, pp. 115-122.
55. D. J. Slotta, J. P. Vergara, N. Ramakrishnan, and L. S. Heath, "Algorithms for feature selection in rank-order spaces," Tech. Rep., 2005.
56. Z. Zhao and H. Liu, "Searching for interacting features," in The 20th International Joint Conference on AI (IJCAI-07), Hyderabad, Jan. 2007, pp. 1156-61.
57. M. A. Hall, "Correlation-based feature selection for discrete and numeric class machine learning," in Proceedings 17th International Conference on Machine Learning, Stanford, Jun. 2000, pp. 359-366.
58. P. Mitra, C. A. Murthy, and S. K. Pal, "Unsupervised feature selection using feature similarity," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 24, pp. 301- 312, March 2002.
59. B. Guo, R. I. Damper, S. R. Gunna, and J. D. B. Nelsona, "A fast separability-based feature-selection method for high-dimensional remotely sensed image classification," Pattern Recognition, vol. 41, pp. 1653-62, May 2008.
60. B. Cao, D. Shen, J. Sun, Q. Yang, and Z. Chen, "Feature selection in a kernel space," in International Conference on Machine Learning, Oregon, Jun. 2007, pp. 121-128.
61. J. Yang and J. Y. Yang, "Generalized K-L transform based combined feature extrac-tion," Pattern Recognition, vol. 35, pp. 295-297, January 2002.
62. W. Li, W. Gong, Y. Liang, and W. Chen, "Feature selection based on KPCA, SVM and GSFS for face recognition," in International Conference on Advances in Pattern Recognition, Bath, Aug. 2005, pp. 344-350.
63. H. M. Genc, Z. Cataltepe, and T. Pearson, "A PCA/ICA based feature selection method and its application for corn fungi detection," in Eusipco (European Signal Processing Conference), Poland, Sep 2007, pp. 970-974.
64. B. Paskaleva, M. M. Hayat, Z. Wang, J. S. Tyo, and S. Krishna, "Canonical correlation feature selection for sensors with overlapping bands: Theory and application," IEEE Trans. on Geoscience and Remote Sensing, vol. 46, pp. 3346-58, October 2008.
65. Q. S. Sun, S. G. Zeng, P. A. Heng, and D. S. Xia, "Feature fusion method based on canonical correlation analysis and handwritten character recognition," in IEEE Conference on Control, Automation, Robotics and Vision Conference, Kunming, Dec. 2004, pp. 1547-52.
66. D. Puig and M. A. Garcia, "Automatic texture feature selection for image pixel classification," Pattern Recognition, vol. 39, pp. 1996-09, November 2006.

AUTHORS

Utthara Gosa Mangai received her B. Tech degree in Information and Technology from Adhiyamaan College of Engineering, Anna University, Hosur, Tamil Nadu, India in 2005. Currently, she is pursuing her Master of Science (MS) degree in Computer Science and Engineering from Indian Institute of Technology, Madras, Chennai, India. Her research interests are in Classifier Combination, Pattern Recognition and Image Segmentation.

E-mail: utthara.a@gmail.com

Suranjana Samanta received her B. Tech degree in Computer Science and Engineering from Kalyani Government Engineering College, West Bengal, in 2007. She completed her Master of Science (MS) degree in Computer Science and Engineering from Indian Institute of Technology, Madras, Chennai, India in 2010. Her research interests are in Computer Vision, Machine Learning and Pattern Recognition.

E-mail: suranjanasamanta@yahoo.co.in

Sukhendu Das was born in 1962, in Kharagpur, W.B., India. He is currently working as an Associate Professor in the Deptt. Of Computer Science and Engg., IIT Madras, Chennai, India. He completed his B.Tech degree from IIT Kharagpur in the Deptt. Of Electrical Engg. in 1985 and M. Tech Degree in the area of Computer Technology from IIT Delhi in 1987. He then obtained his Ph.D degree from IIT Kharagpur in 1993. His current areas of research interests are: Visual Perception, Computer Vision: Digital Image and Video Processing and Pattern Recognition, Computer Graphics, Artificial Neural Networks, Computational Science and engineering and Soft Computing. Dr. Sukhendu Das has been a faculty of the Deptt. of CS&E, IIT Madras, India since 1989. He has worked as a visiting scientist in the University of Applied Sciences, Pforzheim, Germany, for post-doctoral research work, from Dec. 2001 till May 2003; and in the Univ. of UWA, Perth Australia, during June-Aug. 2006 and July-Sept. 2008. He has completed guiding two Ph.D (currently guiding two more) students, 21 M.S. (currently guiding five), several M. Tech and B. Tech students. He had completed several international and national sponsored projects and consultancies, both as principle and co-investigators. Currently, he is involved in three (3) sponsored projects/consultancies in IIT Madras. He has published about 100 technical papers in international and national journals and conferences. He has reviewed several papers in international journals (IEEE, Elsevier etc.) and chaired several sessions in conferences. He has received two best papers and a best design contest award.

E-mail: sdas@iitm.ac.in

Pinaki Roy Chowdhury was born in Calcutta in July 1968. He did his B.Sc. (Hons.) in Physics from Delhi University in 1988. Subsequently he completed his M.Sc. in Computer Science from School of Computer Science, Devi Ahilya Vishwavidyalaya, Indore in 1990 as a sponsored candidate of DRDO. He joined DRDO in TBRL, Chandigarh in August 1990 and subsequently moved to DTRL, Delhi in April 1992. He completed his Ph.D. in Computer Engineering from IT-BHU in 2001. He has worked in the field of machine learning, pattern recognition, global optimization, softcomputing and their applications in the area of image processing. He is a Life Member of Indian Society of Remote Sensing (ISRS), a Senior Member of IEEE and a Professional Member of ACM.

E-mail: rcpinaki@yahoo.com

DOI: 10.4103/0256-4602.64604; Paper No TR_136_09; Copyright © 2010 by the IETE