

# Matching-Pursuits Dissimilarity Measure for Shape-Based Comparison and Classification of High-Dimensional Data

Raazia Mazhar, *Student Member, IEEE*, Paul D. Gader, *Senior Member, IEEE*,  
and Joseph N. Wilson, *Member, IEEE*

**Abstract**—In this paper, a new matching-pursuits dissimilarity measure (MPDM) is presented that compares two signals using the information provided by their matching pursuits (MP) approximations, without requiring any prior domain knowledge. MPDM is a flexible and differentiable measure that can be used to perform shape-based comparisons and fuzzy clustering of very high-dimensional, possibly compressed, data. A novel prototype-based classification algorithm, which is termed the computer-aided minimization procedure (CAMP), is also proposed. The CAMP algorithm uses the MPDM with the competitive agglomeration (CA) fuzzy clustering algorithm to build reliable shape-based prototypes for classification. MP is a well-known sparse-signal approximation technique, which is commonly used for video and image coding. The dictionary and coefficient information produced by MP has previously been used to define features to build discrimination- and prototype-based classifiers. However, existing MP-based classification applications are quite problem-domain specific, thus making their generalization to other problems quite difficult. The proposed CAMP algorithm is the first MP-based classification system that requires no assumptions about the problem domain and builds a bridge between the MP and fuzzy clustering algorithms. Experimental results also show that the CAMP algorithm is more resilient to outliers in test data than the multilayer perceptron (MLP) and support-vector-machine (SVM) classifiers, as well as prototype-based classifiers using the Euclidean distance as their dissimilarity measure.

**Index Terms**—Competitive agglomeration (CA), fuzzy clustering, matching pursuits (MP), matching-pursuits dissimilarity measure (MPDM), outlier detection, prototype-based classification.

## I. INTRODUCTION

MATCHING pursuits (MP) is a well-known technique for sparse signal representation [1], [2]. MP is a greedy algorithm that finds linear approximations of signals by iteratively projecting them over a redundant, possibly nonorthogonal, set of signals, which is called the dictionary. It is useful for approximations when it is difficult to come up with an optimal orthogonal basis, as in the case of high-dimensional data or images. One of the important applications of MP is video encoding at very low bit rates [3]–[6], which has been shown to produce better picture quality and compression than the standard video coding techniques involving the discrete cosine transform [3]. MP is

Manuscript received November 11, 2008; revised March 6, 2009; accepted April 29, 2009. First published June 5, 2009; current version published October 8, 2009. This work was supported in part by the U.S. Army Communications-Electronics Command under Contract W909MY-05-D-0001.

The authors are with the Department of Computer and Information Science and Engineering, University of Florida, Gainesville, FL 32611 USA (e-mail: rmazhar@gmail.com; pgader@cise.ufl.edu; jnw@cise.ufl.edu).

Digital Object Identifier 10.1109/TFUZZ.2009.2024413

also suitable for coding other signal types, like images [7]–[10], audio [11], [12], and scattered waves [13], [14].

The MP approximation of a signal contains accurate shape-based information about the signal. Therefore, this shape-based information can be used to form feature vectors for classification. This idea has been used to build classification applications where MP has been used as a feature extractor to train discrimination-based classifiers or to build templates for prototype-based classifiers [12], [15]–[17]. Although MP is a generic approximation technique, the existing classification applications using MP as a feature extractor usually incorporate underlying assumptions about the data, like some specific dictionary structure or calculation methods. This binds these classifiers to their specific problem domains, thus making their generalization quite difficult. We have yet to find an MP-based classification application that requires no prior knowledge.

Therefore, in order to reduce the dependence of MP-based classification applications on the problem domain, in this paper, we propose a matching pursuits dissimilarity measure (MPDM) that can compare two signals<sup>1</sup> without requiring prior knowledge about the problem domain. It compares two signals based on the basic principle of the MP algorithm, i.e., iterated projections over dictionary elements. It compares two signals by comparing their best-matching dictionary elements and projection coefficients, as well as their approximation errors, called residues, produced during the MP process. MPDM is capable of performing shape-based comparisons of very high dimensional data, and it can also be adapted to perform magnitude-based comparisons, similar to the Euclidean (EUC) distance. Since MPDM is a differentiable measure, it can be seamlessly integrated with existing clustering or discrimination algorithms. Therefore, MPDM may find application in a variety of classification and approximation problems of very high dimensional data. Also, to use the MPDM, one needs to find an MP approximation of only one of the two signals, saving half the time of finding MP approximations of both signals and comparing them. Above all, the MPDM can compare two compressed signals, as opposed to uncompressed signals needed for a non-MP based similarity measure. This property of MPDM can be quite useful where the data is very high dimensional and the system has

<sup>1</sup>A 1-D signal  $x(t)$ , for  $t = [1, \dots, n]$ , can also be considered to be a point in the high-dimensional vector space  $\mathbb{R}^n$ . Hence, the terms *signal* and *high-dimensional data* will be used interchangeably throughout the text for a signal  $x$ .

limited storage resources. A preliminary discussion of MPDM was also presented in our earlier work in [18].

In this paper, we also propose a prototype-based classification algorithm using MPDM that builds shape-based prototypes for classification of very-high-dimensional data. MPDM is used with the competitive agglomeration (CA) fuzzy clustering algorithm [19] to yield a probabilistic classification model. This algorithm has been named computer-aided minimization procedure (CAMP), since it is a combination of the CA and MP algorithms. The CA algorithm is a fuzzy clustering algorithm that learns the optimal number of clusters during training, thus eliminating the need to manually prespecify the number of clusters. For a two-class problem ( $y \in \{0, 1\}$ ), CAMP clusters members of each class separately and uses the cluster representatives as prototypes. The prior probability  $p(y|c_j)$  of a class is computed based on similarity of the cluster  $c_j$  to clusters of the other class. The likelihood  $p(x|c_j)$  of a point  $x$  is determined using MPDM. The likelihood  $p(x|c_j)$  and the prior  $p(y|c_j)$  are used to compute the posterior probability  $p(y|x)$  of  $x$  belonging to a class  $y$ . A test point  $t$  that has low posterior probabilities for both classes may be considered to be an outlier.

The need to build a prototype-based classifier that can perform shape-based comparison arises from real-world applications where discrimination-based classifiers may not work well on outliers in the data. An outlier may be defined as a test pattern that does not belong to the distribution of any class in the training set. Linear discrimination-based classifiers, like multilayer perceptrons (MLPs) and support vector machines (SVMs) [20], differentiate between two classes by using a decision boundary that often extends to infinity in all directions. However, the decision boundary is meaningless beyond the region containing the training points. Therefore, the class label that is assigned to an outlier using this boundary will essentially be a random value. Therefore, a prototype-based classifier may be used as they have outlier-detection mechanism inherently built in their construction. In a prototype-based classifier, a test point is assigned a class label based on its similarity to the prototypes of that class. If a test point is dissimilar to the prototypes of all classes, it will not be assigned a class label and may be considered to be an outlier. Clustering-based algorithms are examples of prototype-based classifiers [21]. Using the CAMP algorithm with the MPDM as comparison measure helps build robust shape-based prototypes and helps to identify outliers in the test data.

MPs have been previously used as a feature extractor for discrimination-based classifiers. However, the CAMP algorithm is the first method that builds a bridge between fuzzy clustering and MP techniques. The experimental results also show the usefulness of CAMP for classification of high-dimensional data. The CAMP algorithm has been used to classify real landmine detection data collected using an electromagnetic induction (EMI) sensor. Each mine and nonmine object type has a signature shape-based metal response received in high-dimensional vectors. The classification performance of the CAMP algorithm has been found to be better than an existing MLP-based system for these data. Our CAMP algorithm also outperformed SVMs using nonlinear radial basis function (RBF) as the

kernel. The experimental results also demonstrate the superiority of MPDM over the EUC distance for shape-based comparisons in high dimension. An extensive experiment using simulated data is also reported to demonstrate the outlier-detection capabilities of CAMP over discrimination-based classifiers and the prototype-based classifier using the EUC distance.

The rest of the paper is organized as follows. Section II describes the MP algorithm and its existing uses in classification applications. The MPDM measure is defined in Section III. Section IV describes the CAMP algorithm. The experimental results are reported in Section V, while Section VI concludes the paper.

## II. BACKGROUND

### A. MP Algorithm

The MP algorithm is an iterative, greedy process that approximates a signal  $x$  using an overcomplete set of signals, which is called the dictionary  $D$  [2]. Let  $H$  be a Hilbert space, with  $x \in H$  and  $D = \{g_d\}_{d=1}^M \subseteq H$ , such that  $\|g_d\| = 1$ . Then, the MP algorithm generates an approximation of  $x$  as a linear combination of  $p$  elements from  $D$  as follows:

$$\hat{x} = w_0^{(x)} g_{d_0}^{(x)} + w_1^{(x)} g_{d_1}^{(x)} + \cdots + w_{p-1}^{(x)} g_{d_{p-1}}^{(x)} \quad (1)$$

where  $\hat{x}$  denotes the approximation of  $x$ , and  $w_j^{(x)}$  is the coefficient of the dictionary element  $g_{d_j}^{(x)}$ . The algorithm starts by finding the dictionary element  $g_{d_0}$  that produces the biggest absolute coefficient when  $x$  is projected on it

$$g_{d_0}^{(x)} = g_{d_0}, \quad \text{where } d_0 = \arg \max_k |\langle x, g_k \rangle|. \quad (2)$$

The difference between  $x$  and  $g_{d_0}^{(x)}$  times its magnitude is called the residue  $R_1^{(x)}$  and is given by

$$R_1^{(x)} = x - (w_0^{(x)} g_{d_0}^{(x)}) \quad (3)$$

where  $w_0^{(x)} = \langle x, g_{d_0}^{(x)} \rangle$  is the coefficient of  $g_{d_0}^{(x)}$ .

This process continues iteratively by projecting the subsequent residues  $R_i^{(x)}$  onto dictionary atoms and updating the next  $R_{i+1}^{(x)}$  accordingly. After  $p$  iterations,  $x$  can be written as a linear combination of the chosen dictionary elements and the residue as follows:

$$x = \sum_{j=0}^{p-1} w_j^{(x)} g_{d_j}^{(x)} + R_p^{(x)} \quad (4)$$

where  $w_j^{(x)} = \langle R_j^{(x)}, g_{d_j}^{(x)} \rangle$ , and  $R_0(x) \equiv x$ . Therefore, MP completely defines a signal  $x$  by three things:

- 1) the projection sequence of  $x$ , i.e., the ordered list of dictionary elements chosen to approximate  $x$ , which is represented as  $G(x) = \{g_{d_0}^{(x)}, g_{d_1}^{(x)}, \dots, g_{d_{p-1}}^{(x)}\}$ ;
- 2) the coefficient vector, i.e., the set of coefficients  $w_{d_j}^{(x)}$  that is denoted by  $W(x, G(x)) = \{w_0^{(x)}, w_1^{(x)}, \dots, w_{p-1}^{(x)}\}$ ;
- 3) the final residue  $R_p^{(x)}$  of  $x$  after  $p$  iterations, which is denoted by  $R(x, G(x))$ .

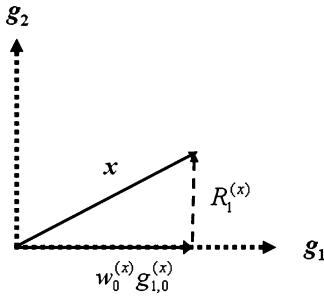


Fig. 1. MP approximation of  $x$  using the standard basis of  $\mathbb{R}^2$  as the dictionary and  $p = 1$ . Note that the residue is orthogonal to the projection of  $x$  onto  $g_1$ .

Fig. 1 shows the graphical illustration of the MP approximation where a signal  $x \in \mathbb{R}^2$  is projected on a simple dictionary consisting of the standard basis of  $\mathbb{R}^2$ .

The total number of iterations  $p$  can either be fixed beforehand or it can be chosen dynamically by iterating until  $R_p^{(x)}$  becomes smaller than a threshold. Since MP is a greedy algorithm, the chosen coefficients should get smaller as the iteration index  $j$  increases. Hence, the maximum information about the signal  $x$  is contained in its first few MP coefficients. Therefore, MP also has a denoising effect on the signal  $x$ . Choice of  $D$  depends on the expected characteristics of the data for a given problem. Choosing a dictionary that is well suited to a given dataset is still an open area of research [22]–[25].

The MP algorithm does not require the dictionary elements to be mutually orthogonal. Therefore, the approximation solution given by MP may not be a globally optimal one. However, it is useful for speedy approximation of signals when it is difficult to come up with an optimal orthogonal basis, as in the case of high-dimensional data or images. Various enhancements have been proposed in the literature that attempt to find the globally optimal and orthogonal representations of  $x$  using MP [26]–[28].

### B. Classification Using MP

Although, MP is commonly used as a signal representation and compression technique [3], [15], it has also been used in some classification applications. The most common use of MP for classification is as a feature extractor for discrimination-based classifiers. The MP approximation of each training pattern  $x_i$  is first computed. Then, the projection sequence  $G(x_i)$  and the coefficients vector  $W(x_i, G(x_i))$  are concatenated together to form the feature vector for  $x_i$  [16], [17], [29]. The advantage of using this approach is that it gives explicit control over the dimensionality of the feature vector by setting an appropriate  $p$  while computing the MP approximations of the training dataset. However, this method suffers from the traditional weaknesses of discrimination-based classifiers in identifying outliers in the data.

A prototype-based classifier using MP can be built by learning separate MP dictionaries for each class [30]–[32]. The MP approximations of the test pattern  $t$  are found using these dictionaries. It is assigned the label of the class whose dictionary gives the best MP approximation of  $t$ , i.e., the dictionary that produced the smallest residue. However, a signal can be reason-

ably approximated using many different types of dictionaries. Therefore, assigning  $t$  a class label based on minimal residue may not be the most reliable method of classification.

A prototype-based classifier can also be built by using one dictionary for the entire training dataset and using the projection sequences and coefficient vectors of training data to build prototypes. Let  $X_C = \{x_{c_1}, \dots, x_{c_C}\}$  be a set of class representatives that are chosen based on the training set. Then, a prototype-based classifier using MP can be built by storing pairs of projection sequences and coefficient vectors of each  $x_{c_i}$  as prototypes for each class  $c_i$  [9]. A test pattern  $t$  is projected onto each projection sequence  $G(x_{c_i})$  and the corresponding coefficient vector  $W(t, G(x_{c_i}))$  is noted. The point  $t$  is assigned to the class whose  $W(x_{c_i}, G(x_{c_i}))$  has the smallest EUC distance to the computed coefficient vector  $W(t, G(x_{c_i}))$ . This coefficient comparison method for classification may work well when the test object is quite similar to the matching prototype and the residue  $R(t, G(x_{c_i}))$  is negligible. However, when  $t$  is different from all the prototypes, its residue will not be negligible. Similarly, Naoto *et al.* [33] compared two images by comparing their MP approximations.

The approach of comparing coefficient vectors of projected and original objects has also been used to build MP filters (MPFs) [7], [34]. The MPFs are used to locate objects in test images. The filter is correlated with the image by projecting it onto the projection sequence of the filter. The location in which the coefficient vectors of the projected image and the filter are most similar is declared the matching location. Since the residue is not considered while comparing the filter to the image, MPFs also suffer from the risk of false matches in the image.

### III. MATCHING PURSUITS DISSIMILARITY MEASURE

As discussed earlier, the MP approximation completely defines a signal  $x$  in terms of its projection sequence  $G(x)$ , coefficient vector  $W(x, G(x))$ , and residue  $R(x, G(x))$ . Therefore, in order to accurately compare two signals  $x_1$  and  $x_2$ , we need to compare these. This can be done by projecting  $x_1$  onto the projection sequence  $G(x_2)$  of  $x_2$  and noting the corresponding coefficient vector  $W(x_1, G(x_2))$  and residue  $R(x_1, G(x_2))$ . Based on these factors, the MPDM is defined as

$$\delta(x_1, x_2) = \sqrt{\alpha D_R(x_1, x_2) + (1 - \alpha) D_W(x_1, x_2)} \quad (5)$$

where  $D_R(x_1, x_2)$  is the difference between the residues of  $x_1$  and  $x_2$  when both are projected onto the projection sequence  $G(x_2)$  of  $x_2$

$$D_R(x_1, x_2) = \|R(x_1, G(x_2)) - R(x_2, G(x_2))\|^2 \quad (6)$$

and  $D_W(x_1, x_2)$  compares their corresponding MP coefficients and is given by

$$D_W(x_1, x_2) = \|W(x_1, G(x_2)) - W(x_2, G(x_2))\|^2. \quad (7)$$

MPDM compares two signals by projecting them over the same subspace defined by  $G(x_2)$ . The first half of the MPDM, where the residues are being compared, compares the distance of  $x_1$  and  $x_2$  from this subspace. The second half of MPDM, where coefficients are being compared, compares the distance

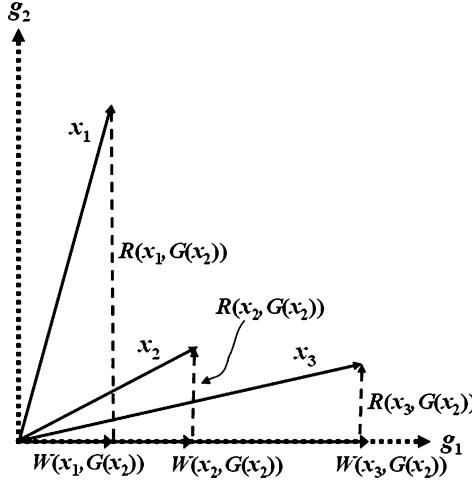


Fig. 2. Role of  $\alpha$  in determining the value of MPDM. To compare the signals  $x_1$  and  $x_3$  with  $x_2$ , they are projected onto the projection sequence of  $x_2$ , i.e.,  $G(x_2) = \{g_1\}$ . When  $\alpha \rightarrow 1$ ,  $x_2$  and  $x_3$  are deemed more similar as they are similar in shape (or orientation), but when  $\alpha \rightarrow 0$ , more emphasis is given to comparing the projection coefficients of two signals in subspace defined by  $G(x_2)$ , thus making  $x_1$  and  $x_2$  more similar.

between projections of  $x_1$  and  $x_2$  within the subspace  $G(x_2)$ . The importance of comparing both coefficients and the residue for MPDM is demonstrated in [18, Figs. 2 and 3].

The parameter  $\alpha \in [0, 1]$  determines the relative importance of the residues and coefficients in the final value of  $\delta$ . The  $\alpha$  value can be set to obtain many different types of comparisons using MPDM. For example, if we are mainly interested in comparing the distance of two points  $x_1$  and  $x_2$  from a reference subspace  $G(x)$ , we would like to set  $\alpha$  closer to 1, thereby giving more weight to the residues. The effect will be like comparing the shape of two signals while disregarding their actual intensity values. On the other hand, if we ignore the residues and are only interested in comparing the distance between images of  $x_1$  and  $x_2$  in the subspace  $G(x_2)$ , we would like to set  $\alpha$  closer to 0, which will yield a comparison of intensities of the two signals, regardless of whether their shapes match or not. Fig. 2 illustrates the role of  $\alpha$  in determining the type of comparison being made using MPDM.

Note that the projection of  $x_1$  and  $x_2$  onto the subspace of  $G(x_2)$  is not necessarily orthogonal, unless the orthogonal MP or some other explicit orthogonalization method is used. However, if the dictionary forms an orthogonal basis and all its elements are used in the approximation, i.e.,  $p = M$ , then the approximation is exact. Also, if  $\alpha = 0$  in case of an exact approximation, the measure reduces to the EUC norm. Usually, however, in applications where MPDM is used, the signals or images are very high dimensional (hundreds or thousands of dimensions), and the dictionaries are not orthogonal. Also, for ease of computation, the number of dictionary elements used is much smaller than the size of dictionary, i.e.,  $p \ll M$ .

The reason why MPDM can work well with high-dimensional data is that its complexity does not depend upon the dimensionality of the input signals; instead, it depends on the total number  $p$  of dictionary elements used. If the residue of  $x_2$  is

negligible, we can discard it. Then, we only need to store the projection sequence  $G(x_2)$  and the corresponding coefficients  $W(x_2, G(x_2))$  of  $x_2$  for comparison. In this case, MPDM has the unique property that it can compare two signals while working on compressed data. This property of MPDM can be quite useful where the data are very high dimensional and the system has limited storage resources.

The MPDM is not a symmetric measure. However, it can be made symmetric as follows:

$$\bar{\delta}(x_1, x_2) = \frac{1}{2}(\delta(x_1, x_2) + \delta(x_2, x_1)). \quad (8)$$

It can be shown that when  $\alpha \in (0, 1)$ ,  $\bar{\delta}(x_1, x_2)$  is a pseudometric. The proof can be found in Appendix A. Also as discussed earlier, when an MP dictionary forms an orthogonal basis and  $\alpha = 0$ , MPDM reduces to the EUC distance. In this special case, MPDM is also a metric.

Therefore, by only using the MP projection sequence and corresponding coefficients of the signals, the MPDM eliminates the need to incorporate any domain-specific information into signal comparison. Since the complexity of the MPDM depends upon all the dictionary elements used for MP approximation, it can reliably perform shape- or magnitude-based comparisons of very high-dimensional data by only varying the parameter  $\alpha$ . As can be noted from Appendix A, the MPDM is also a pseudometric. Above all, the MPDM can compare two compressed signals, as opposed to uncompressed signals needed for a non-MP-based similarity measure. Since the MPDM is a differentiable measure, it can be seamlessly integrated with existing clustering or discrimination algorithms. Therefore, in the following section, we present the CAMP algorithm, which is a prototype-based classifier that uses the MPDM for clustering and classification.

#### IV. CLASSIFICATION USING MPDM

Let  $X = \{(x_1, y_1), \dots, (x_N, y_N)\}$  be a dataset, where each  $x_i$  is a signal with binary label  $y_i$  associated with it, such that  $y_i \in \{0, 1\}$ . The labels 0 and 1 represent class 0 and class 1, respectively. Given this labeled training data, we wish to use Bayes decision rule to assign class label  $y$  to a test point  $x$  as follows:

$$y = \begin{cases} 1, & \text{if } P(y=1|x) > P(y=0|x) \\ 0, & \text{otherwise.} \end{cases} \quad (9)$$

Therefore, the learning task is to learn the parameters for  $p(y|x)$ . Let  $C = \{c_1, \dots, c_M\}$  be a set of prototypes representing the dataset  $X$ , where  $M \ll N$ . These prototypes are a quantization of  $X$  and can be the cluster representatives obtained from unsupervised clustering of  $X$ . Each prototype  $c_j$  has probabilities of class labels  $p(y=0|c_j)$  and  $p(y=1|c_j)$ . Given the probabilities  $p(y|c_j)$ , we can introduce probabilities conditional on  $c_j$  for  $p(y=1|x)$  in the Bayesian decision rule of (9) as follows:

$$p(y=1|x) = \sum_{j=1}^M p(y=1|c_j)p(c_j|x) \quad (10)$$

or, by Bayes rule, which is given by

$$p(y=1|x) = \sum_{j=1}^M \frac{p(y=1|c_j)p(x|c_j)p(c_j)}{p(x)}. \quad (11)$$

Assuming that all the prototypes are equally likely and ignoring the normalization constant in the denominator, we have

$$p(y=1|x) \propto \sum_{j=1}^M p(y=1|c_j)p(x|c_j). \quad (12)$$

Similarly, the probability of  $x$  having a label  $y = 0$  is given by

$$p(y=0|x) \propto \sum_{j=1}^M p(y=0|c_j)p(x|c_j). \quad (13)$$

Therefore, the training process requires estimation of the prototypes  $C = \{c_j\}_{j=1}^M$  and associated probabilities  $p(y|c_j)$ . Then, the testing process, i.e., assigning a class label to a point  $x$ , requires estimation of the probabilities  $p(x|c_j)$  and application of the Bayes rules (12) and (13). The idea is to use the unsupervised clusters  $C$  to build a fuzzy-nearest-prototype-based classification system using the MPDM and also to use the fuzzy membership of a point  $x$  into the cluster  $c_j$  as  $p(x|c_j)$  to assign a class label  $y$  to  $x$ . An equivalent fuzzy interpretation of the previous equations can be found in [35], where Frigui and Gader treat the class prior  $p(y=1|c_j)$  as the fuzzy membership of the prototype  $c_j$  into class  $y$ . Here, we are addressing the two-class problem, but the generalization to  $n$  classes is straightforward.

Choosing a suitable algorithm for clustering the dataset  $X$  to obtain the cluster representatives  $c_j$  can be tricky. We want the memberships of data points to vary smoothly between various clusters. Therefore, fuzzy clustering is preferred over crisp clustering algorithms, like  $K$ -means. However, since our goal is to minimize human intervention in the training process, instead of using the standard fuzzy  $C$ -means (FCM) algorithm [36], we use a clustering algorithm that discovers the optimal number of clusters during training. Therefore, we use the CA fuzzy clustering algorithm [19]. CA is a generalized form of the FCM algorithm that discovers the optimal number of clusters during training. Given  $X = \{x_1, \dots, x_N\}$ ,  $x_i \in \mathbb{R}^n$ , and cluster centers  $C = \{c_1, \dots, c_M\}$ , the CA algorithm minimizes the following objective function:

$$J(B, U, X) = \sum_{j=1}^M \sum_{i=1}^N u_{ij}^2 d^2(x_i, c_j) - \eta \sum_{j=1}^M \left[ \sum_{i=1}^N u_{ij} \right]^2 \quad (14)$$

subject to  $\sum_{j=1}^M u_{ij} = 1$ , for  $i \in \{1, \dots, N\}$ .

The function  $d^2(x_i, c_j)$  measures the dissimilarity of the data point  $x_i$  to the center  $c_j$  and  $u_{ij}$  represents the degree of membership of  $x_i$  in the cluster  $c_j$ . The first component of (14) tries to minimize the dissimilarity between a cluster center and its members, and the second component tries to minimize the total number of clusters. Hence, the CA algorithm reduces redundancies in cluster representatives by merging together similar clusters, while preserving dissimilar clusters. The update equation for the memberships  $u_{ij}$  and a suitable choice for  $\eta$  can be found in Appendix B.

Since we are dealing with high-dimensional data  $x_i$ , we will use the MPDM as the dissimilarity measure with the CA algorithm. Using MPDM will give us meaningful shape-based comparisons between cluster centers  $c_j$  and  $x_i$ . MPDM is used as the dissimilarity measure in (14) to yield

$$\begin{aligned} J(C, U, X) = & \sum_{j=1}^M \sum_{i=1}^N u_{ij}^2 [\alpha D_R(x_i, c_j) + (1 - \alpha) D_W(x_i, c_j)] \\ & - \eta \sum_{j=1}^M \left[ \sum_{i=1}^N u_{ij} \right]^2 \end{aligned} \quad (15)$$

subject to  $\sum_{j=1}^M u_{ij} = 1$ , for  $i \in \{1, \dots, N\}$ .

By using the MPDM with CA, we get a three-phase optimization algorithm, i.e., updating the cluster centers  $c_j$ , updating the MP approximations of cluster centers  $\hat{c}_j(G(c_j))$ , and updating the membership  $u_{ij}$ . We call this algorithm CAMP, as it is an abbreviation of CA and MPDM.

In order to find an update equation for  $c_j$ , let us make the dependence of (15) on  $c_j$  explicit. From (4), we have  $R(x_i, G(x_i)) = x_i - \hat{x}_i$ , where  $\hat{x}_i$  is defined in (1). Thus, (15) becomes

$$\begin{aligned} J(C, U, X) = & \alpha \sum_{j=1}^M \sum_{i=1}^N u_{ij}^2 \| (x_i - \hat{x}_i(G(c_j)) - (c_j - \hat{c}_j(G(c_j))) \|^2 \\ & + (1 - \alpha) \sum_{j=1}^M \sum_{i=1}^N u_{ij}^2 D_W(x_i, c_j) - \eta \sum_{j=1}^M \left[ \sum_{i=1}^N u_{ij} \right]^2 \end{aligned} \quad (16)$$

where  $\hat{x}_i(G(c_j)) \equiv x_i - R(x_i, G(c_j))$ , i.e., the approximation of  $x_i$  when projected on the projection sequence of  $c_j$ . Let us assume the memberships  $u_{ij}$  and MP approximations  $\hat{c}_j(G(c_j))$  as constants for this computation. Then, differentiating (16) with respect to  $c_t$  and setting it equal to 0 yields

$$-2\alpha \sum_{i=1}^N u_{it}^2 [x_i - \hat{x}_i(G(c_t)) - c_t + \hat{c}_t(G(c_t))] = 0. \quad (17)$$

Rearranging terms to obtain an update equation for  $c_t$ , we have

$$c_t = \hat{c}_t(G(c_t)) + \left( \sum_{i=1}^N v_{it} x_i - \sum_{i=1}^N v_{it} \hat{x}_i(G(c_t)) \right) \quad (18)$$

where

$$v_{it} = \frac{u_{it}^2}{\sum_{k=1}^N u_{ik}^2} \quad (19)$$

or more succinctly, we write (18) as follows:

$$c_t = \hat{c}_t + (\mu_t - \tilde{\mu}_t) \quad (20)$$

where  $\mu_t$  and  $\tilde{\mu}_t$  denote the weighted average of the original and approximated signals in cluster  $t$ , respectively. Note that  $\tilde{\mu}_t$  is not the projection of  $\mu_t$  in the subspace  $G(c_t)$  but a weighted average of the projections  $\hat{x}_i(G(c_t))$ . After  $c_t$  has been estimated,  $\hat{c}_t(G(c_t))$  can be recomputed by calculating the MPs decomposition of  $c_t$  over the dictionary  $D$ . The memberships  $u_{ij}$  can

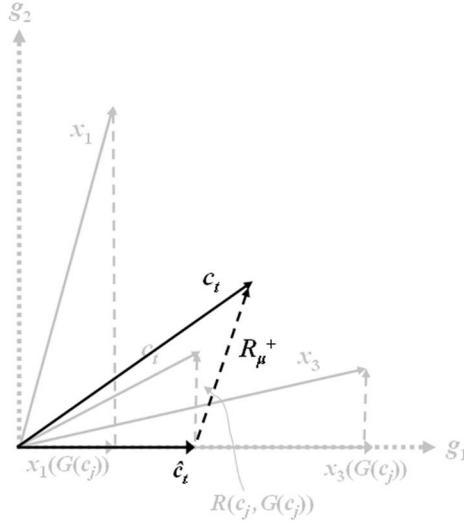


Fig. 3. Update equation for  $c_t$ . For some membership values  $\mu$ , the  $R_\mu^+$  may update the  $c_t$ , as shown. Note that  $R_\mu^+$  is not necessarily orthogonal to  $\hat{c}_t$ . On the other hand, for MP approximation, the residue  $R(c_j, G(c_j))$  is always orthogonal to the approximation  $\hat{c}_t$  of the signal  $c_t$  (see Fig. 1).

then be updated and the three-phase alternating optimization proceeds in this fashion.

Let us denote the error vector between  $\mu_t$  and  $\tilde{\mu}_t$  with  $R_\mu^+$ , and call it the pseudoresidue. Then, (20) can be written as

$$c_t = \hat{c}_t + R_\mu^+. \quad (21)$$

The closeness of each point  $x_i$  to the subspace  $G(c_t)$  and the memberships  $u_{it}$  affect the overall value of  $R_\mu^+$ . If an  $x_i$  is similar to  $c_t$ , the weighted difference between  $x_i$  and  $\hat{x}_i(G(c_t))$  will be small. Their difference can also be small if the membership  $u_{it}$  of  $x_i$  in  $c_t$  is low. Fig. 3 shows a graphical interpretation of (21), which is quite similar in construction to the MP approximation equation (4). However, (21) defines a process that is the exact opposite of MP. In MP, the approximated signal  $\hat{x}$  is built using the original signal  $x$ , and the residue  $R_p^{(x)}$  is the by-product. On the other hand, while updating the cluster center  $c_t$ , the approximated signal  $\hat{c}_t$  is calibrated using the residue  $R_\mu^+$  to build the original signal  $c_t$ . This interpretation of (21) is consistent with the standard update equations of clustering algorithms where cluster centers are updated using the cluster members. Therefore, CAMP builds a bridge between the MPs and the clustering algorithms. Since MPDM can work with any clustering algorithm, similar update equations can be derived for other clustering algorithms, like  $K$ -means, etc.

Once the cluster centers  $c_j$  have been found, the probabilities  $p(y|c_j)$  to each cluster, which is represented by  $c_j$ , can be assigned in many ways. However, since we are building separate models for both classes to compare with  $x$ , we cluster the samples from both classes separately using the CAMP algorithm. Now, all the clusters centers  $c_j$  belonging to the class 1 will have  $p(y = 1|c_j) = 1$  and  $p(y = 0|c_j) = 0$ , and vice versa. Therefore, we need to adjust these probabilities for  $c_j$  based on

its similarity to a  $c_l$ , which belongs to the other class

$$p(y = 1|c_j) = \frac{\omega_{c_j 1}}{\omega_{c_j 0} + \omega_{c_j 1}} \quad (22)$$

where  $\omega_{c_j k}$  measures the similarity of  $c_j$  to the closest cluster with the class label  $k \in \{0, 1\}$  and is given by

$$\omega_{c_j k} = f(\min_{l|y_{c_l}=k} \bar{\delta}(c_j, c_l)). \quad (23)$$

The function  $f(\cdot)$  is a monotonically decreasing function that is used to convert the dissimilarity measure MPDM into a similarity measure. If both  $c_j$  and  $c_l$  have the same class label  $k$ , then  $c_j = c_l$ , and  $\omega_{c_j k}$  will have the maximum value. On the other hand, when both  $c_j$  and  $c_l$  have opposite labels, the value of  $\omega_{c_j k}$  will depend on their dissimilarity, as measured by the MPDM. Similar to (22), the equation for  $p(y = 0|c_j)$  is given by

$$p(y = 0|c_j) = \frac{\omega_{c_j 0}}{\omega_{c_j 0} + \omega_{c_j 1}}. \quad (24)$$

The probabilities  $p(y = 0|c_j)$  and  $p(y = 1|c_j)$  can be interpreted as the fuzzy memberships of the prototype  $c_j$  in the classes  $y = 0$  and  $y = 1$ , respectively. Equations (22) and (24) use the minimum distance between prototypes of both classes and FCM-based labeling to assign these memberships [35].

Once the probabilities  $p(y|c_j)$  have been updated for all  $c_j$ , the probability  $p(x|c_j)$  of a point  $x$  belonging to a cluster  $j$  can be calculated as a similarity between  $x$  and  $c_j$  as

$$p(x|c_j) \propto f(\delta(x, c_j)). \quad (25)$$

Since we are anticipating presence of outliers in the test data, we will not normalize (25) over all  $c_j$  because normalization would remove the notion of distance from  $p(x|c_j)$ . Suppose  $x_1$  is an outlier, and  $x_2$  is not. Then, regardless of their actual distance from the prototypes,  $p(x_1|c_j) = p(x_2|c_j)$  if both  $x_1$  and  $x_2$  lie at the same relative distance from all  $c_j$  (see [37], Fig. 8]). Therefore, in order to be able to detect outliers in the test data, we use the absolute similarity of  $x$  to  $c_j$  in (25).

Given  $c_j$ ,  $p(y|c_j)$  and  $p(x|c_j)$ , the relations defined in (12) and (13) can be used to assign  $p(y = 1|x)$  and  $p(y = 0|x)$ , respectively, to  $x$ . If both  $p(y = 1|x)$  and  $p(y = 0|x)$  are low,  $x$  can be identified as an outlier. Otherwise,  $x$  is assigned a class label using the Bayesian decision rule of (9).

Therefore, given a labeled dataset  $X$ , the following steps summarize the process of training a CAMP classifier on  $X$ .

- 1) Build the prototypes  $c_j$  that represent the dataset  $X$  using the CAMP algorithm.
- 2) Find the probabilities of class labels  $p(y = 0|c_j)$  and  $p(y = 1|c_j)$  using (22) and (24).

A test point  $t$  can be assigned a confidence value by the following steps.

- 1) Find the probability of  $t$  belonging to each cluster  $c_j$ , i.e.,  $p(t|c_j)$  using (25).
- 2) Use (12) and (13) to find the probabilities  $p(y = 1|t)$  and  $p(y = 0|t)$ .
- 3) If both  $p(y = 1|t)$  and  $p(y = 0|t)$  are low, identify  $t$  as an outlier. Otherwise, apply the Bayesian decision rule (9) to assign a class label to  $t$ .

The strength of the CAMP algorithm lies in its reliable shape-based comparisons using MPDM and finding required number of clusters during training using the CA algorithm. Since CAMP is able to detect the outliers in test data, its misclassification rate is lower, thus leading to a robust and reliable classifier. However, since the outcome of the MPDM comparison depends on the choice of  $\alpha$ , various shape- or magnitude-based classifiers can be built using CAMP by only altering the  $\alpha$  parameter.

The biggest advantage of the CAMP algorithm is that it tries to minimize the incorporation of human interpretation in the form of free parameters in the training and classification processes. This was the main motivation behind using the CA algorithm for clustering as opposed to the standard FCM. CAMP is also the first algorithm that builds a bridge between the MPs and the fuzzy clustering algorithms. Its classification and outlier-detection abilities are better than the discrimination-based classifiers. We demonstrate this superiority of the CAMP algorithm in Section V.

## V. EXPERIMENTAL RESULTS

### A. Results on Real Data

In order to determine the potential usefulness of the proposed method for real-world problems, the CAMP algorithm has been tested on landmine detection data. The data were collected using an EMI sensor installed on a handheld landmine-detection unit [38]. The response of a target to the EMI sensor depends on the metallic properties of the buried object such as metal content and conductivity. Each target object has a signature response that is fairly consistent across various weather and soil conditions. However, there may be considerable similarities between metallic signatures of landmines and metallic clutter objects. Thus, we seek to build reliable shape-based models of mine and nonmine targets using their EMI response. The EMI data are collected by sweeping the unit across a suspected target location ten times. The EMI sensor collects data in two channels. The signals from each channel of all sweeps are interpolated to a standard length of  $n = 180$  and are smoothed to remove noise. Features are extracted from these processed training data to train the classifiers.

The training data consisted of mines and nonmines from three data collections  $A$ ,  $B$ , and  $C$ . Collections  $A$  and  $B$  are from two temperate testing sites in the Eastern U.S., and collection  $C$  is from an arid test site in the Western U.S. Overall, the training dataset comprises 500 targets, 203 of which are mines, and the rest are various metallic and nonmetallic nonmine objects. Two datasets  $T_1$  and  $T_2$  were used to test the performance of various classifiers. The dataset  $T_1$  was collected over the same site as training dataset  $B$  but on a different date. Since samples of an earlier collection over the same site were part of the training data,  $T_1$  was quite similar to the training data. On the other hand, the dataset  $T_2$  came from a different temperate site in the Eastern U.S. and contains data from some target types that were not present in any of the training dataset sites. Therefore,  $T_2$  contained many samples that were quite different from the training dataset. Table I summarizes the mine/nonmine composition of these datasets.

TABLE I  
NUMBER OF MINES AND NONMINES IN DATASETS

Dataset	Mines	NonMines	Total
Training	203	297	500
Testing $T_1$	44	108	152
Testing $T_2$	112	34	146

An existing system uses a discrimination-based approach called the feedforward ordered weighted average (FOWA) [38], [39], which is a variation of MLP neural network system. FOWA assigns a mine confidence value to a test pattern based on a set of features extracted from multiple sweeps of its EMI data. The training data are divided into four classes based on the metal content and classification of the target, namely the high-metal mine (HMM), low-metal mine (LMM), high-metal clutter (HMC), and low-metal clutter (LMC) classes. Four separate MLP networks are trained that specialize in finding each class type. Outputs from these networks are combined to assign a mine confidence value  $y_{\text{mine}}$  to a test point as follows:

$$y_{\text{mine}} = \max(y_{\text{HMM}}, y_{\text{LMM}}) \times (1 - \max(y_{\text{HMC}}, y_{\text{LMC}})). \quad (26)$$

A higher confidence value means that the probability of the target being a mine is high and *vice versa*. A detailed discussion of these features and training of the FOWA system can be found in [38]. We use the CAMP algorithm for prototype-based classification of the EMI data and compare our results with that of the FOWA network.

SVMs are considered robust discrimination-based classifiers. Therefore, in addition to the FOWA network, an SVM-based system was also trained for performance comparison. Since the features used by FOWA have been shown to work quite well with a discrimination-based classifier, i.e., an MLP, we used the same features to train the SVM. Like FOWA, four separate SVMs were trained for each class, and their outputs were combined to assign a mine confidence to a test point. RBFs were used as the kernel function for the SVMs. The values for the regularization parameter  $C$ , which controls the generalization capabilities of an SVM [20], and  $\sigma^2$  for the RBF kernel were found by searching over a range of values between 0 and 100 for each parameter. While looking for optimal values of  $C$  and  $\sigma^2$ , each parameter was incremented with a step size of 0.1 between 0 and 1, with a step size 1 between 1 and 10, and with a step size of 10 between 10 and 100. After training with each pair of values for  $C$  and  $\sigma^2$ , the receiver operating characteristic (ROC) curve<sup>2</sup> for classification result of dataset  $T_1$  was generated. The set of parameters that produced the smallest mean probability of false alarm (PFA) at the probabilities of detection (PD) of 90, 95, and 100 was chosen as the final parameter values for the reported SVM, thereby producing the best SVM over this wide range of values. The parameter values corresponding to this SVM classifier were  $\sigma^2 = 0.1$  and  $C = 0.1$ .

<sup>2</sup>An ROC is a graphical plot of the probability of correct classification versus probability of misclassification for a binary classifier system as its discrimination threshold is varied.

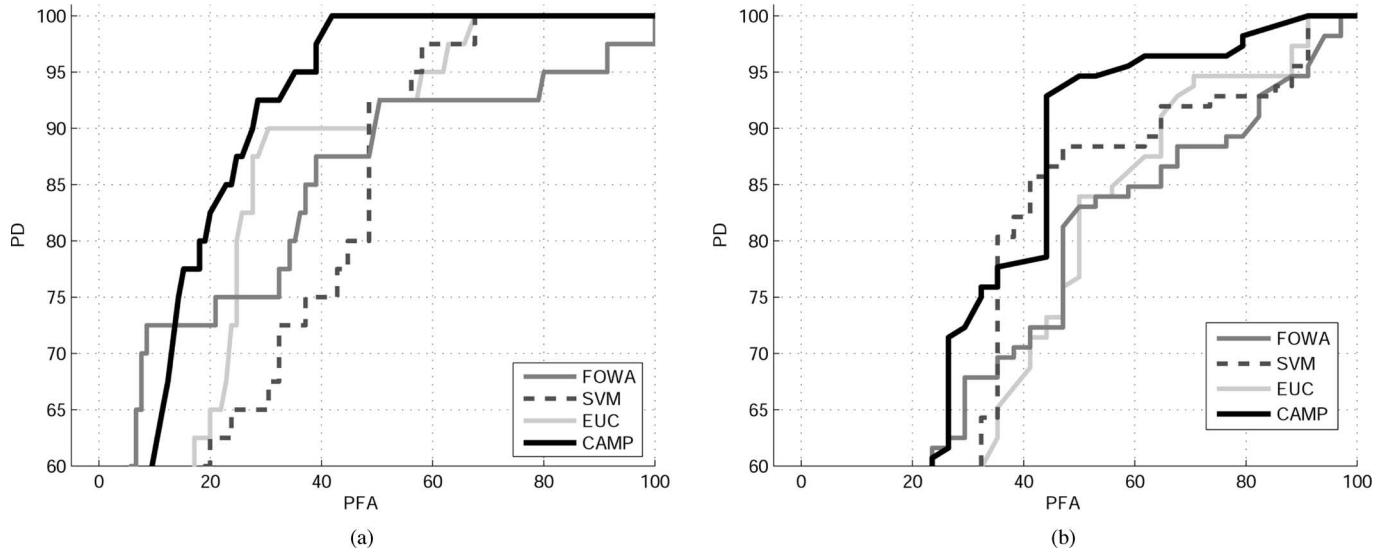


Fig. 4. ROC of PD versus PFA for mine and nonmine classes. (a) Test dataset  $T_1$ . (b) Test dataset  $T_2$ .

In order to build a prototype-based classifier using the CAMP algorithm, the signals from both channels are concatenated together to form signals of length  $2n$ . For training purposes, each sweep was treated as an independent sample. The MP dictionary was built from the training data by an automated process that segments each signal based on its zero crossings, retains each segment as a separate dictionary element, and then reduces the dictionary size using the enhanced K-SVD (EKSVD) dictionary-learning algorithm discussed in [40]. In this experiment, the dictionary size was reduced from 1088 segments to 21 elements. Samples from the mine and the nonmine classes were clustered separately using the CAMP algorithm. The prototypes  $c_j$  and their class probabilities  $p(y|c_j)$  were found using the method explained in the previous section. For  $f(z)$  in (23), we used the exponential decay function

$$f(z) = \exp^{(-(1/\sigma^2)z)} \quad (27)$$

where  $\sigma^2$  is a normalization constant. The value  $\sigma^2 = 0.16$  was empirically found to be useful for both mine and nonmine clusters on the training data. For these EMI data, not only the shape but also the intensity of the received signal constitute the signature of a target. For this reason, the value of  $\alpha$  is set equal to 0.7 for the MPDM. The CAMP algorithm learned a total of 110 prototypes, with 60 being mine prototypes and 50 being nonmine prototypes.

To compare performance of the MPDM and EUC measures, another prototype-based classifier was also learned using the EUC distance. The EUC classifier was quite similar to CAMP, except it uses the EUC distance instead of the MPDM for clustering and class assignments. The same  $f(z)$  defined in (27) was used with  $\sigma^2 = 0.16$ . A total of 225 prototypes were learned with 126 being mine prototypes and the rest being nonmine prototypes. For both the MPDM and EUC methods, each sweep in the test dataset was independently assigned a mine and nonmine confidence. The final confidence values assigned to a test target

was generated by averaging the top three confidence values of its sweeps.

For notational ease, let us denote these four trained classifiers as FOWA, SVM, EUC, and CAMP, respectively. Fig. 4(a) and (b) shows ROC curves for classification results of each dataset for the four classifiers. Each test pattern is assigned a confidence value  $p(y = 1|x)$  of being a mine. Therefore, these ROCs record the percentage of detection of mines versus the percentage of incorrect classification of nonmines, i.e., the percentage of false alarms.

CAMP performs better than the discrimination-based classifiers FOWA and SVM on both datasets at  $PD \geq 90$  because it uses a prototype-based approach to classification with robust shape-based dissimilarity measure. Note that although EUC is also a prototype-based classifier, it does not do any better than FOWA or SVM. This is because the EUC distance is unable to perform accurate shape-based comparisons in high-dimensional vectors.

The results on dataset  $T_2$  are especially interesting as the performance of all four classifiers suffers from the presence of outliers in both mine and nonmine classes. The PD of FOWA and SVM is adversely affected for  $T_2$  by misclassifying these outliers from both classes. On the other hand, CAMP gives a low mine confidence to any test pattern that is different from its mine prototypes, i.e., the nonmines and the outliers. Therefore, the performance of CAMP for detecting mines suffers only because of assigning low confidence to outliers in the mine class. Note that CAMP also assigns a low nonmine confidence to all outliers. Therefore, all test points that have a low  $p(y = 1|x)$  and a low  $p(y = 0|x)$  can be identified as the outliers. For this landmine and other applications, the fact that an algorithm is able to identify patterns that are not represented in the training data is important as the decision can be deferred to a human operator monitoring this system. Once again, the EUC classifier suffers because of its inability to perform shape-based comparisons.

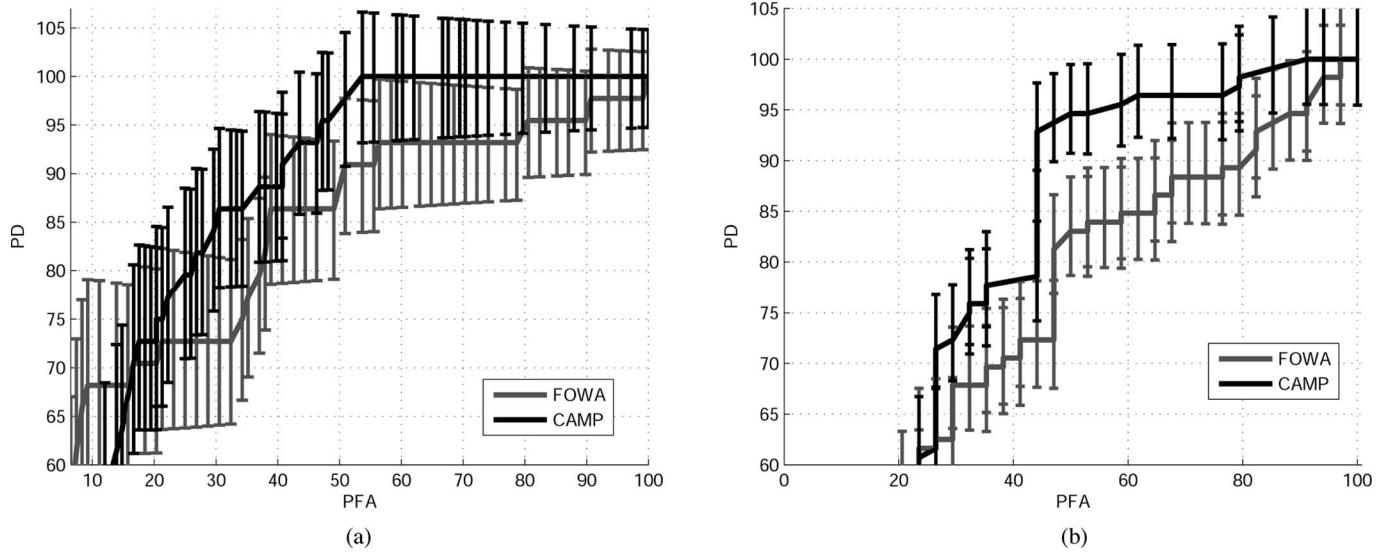


Fig. 5. ROC with error bars for FOWA and CAMP algorithms. (a) Test dataset  $T_1$ . (b) Test dataset  $T_2$ .

Fig. 5 shows the error bars for the CAMP and FOWA ROCs, denoting the degree of uncertainty associated with their results. These error bars were created by treating each instance of mine found in the test dataset as a success in a binomial trial. Both CAMP and FOWA show higher confidence in their outputs for higher values of  $p(y = 1|x)$  and *vice versa*. Note that the confidence intervals of FOWA and CAMP overlap for the test dataset  $T_1$ . This is because dataset  $T_1$  is quite similar to the training dataset. On the other hand, since  $T_2$  is somewhat different from the training dataset, their confidence intervals are farther apart.

### B. Results on Simulated Data

In order to further establish the superiority of the CAMP algorithm to build shape-based prototypes and identify the outliers in test data, an extensive study on simulated data was also conducted. Ho *et al.* [41] described a parametric model that emulates the output of the EMI sensor used to collect the landmine data in the previous experiment. This mathematical model produces the signal that the EMI sensor would produce when swept over an object buried in the ground. Therefore, by altering only four parameters of the model, realistic EMI responses can be generated that are different from the actual landmine data used to train the classifiers in the previous experiment. Since these generated signals are different from the training dataset, they can be considered outliers. Using this model, a total of 10 000 signals were generated as test patterns.

Since these test patterns are not mine samples, theoretically, they should be assigned low confidence values by all classifiers. In order to test this hypothesis, the mine-confidence thresholds where the PD for the training set was between 55 and 100 were noted for each classifier. Then, the test patterns were assigned mine-confidence values using the four classifiers FOWA, SVM, EUC, and CAMP, which are trained in the previous experiment. FOWA and SVM classifiers were assigned mine confidence using (26), while CAMP and EUC were assigned mine confidence as  $p(y = 1|x)$ . The mine confidences assigned by each classifier

were thresholded by their respective PD values, and the number of patterns below each threshold were noted for all four classifiers. Fig. 6(a) shows the percentage of test patterns below the PD thresholds for all classifiers. Since none of the patterns were from the mine class, the superior shape-based prototypes approach of CAMP was able to reject almost all the patterns. On the other hand, FOWA and SVM rejected fewer patterns. Fig. 7 shows the mine confidence intervals for CAMP and FOWA classifiers.

An unseen pattern should have a low mine, and a low nonmine confidence should be considered to be an outlier. Therefore, the same experiment was repeated to assign nonmine confidence to each unseen pattern. For FOWA and SVM, the following relation was used to assign the nonmine confidence:

$$y_{\text{nm}} = \max(y_{\text{HMC}}, y_{\text{LMC}}) \times (1 - \max(y_{\text{HMM}}, y_{\text{LMM}})) \quad (28)$$

while for CAMP and EUC,  $p(y = 0|x)$  was used. Fig. 6(b) shows the percentage of test patterns below the PD threshold (of the nonmine class) for all classifiers. Once again, CAMP assigned very low nonmine confidence to almost all the patterns, while FOWA and SVM rejected significantly fewer patterns.

Based on CAMP's ability to reject unseen patterns, a decision rule to identify outliers in the data can be devised. Fig. 8 shows one such rule. The  $p(y = 1|x)$  and  $p(y = 0|x)$  assigned by the CAMP algorithm to each pattern in the training set is rank-normalized and plotted in Fig. 8. Rank normalization is a technique that converts an unknown distribution into a uniform distribution. All the values in a given distribution are sorted in an ascending order. Each point of the original distribution is assigned a new value based on its index in this sorted list. The mine and nonmine confidence values of the training dataset were rank-normalized separately and are plotted against each other in Fig. 8. Using these rank indexes, 1000 unseen patterns were also assigned the mine and nonmine rank confidences and are plotted in Fig. 8. Almost all of the unseen patterns lie within an isocircle of radius 0.25. Therefore, this rank-normalized plot

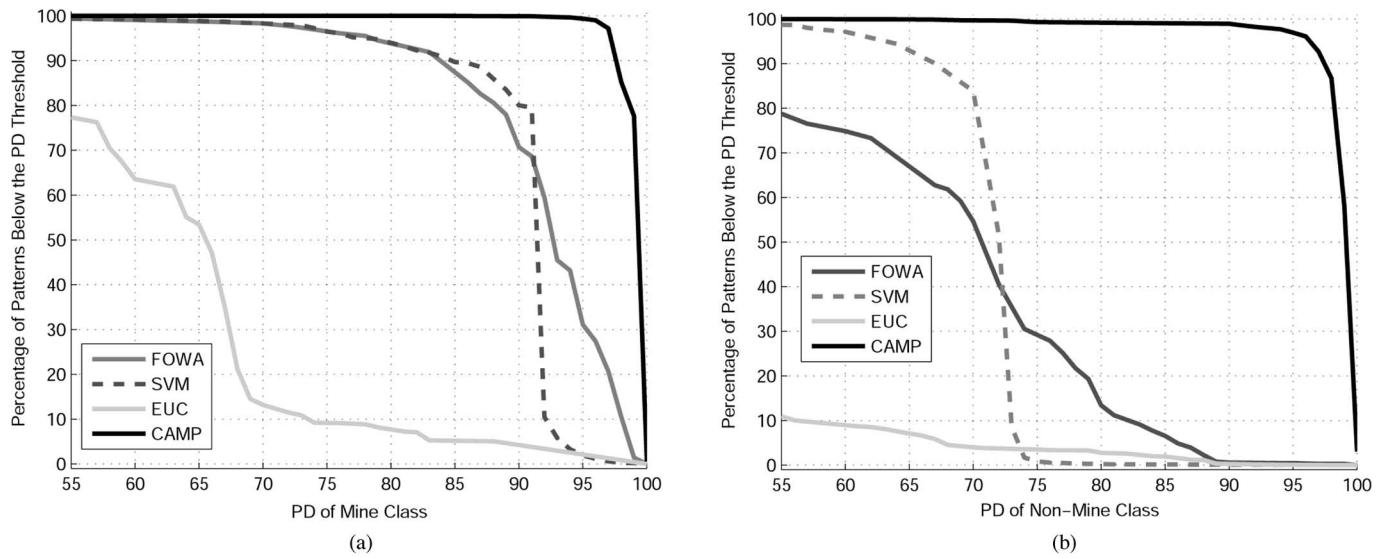


Fig. 6. Performance comparison of FOWA, SVM, EUC, and CAMP classifier to perform accurate shape-based classification of data. (a) Percentage of unseen patterns below the PD of mine class. (b) Percentage of unseen patterns below the PD of nonmine class.

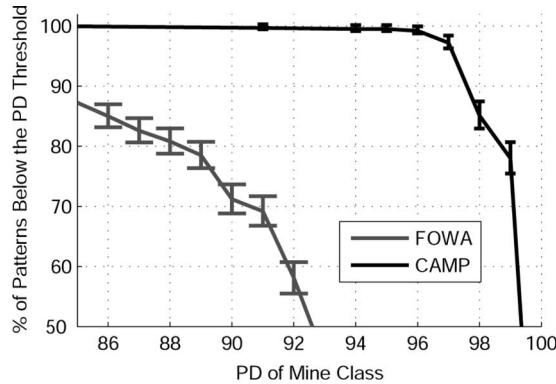


Fig. 7. Confidence intervals for FOWA and CAMP for mine class.

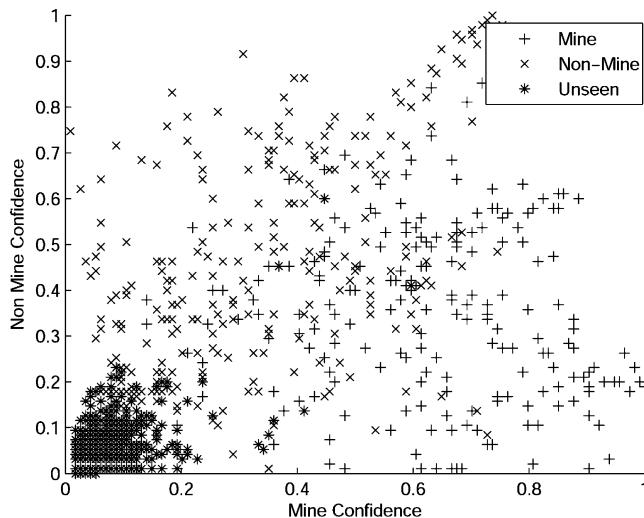


Fig. 8. Rank-normalized plot of  $p(y = 1|x)$  versus  $p(y = 0|x)$  for the training dataset and the resultant confidence values assigned to the unseen patterns.

can act as a guide to choose a radius within which each data point will be considered to be an outlier. Outside this radius, a test pattern can be assigned a class label using the Bayesian decision rule of (9).

Therefore, better classification performance and better outlier rejection by the proposed method yields a more reliable system.

## VI. CONCLUSION

In this paper, we presented an MPDM, which is capable of performing accurate shape-based comparisons between high-dimensional data. It extends the MPs signal approximation technique and uses its dictionary and coefficient information to compare two signals. MPDM is also used with the CA fuzzy clustering algorithm to build a prototype-based classifier called CAMP. The CAMP algorithm builds robust shape-based prototypes for each class and assigns a confidence to a test pattern based on its dissimilarity to the prototypes of all classes. If a test pattern is different from all the prototypes, it will be assigned a low confidence value. Therefore, our experimental results show that the CAMP algorithm is able to identify outliers in the given test data better than discrimination-based classifiers, like MLPs and SVMs.

## APPENDIX I

### PROOF OF MPDM BEING A PSEUDOMETRIC

A metric on a set  $X$  is a function  $d : X \times X \rightarrow \mathbb{R}$  that satisfies the following four conditions for all  $x_1, x_2$ , and  $x_3$  in  $X$ .

- 1)  $d(x_1, x_2) \geq 0$ .
- 2)  $d(x_1, x_2) = d(x_2, x_1)$ .
- 3)  $d(x_1, x_2) = 0 \Leftrightarrow x_1 = x_2$ .
- 4)  $d(x_1, x_3) \leq d(x_1, x_2) + d(x_2, x_3)$ .

If a dissimilarity measure satisfies only the first three conditions, then it is called a pseudometric. The symmetric MPDM  $\delta$  defined in (8) satisfies the first two conditions

by construction. The third condition is satisfied when  $\alpha \in (0, 1)$ . However, it may not always hold when  $\alpha = 0$  or  $\alpha = 1$ . When  $\alpha = 0$  and  $W(x_1, G(x_2)) = W(x_2, G(x_2))$ , then  $d(x_1, x_2) = 0$ , despite that  $x_1 \neq x_2$  and the third condition will not hold. One such example is shown in [18, Fig. 3], where the coefficients of  $x_1$  and  $x_2$  are quite similar. Also, in a very unlikely situation, it may be possible for two signals with different projection sequence and different coefficient vectors to have the same residues. Therefore, the third condition may not always hold when  $\alpha = 1$ .

For  $\alpha \in (0, 1)$ , the third condition of  $\bar{\delta}$  being a pseudometric can be proved by contradiction. Suppose that  $x_1 \neq x_2$  but  $\bar{\delta}(x_1, x_2) = 0$ , which also implies that  $\delta(x_1, x_2) = 0$  and  $\delta(x_2, x_1) = 0$ . If  $\delta(x_1, x_2) = 0$ , it means that the residues  $R(x_1, G(x_2))$  and  $R(x_2, G(x_2))$  are equal; in addition, the coefficients  $W(x_1, G(x_2))$  and  $W(x_2, G(x_2))$  are equal to each other, respectively. Since the residues of two different signals can be the same, we will only investigate the equality of  $W(x_1, G(x_2))$  and  $W(x_2, G(x_2))$ . Note that in the MPDM, we compare these coefficient vectors using the EUC distance, which is a metric and satisfies condition 3). Therefore,  $W(x_1, G(x_2))$  and  $W(x_2, G(x_2))$  have to be the same for  $\delta(x_1, x_2)$  to be equal to 0. Now,  $W(x_1, G(x_2))$  and  $W(x_2, G(x_2))$  were obtained by projecting both  $x_1$  and  $x_2$  on the same projection sequence  $G(x_2)$ . However, the iterative greedy projection method of MP assures that each signal  $x$  has a unique projection sequence  $G(x)$ , coefficient vector  $W(x, G(x))$ , and residue  $R(x, G(x))$ . Therefore, if the projection sequence, residue, and the coefficient vector of  $x_1$  and  $x_2$  are exactly the same, it must be that  $x_1 = x_2$ . This contradicts our assumption. Therefore,  $\hat{\delta}(x_1, x_2)$  is a pseudometric.

## APPENDIX II

### UPDATE EQUATIONS FOR $u_{ij}$ AND $\eta$

CA is a twofold iterative optimization algorithm that first assumes cluster centers  $c_j$  are constant and updates the memberships  $u_{ij}$ . Then, it assumes that the memberships  $u_{ij}$  are fixed and updates the cluster centers  $c_j$ . As shown by Frigui and Krishnapuram [19], in order to find the update equation for the class memberships  $u_{ij}$ , one needs to minimize the Lagrangian to obtain

$$u_{ij} = \frac{1/d^2(x_i, c_j)}{\sum_{k=1}^M (1/d^2(x_i, c_k))} + \frac{\eta}{d^2(x_i, c_j)} (N_j - \bar{N}_i) \quad (29)$$

where  $N_j = \sum_{i=1}^N u_{ij}$  is the cardinality of cluster  $j$ , and

$$\bar{N}_i = \frac{\sum_{k=1}^M (1/d^2(x_i, c_k)) N_k}{\sum_{k=1}^M (1/d^2(x_i, c_k))} \quad (30)$$

is the weighted average of the cluster cardinalities. Detailed derivation of (29) can be found in [19]. Note that the first half of (29) is the standard membership update equation for the FCM algorithm [36], and the second half is introduced by the CA algorithm. If the control parameter  $\eta$  is set to zero, then CA reduces to the FCM algorithm. Therefore, CA generalizes the FCM algorithm.

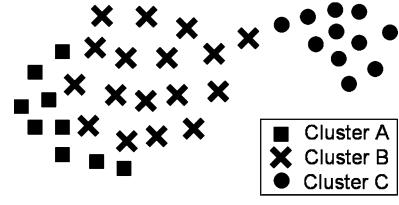


Fig. 9. CA algorithm merges clusters based on their cardinality. Appropriate changes are made to  $\eta$  so that CA merges clusters  $A$  and  $B$  but not  $C$ .

More succinctly, we can write  $u_{ij}$  as

$$u_{ij} = u_{ij}^{\text{FCM}} + \eta u_{ij}^{\text{bias}} \quad (31)$$

where the definitions of  $u_{ij}^{\text{FCM}}$  and  $u_{ij}^{\text{bias}}$  follow from (29). The  $u_{ij}^{\text{FCM}}$  term updates the membership  $u_{ij}$  using the standard FCM method, and  $u_{ij}^{\text{bias}}$  term increases or decreases  $u_{ij}$  based on the size of cluster  $j$ . The memberships of the clusters that need to be pruned are gradually diminished with the help of  $u_{ij}^{\text{bias}}$ . Eventually, the cluster becomes empty and is then pruned.

The parameter  $\eta$  controls the relative weight of  $u_{ij}^{\text{FCM}}$  and  $u_{ij}^{\text{bias}}$ . The  $u_{ij}^{\text{bias}}$  term should dominate initially to quickly remove the redundant clusters. Later on,  $u_{ij}^{\text{FCM}}$  should dominate to refine the cluster memberships of data points. Therefore,  $\eta$  is an exponential decay function of iteration number  $t$ , which is proportional to the update and bias terms of the objective function [19], and is given by

$$\eta(t) = \tau(t) \frac{\sum_{j=1}^M \sum_{i=1}^N u_{ij}^2 d^2(x_i, c_j)}{\sum_{j=1}^M \left[ \sum_{i=1}^N u_{ij} \right]^2} \quad (32)$$

where  $\tau(t)$  is an exponential decay function.

Since CA prunes the clusters based on their cardinalities, it can sometimes merge clusters that are quite dissimilar to each other. For example, consider the three clusters shown in Fig. 9. The clusters  $A$  and  $B$  should have been one big cluster. Therefore, CA should diminish memberships of points assigned to  $A$  so that all members of  $A$  can be merged into  $B$ . However, cluster  $C$  is a distinct cluster located at a distance from  $B$ , but there are a few points from both clusters that connect them through a thin link. This will lead the CA algorithm to believe that  $B$  and  $C$  should be one cluster that would try to prune  $C$ .

Therefore, a mechanism should be introduced in CA training that would protect smaller clusters from being merged into bigger clusters, even if they have some outliers. This can be achieved by modifying the regularization parameter  $\eta$ . We know from (31) that  $u_{ij}^{\text{bias}}$  alters the memberships of a cluster  $j$  based on its size, relative to other clusters. If the distance of cluster  $j$  from the rest of the clusters is larger than some threshold  $T$ , then  $u_{ij}^{\text{bias}}$  should be zero, and membership values of cluster  $j$  should be updated using only  $u_{ij}^{\text{FCM}}$ . This can be done by modifying the regularization parameter  $\eta$  as follows:

$$\eta(t) = \kappa_j \tau(t) \frac{\sum_{j=1}^M \sum_{i=1}^N u_{ij}^2 d^2(x_i, c_j)}{\sum_{j=1}^M \left[ \sum_{i=1}^N u_{ij} \right]^2} \quad (33)$$

where  $\tau(t)$  is an exponential decay function, and  $\kappa_j$  is defined as

$$\kappa_j = \begin{cases} 0, & \text{if } \min_{k,k \neq j} d^2(c_k, c_j) > T \\ 1, & \text{otherwise.} \end{cases} \quad (34)$$

The  $\kappa_j$  will be set to zero if  $c_j$  is farther than  $T$  from every other cluster center  $c_k$ . Therefore,  $\kappa_j$  defines the bound on the maximum size of the clusters, thus preventing them from growing in an unbound fashion. When  $\kappa_j = 1$ , the memberships of the smaller clusters, like cluster A, are gradually diminished so that they can be merged into a nearby bigger cluster, like B. However, when  $\kappa_j = 0$ , it preserves the smaller clusters, like the cluster C shown in Fig. 9.

### APPENDIX III

#### CLUSTER VALIDATION OF THE CAMP ALGORITHM

Like all clustering algorithms, CAMP imposes a clustering structure onto the dataset. We need to validate if the structure imposed by CAMP actually matches the underlying structure present in the data. For this purpose, we ran the CAMP algorithm on a dataset that had well-defined structure to see if CAMP was able to discover that structure. For comparison purposes, the same experiment was repeated using FCM for clustering, using MPDM as the dissimilarity measure. The clustering results of FCM were subjected to various cluster validation indexes to determine if FCM was able to determine the underlying structure of data and whether its outcome matched that of the CAMP algorithm. The reason why CAMP's clustering results were compared with FCM was because CA is a generalization of the FCM algorithm. Therefore, theoretically, the number of clusters validated for FCM should be the same number discovered by CA and should also match the underlying structure of the data.

In order to build a dataset for clustering, six signals of distinct shapes were chosen from the EMI sensor dataset that is discussed in Section V-A. Each signal was multiplied with a factor between 5 and 10 with an increment of 0.05. Hence, a total 101 signals were generated for each signal and introduced into the dataset. Therefore, the dataset consisted of 606 signals belonging to six shape-wise distinct classes, as shown in Fig. 10. For MP, the same dictionary that was used in Section V-A was used. Similarly,  $\alpha$  was set equal to 0.7.

For the CAMP algorithm, the initial number of clusters was set equal to 15. The data were independently clustered with CAMP 50 times with randomly chosen initial cluster centers. Fig. 11 shows the histogram of the number of clusters found by CAMP for 50 runs of the experiment. The number of clusters found for all trials form a normal distribution around 6, which is the actual number of clusters in the data. Fig. 10 shows the cluster centers found by CAMP on one of the trials. Therefore, the CAMP algorithm was able to discover the underlying structure in the data.

For FCM, MPDM was used as the dissimilarity measure with  $\alpha = 0.7$ . Let  $M$  denote the total number of clusters for FCM. For each  $M \in \{2, \dots, 15\}$ , the data were clustered using FCM, where initial cluster centers were chosen randomly. This experiment was repeated 50 times for all values of  $M$ . Since no single

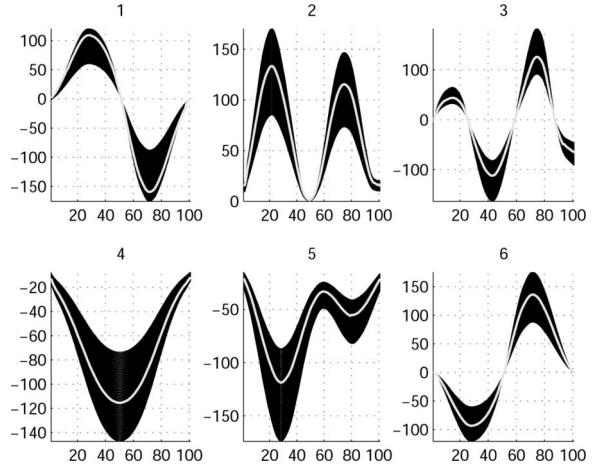


Fig. 10. Six-class dataset used for validating CAMP and FCM clustering. In each subplot, the 101 class members are shown in black and the respective cluster representative found by CAMP are shown in gray.

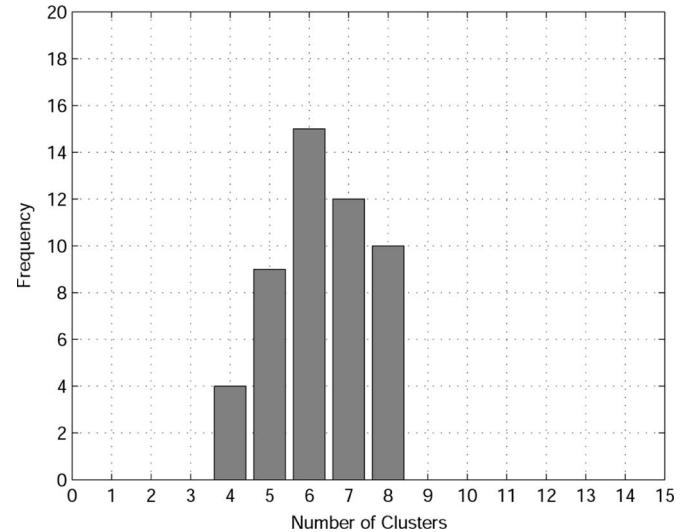


Fig. 11. Histogram of number of clusters discovered by CAMP on 50 independent runs producing a normal distribution with  $\mu = 6.3$  and  $\sigma^2 = 1.2$ .

validity measure is absolutely able to determine the optimal number of clusters for FCM, we computed the following four fuzzy validity indexes for each run of the FCM algorithm [36]:

- 1) partition coefficient index (PC);
- 2) partition entropy coefficient (PE);
- 3) Xie–Beni index (XB);
- 4) Fukuyama–Sugeno index (FS).

Except for PC, all other indexes are minimized for the optimal<sup>3</sup> number of clusters for a given dataset. For each of the 50 experiments, the number of clusters that maximized PC and minimized PE, XB, and FS were noted. Fig. 12 shows the histogram of the optimal number of clusters found by each index. The PC, PE, and FS indexes peak at 6 for the correct number of clusters for FCM. However, their distributions have

<sup>3</sup>Optimal with respect to that validity index. It may or may not match the actual number of clusters in the data.

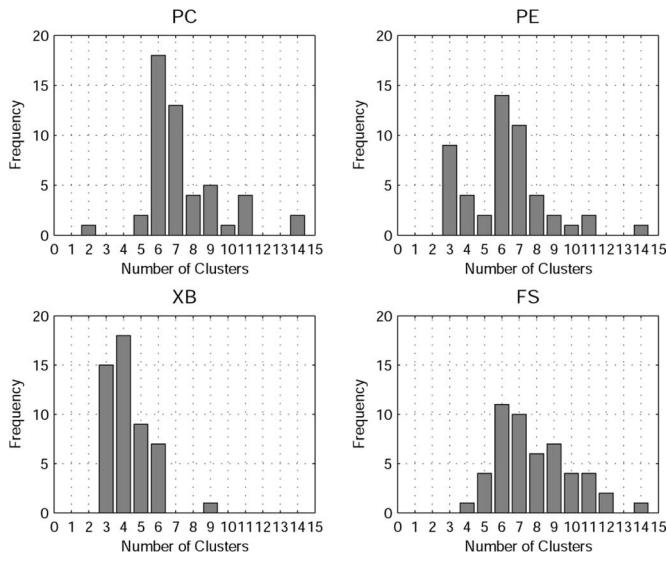


Fig. 12. Histogram of the optimal number of clusters discovered by PC, PE, XB, and FS cluster validity indexes for FCM on 50 independent runs.

wider spread than that of CAMP. On the other hand, the mode of the XB measures was not the correct number of clusters.

Therefore, both FCM and CAMP algorithms were able to discover the underlying structure of data when MPDM was used as the dissimilarity measure. However, using CAMP for clustering is a faster way to cluster because unlike FCM, where one needs to rerun clustering for every number of clusters, CAMP discovers the correct number of clusters by running the algorithm only once.

#### ACKNOWLEDGMENT

The authors would like to thank P. Ngan, R. Cresci, S. Burke, and R. Weaver of the Night Vision and Electronic Sensors Directorate for their support and sponsorship.

#### REFERENCES

- [1] J. H. Friedman and W. Stuetzle, "Projection pursuit regression," *J. Amer. Stat. Assoc.*, vol. 76, no. 376, pp. 817–823, 1981.
- [2] S. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3397–3415, Dec. 1993.
- [3] R. Neff and A. Zakhor, "Very low bit-rate video coding based on matching pursuits," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, no. 1, pp. 158–171, Feb. 1997.
- [4] D. Redmill, D. Bull, and P. Czerepinski, "Video coding using a fast non-separable matching pursuits algorithm," in *Proc. Int. Conf. Image Process. (ICIP)*, Oct. 4–7, 1998, vol. 1, pp. 769–773.
- [5] C.-K. Peng, W.-L. Hwang, and C.-L. Huang, "Matching pursuits low bit rate video coding with codebooks adaptation," in *Proc. Acoust., Speech, Signal Process. IEEE Int. Conf.* Washington, DC: IEEE Signal Process. Soc., 2000, pp. 408–411.
- [6] Y. Morvan, D. Farin, and P. H. N. de With, "Matching-pursuit dictionary pruning for MPEG-4 video object coding," *Internet Multimedia Syst. Appl.*, vol. 1, pp. 476–481, Feb. 2005.
- [7] P. J. Phillips, "Matching pursuit filters applied to face identification," *IEEE Trans. Image Process.*, vol. 7, no. 8, pp. 1150–1164, Aug. 1998.
- [8] F. Bergeaud and S. Mallat, "Matching pursuit of images," in *Proc. ICIP*, 1995, pp. 53–56.
- [9] T. Pham and A. Smeulders, "Sparse representation for coarse and fine object recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 4, pp. 555–567, Apr. 2006.
- [10] P. Vandergheynst and P. Frossard, "Efficient image representation by anisotropic refinement in matching pursuit," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2001, vol. 3, pp. 1757–1760.
- [11] D. Monro, "Basis picking for matching pursuits audio compression," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 21–24, 2006, pp. 2985–2988.
- [12] K. Umapathy, S. Krishnan, and S. Jimaa, "Multigroup classification of audio signals using time-frequency parameters," *IEEE Trans. Multimedia*, vol. 7, no. 2, pp. 308–315, Apr. 2005.
- [13] M. McClure and L. Carin, "Matching pursuits with a wave-based dictionary," *IEEE Trans. Signal Process.*, vol. 45, no. 12, pp. 2912–2927, Dec. 1997.
- [14] P. Runkle, L. Carin, L. Couchman, T. Yoder, and J. Bucaro, "Multiaspect target identification with wave-based matched pursuits and continuous hidden Markov models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 12, pp. 1371–1378, Dec. 1999.
- [15] A. Ebrahimi-Moghadam and S. Shirani, "Matching pursuit-based region-of-interest image coding," *IEEE Trans. Image Process.*, vol. 16, no. 2, pp. 406–415, Feb. 2007.
- [16] G. Herrero, A. Gotchev, I. Christov, and K. Egiazarian, "Feature extraction for heartbeat classification using independent component analysis and matching pursuits," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Mar. 18–23, 2005, vol. 4, pp. iv/725–iv/728.
- [17] B. Krishnapuram, J. Sichina, and L. Carin, "Physics-based detection of targets in SAR imagery using support vector machines," *IEEE Sens. J.*, vol. 3, no. 2, pp. 147–157, Apr. 2003.
- [18] R. Mazhar, P. Gader, and J. Wilson, "A matching pursuit based similarity measure for fuzzy clustering and classification of signals," in *Proc. IEEE Int. Fuzzy Syst. Conf.*, Jun. 2008, pp. 1950–1955.
- [19] H. Frigui and R. Krishnapuram, "Clustering by competitive agglomeration," *Pattern Recognit.*, vol. 30, no. 7, pp. 1109–1119, 1997.
- [20] S. Haykin, *Neural Networks: Comprehensive Foundation*. Englewood Cliffs, NJ: Prentice-Hall, 1998.
- [21] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed. New York: Wiley–Interscience, Nov. 1973.
- [22] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4311–4322, Nov. 2006.
- [23] P. Schmid-Saugeon and A. Zakhor, "Dictionary design for matching pursuit and application to motion-compensated video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 6, pp. 880–886, Jun. 2004.
- [24] B. Olshausen and D. Field, "Sparse coding with an overcomplete basis set: A strategy employed by V1?" *Vis. Res.*, vol. 37, no. 23, pp. 3311–3325, Dec. 1997.
- [25] P. Czerepinski, C. Davies, N. Canagarajah, and D. Bull, "Matching pursuits video coding: Dictionaries and fast implementation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 7, pp. 1103–1115, Oct. 2000.
- [26] Y. Pati, R. Rezaifar, and P. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," presented at 27th Annu. Asilomar Conf. Signals, Syst., Comput., Pacific Grove, CA, Nov. 1993.
- [27] G. M. Davis, S. G. Mallat, and Z. Zhang, "Adaptive time-frequency decompositions with matching pursuit," *Proc. SPIE*, vol. 2242, pp. 402–413, Mar. 1994.
- [28] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Rev.*, vol. 43, no. 1, pp. 129–159, 2001.
- [29] T. Hoang and D. Nguyen, "Matching pursuit for the recognition of power quality disturbances," in *Proc. IEEE 33rd Annu. Power Electron. Spec. Conf.*, 2002, vol. 4, pp. 1791–1796.
- [30] F. Mendels, P. Vandergheynst, and J.-P. Thiran, "Rotation and scale invariant shape representation and recognition using matching pursuit," in *Proc. 16th Int. Conf. Pattern Recognit.*, 2002, vol. 04, pp. 40-326–40-329.
- [31] W. Dangwei, M. Xinyi, and S. Yi, "Radar target identification using a likelihood ratio test and matching pursuit technique," *Inst. Electr. Eng. Proc.—Radar, Sonar Navigat.*, vol. 153, no. 6, pp. 509–515, Dec. 2006.
- [32] F. Mendels, P. Vandergheynst, and J. Thiran, "Affine invariant matching pursuit-based shape representation and recognition using scale-space," iTS, Ecublens, Switzerland, Tech. Rep., 2003.
- [33] K. Naoto, S. Masanori, Y. Keinosuke, and N. Makoto, "Color image decomposition by the matching pursuits and its application to image similarity measure," *IEIC Tech. Rep.*, vol. 102, no. 403, pp. 57–62, 2002.

- [34] P. Phillips, "Matching pursuit filter design," in *Proc. 12th IAPR Int. Conf. Pattern Recognit., Conf. C: Signal Process.*, Oct. 9–13, 1994, vol. 3, pp. 57–61.
- [35] H. Frigui and P. Gader, "Detection and discrimination of landmines based on edge histogram descriptors and fuzzy K-nearest neighbors," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, 2006, pp. 1494–1499.
- [36] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*, 2nd ed. New York: Elsevier/Academic, 2003.
- [37] H. Frigui and P. D. Gader, "Detection and discrimination of land mines in ground-penetrating radar based on edge histogram descriptors and a possibilistic K-nearest neighbor classifier," *IEEE Trans. Fuzzy Syst.*, to be published.
- [38] P. Ngan, S. P. Burke, R. Cresci, J. N. Wilson, P. D. Gader, D. Ho, E. E. Bartosz, and H. A. Duvoisin, "Development of processing algorithms for HSTAMIDS: Status and field test results," in *Proc. SPIE Conf. Detection Remediation Technol. Mines Minelike Targets X*, Apr. 2007, pp. 65532D-1–65532D-11.
- [39] W.-H. Lee, P. D. Gader, and J. N. Wilson, "Optimizing the area under a receiver operating characteristic curve with application to landmine detection," *IEEE Trans. Geosci. Remote Sens.*, vol. 45, no. 2, pp. 389–397, Feb. 2007.
- [40] R. Mazhar and P. D. Gader, "EK-SVD: Optimized dictionary design for sparse representations," in *Proc. 19th Int. Conf. Pattern Recognit.*, Dec. 2008, pp. 1–4.
- [41] K. Ho, L. Collins, L. Huettel, and P. Gader, "Discrimination mode processing for EMI and GPR sensors for hand-held landmine detection," *IEEE Trans. Geosci. Remote Sens.*, vol. 42, no. 1, pp. 249–263, Jan. 2004.

**Raazia Mazhar** (S'08) received the B.Sc. degree in computer science from the National University of Computer and Emerging Sciences, Islamabad, Pakistan, in July 2003 and the Ph.D. degree in computer engineering from the University of Florida, Gainesville, in May 2009.

From July 2003 to June 2004, she was a Software Developer at Landmark Resources, Islamabad. She is currently with the Department of Computer and Information Science and Engineering, University of Florida, where he was a Research Assistant from January 2005 to December 2008. Her current research interests include machine learning, image and signal analysis, signal compression and approximation, and automated feature learning and clustering.



**Paul D. Gader** (M'87–SM'99) received the Ph.D. degree in mathematics from the University of Florida, Gainesville, in 1986.

He was a Senior Research Scientist at Honeywell's Systems and Research Center. He was a Research Engineer and the Manager at the Environmental Research Institute of Michigan. He was a Faculty Member at the University of Wisconsin and the University of Missouri. He is currently a Professor of computer and information science and engineering with the University of Florida. He led teams involved

in real-time, handwritten address recognition systems for the U.S. Postal Service and teams that devised and tested several real-time algorithms in the field for mine detection. His current research interests include landmine detection, handwriting recognition, machine learning and pattern recognition, image and signal analysis, fuzzy sets, random sets, and Choquet integration. He has over 340 technical publications in the areas of image and signal processing, applied mathematics, and pattern recognition, including over 68 refereed journal articles.



**Joseph N. Wilson** (M'05) received the B.S. degree in applied mathematics with emphasis on computer science from Florida State University, Tallahassee, in 1977 and the M.S. degree in applied mathematics and computer science and the Ph.D. degree in computer science from the University of Virginia, Charlottesville, in 1980 and 1985, respectively.

Since 1984, he has been a member of the Faculty with the Department of Computer and Information Science and Engineering, University of Florida, Gainesville, where he was an Associate Chair from 1994 to 2001. His current research interests include machine intelligence, image and signal processing, programming languages, and file systems.