Explosive Hazard Detection with Feature and Decision Level Fusion, Multiple Kernel Learning, and Fuzzy Integrals

Anthony J. Pinar[a], Joseph Rice[b], Timothy C. Havens[a,b], Matthew Masarik[c], and Joseph Burns[c]
[a]Department of Electrical and Computer Engineering
[b]Department of Computer Science
[c]Michigan Tech Research Institute
Michigan Technological University
Houghton, Michigan, 49931 USA
email: {ajpinar, jsrice, thavens, mpmasari, joseph.burns}@mtu.edu

Derek T. Anderson
Department of Electrical and Computer Engineering
Mississippi State University
Mississippi State, MS 39759, USA
e-mail: anderson@ece.msstate.edu

***Abstract*—Kernel methods for classification is a well-studied area in which data are implicitly mapped from a lower-dimensional space to a higher-dimensional space to improve classification accuracy. However, for most kernel methods, one must still choose a kernel to use for the problem. Since there is, in general, no way of knowing which kernel is the best, *multiple kernel learning* (MKL) is a technique used to learn the aggregation of a set of valid kernels into a single (ideally) superior kernel. The aggregation can be done using weighted sums of the pre-computed kernels, but determining the summation weights is not a trivial task. A popular and successful approach to this problem is *MKL-group lasso* (MKLGL), where the weights and classification surface are simultaneously solved by iteratively optimizing a min-max optimization until convergence. In this work, we compare the results of two previously proposed MKL algorithms to MKLGL in the context of explosive hazard detection using *ground penetrating radar* (GPR) data. The first MKL algorithm we employ is an ℓ_p-normed *genetic algorithm MKL* (GAMKL$_p$), which uses a genetic algorithm to learn the weights of a set of pre-computed kernel matrices for use with MKL classification. A second algorithm, called *decision-level fuzzy integral MKL* (DeFIMKL), is also employed, where a fuzzy measure with respect to the fuzzy Choquet integral is learned via quadratic programming, and the decision value—viz., the class label—is computed using the fuzzy Choquet integral aggregation. Experiments using government furnished GPR data show that these MKL algorithms can outperform MKLGL when applied to *support vector machine* (SVM)-based classification.**

***Index Terms*—multiple kernel learning, Choquet fuzzy integral, fuzzy measure, quadratic programming, genetic algorithm, support vector machine**

I. Introduction

Consider a set of numeric *feature-vector* data that has the form $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \subset \mathbb{R}^d$, where the coordinates of $\mathbf{x}_i$ provide feature values (e.g., bits per second, speed, volts, etc.) describing some object (e.g., a wireless sensor network node, traffic camera, or radar). We are also given a set of training labels for each feature vector, such that we have the pair $(\mathbf{y}, X)$, where $\mathbf{y} = (y_1, \ldots, y_n)^T$ and y_i is the label of ith object. Each y_i is associated with a respective feature vector $\mathbf{x}_i$. The classifier learning task is thus to learn some prediction function f, such that we can predict the label of feature-vectors, i.e., $y = f(\mathbf{x})$.

Most classifiers delineate the classes by finding some "best" decision boundary in the feature space. Perceptrons and linear *support vector machines* (SVMs) find hyperplanes. These classifiers are easy to train, often can be effective, and are computationally very efficient (the operational decision is just a single dot-product in the feature space). However, they are ineffective for classes that are not linearly separable, i.e., by a hyperplane. Hence, we will use kernel classifiers to non-linearly project the features into a high-dimensional space, where hyperplanes may be more easily found that serve as good decision boundaries.

Specifically, we will focus on *multiple kernel learning* (MKL) in this paper. As its name implies, MKL combines multiple kernels together to form a new kernel, and thus a new decision space. There are many works that discuss MKL [1]–[5], and nearly all of them rely on operations that aggregate kernels in ways that preserve symmetry and positive semi-definiteness, such as element-wise addition and multiplication. Most MKL algorithms learn a "best" kernel space in which to classify by learning respective weights on each component kernel. Details are contained in Section IV.

The MKL formulations reviewed in this paper have been previously proposed and applied to benchmark data sets [6], and focus on aggregation using a genetic algorithm and the Choquet *fuzzy integral* (FI) with respect to a *fuzzy measure* (FM) [7], respectively. Previous work showed that our genetic algorithm approach, GAMKL, is a generalized form of a previously proposed FI-based algorithm, *fuzzy integral: genetic algorithm* (FIGA) [4], [5]. It learns an MKL classifier using a genetic algorithm and a generalized p-norm weight domain, and aggregates kernel matrices at the feature-level, producing a new feature representation. We also employ the decision-level MKL algorithm called DeFIMKL. This algorithm learns

a FM with respect to the Choquet FI to fuse decisions from individual kernel classifiers. The FM is learned from training data with a regularized *quadratic program* (QP) approach [8].

The FI-based MKL approaches will be compared with a leading machine learning MKL method, called MKL *group lasso* (MKLGL) [2]; they are applied to a hazard detection dataset derived from government furnished ground penetrating radar (GPR) data discussed in Section V. We will also investigate the behavior of regularization on the results of DeFIMKL. Section II presents a short review of data fusion techniques, Section III introduces FMs and FIs, specifically the fuzzy Choquet integral, and Section IV reviews the MKL methods. Experimental results are presented in Section VI and we provide concluding remarks and ideas for future work in Section VII.

II. Data Fusion

Data fusion is a broad term for methods that use multiple sets of data, perhaps data from different sensors or the output of multiple processes applied to the same data set, to improve some performance metric from a baseline established using only *one* set [9]. It is a very broad area of study, and there exists a vast pool of literature relating to it; for a review of data fusion methods see [10] and [11]. Because of the breadth of the topic, we restrict this brief overview to the types of fusion techniques most related to the methods we employ.

Data fusion can be classified in many ways [12]–[14]. The taxonomy provided by Dasarathy in [13] is most appropriate here and describes five categories of data fusion; the categories that encompass our fusion methods are termed *feature in-feature out* (FEI-FEO) and *decision in-decision out* (DEI-DEO).

A. Feature In-Feature Out Fusion

FEI-FEO fusion is also known as feature fusion, on which many computer vision methods rely [15]–[18]. A popular and powerful method of feature fusion combines the features in a multidimensional feature space using kernel methods [19]–[22]. This allows the use of multiple kernels with classification, giving the advantage that particular kernels can exploit certain features better than other kernels. The SVM is a popular classifier for MKL classification; however, comparable results have been shown using a logistic regression-based classifier [23].

B. Decision In-Decision Out Fusion

DEI-DIO fusion is commonly referred to as decision fusion. This approach is very closely related to concept of ensemble learning, where the decisions from multiple classifiers are combined to determine the overall decision. Indeed, this is precisely what the DeFIMKL algorithm discussed in Section IV-C does. Due to the use of multiple classifiers, decision fusion is generally slower than feature fusion, which only requires one classifier [24].

Decision fusion can be done in two general ways: hard or soft. Hard decision fusion is done using the class labels from the ensemble of classifiers. A straightforward method of hard decision fusion is the majority vote approach. Soft decision fusion is performed using the posterior probabilities (or, more generally, the soft decision variables) from the classifier ensemble. A simple method in this case is to linearly combine the posterior probabilities [25]. For ensembles of fuzzy classifiers, the soft decision fusion approach can be used by aggregating the fuzzy class memberships determined by the classifiers [26].

III. Fuzzy Measures and Fuzzy Integrals

FIs and FMs have been proposed for many applications and for many types of data, from simple numeric data to intervals and type-2 fuzzy sets [27]–[37]. While manual specification of the FM works for small sets of sources (there are already 16 possible combinations of sources in the power set of 4 sources), manually specifying the values of the FM for large collections of sources is virtually impossible. Thus, automatic methods have been proposed, such as the Sugeno λ-measure [30] and the S-decomposable measure [38], which build the measure from the densities (the worth of individual sources), and genetic algorithm [4], [5], [29], [39], Gibbs sampling [40] and other learning methods [8], [41], [42], which build the measure by using training data. Other works [43]–[45] have proposed learning FMs that reflect trends in the data and have been specifically applied to crowd-sourcing, where the worth of individuals is not known, and is thus extracted from the data.

A. Fuzzy measures

A measurable space is the tuple (X, Ω), where X is a set and Ω is a Ω-algebra or set of subsets of X such that

P1. $X \in \Omega$;
P2. For $A \subseteq X$, if $A \in \Omega$, then $A^c \in \Omega$;
P3. If $\forall A_i \in \Omega$, then $\bigcup_{i=1}^{\infty} A_i \in \Omega$.

A FM is a set-valued function, $g : \Omega \to [0, 1]$, with the following properties:

P4. (Boundary conditions) $g(\emptyset) = 0$ and $g(X) = 1$;
P5. (Monotonicity) If $A, B \in \Omega$ and $A \subseteq B$, $g(A) \leq g(B)$.

If Ω is an infinite set, then there is also a third property guaranteeing continuity; in practice and in this paper, Ω is finite and thus this property is unnecessary. The FM values of the singletons, $g(\{x_i\}) = g^i$ are commonly called the *densities*. Figure 1 illustrates the lattice of a FM for the case of $n = 3$.

The arguably most popular FM is the Sugeno λ-measure, which has the attractive property of being able to be defined completely by the values of the densities. The λ-measure has the following additional property. For $A, B \in \Omega$ and $A \cap B = \emptyset$,

$$g_\lambda(A \cup B) = g_\lambda(A) + g_\lambda(B) + \lambda g_\lambda(A) g_\lambda(B), \tag{1a}$$

where it can be shown that λ can be found by solving [30]

$$\lambda + 1 = \prod_{i=1}^{n} \left(1 + \lambda g^i\right), \quad \lambda > -1. \tag{1b}$$

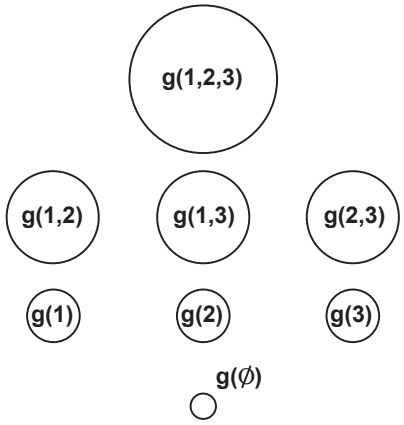

Fig. 1: Lattice of FM elements for $n = 3$. Monotonicity (P5) is illustrated by the size of each circle, i.e., $g(\{x_1\}) \leq g(\{x_1, x_2\})$ as $\{x_1\} \subset \{x_1, x_2\}$.

While fuzzy measures provide a way for quantifying the worth of combinations of sources, fuzzy integrals can be used to aggregate the information from these sources.

B. Fuzzy integrals

There are many forms of the FI; see [30] for detailed discussion. In practice, FIs are frequently used for evidence fusion [39], [46]–[49]. They combine sources of information by accounting for both the support of the question (the evidence) and the expected worth of each subset of sources (as supplied by the FM g). Here, we focus on the fuzzy Choquet integral, proposed by Murofushi and Sugeno [50], [51]. Let $h : X \to \mathbb{R}$ be a real-valued function that represents the evidence or support of a particular hypothesis.[1] The discrete (finite Ω) fuzzy Choquet integral is defined as

$$\int_C h \circ g = C_g(h) = \sum_{i=1}^{n} h(x_{\pi(i)}) \left[g(A_i) - g(A_{i-1}) \right], \quad (2)$$

where π is a permutation of X, such that $h(x_{\pi(1)}) \geq h(x_{\pi(2)}) \geq \ldots \geq h(x_{\pi(n)})$, $A_i = \{x_{\pi(1)}, \ldots, x_{\pi(i)}\}$, and $g(A_0) = 0$ [7], [33]. Detailed treatments of the properties of FIs can be found in [7], [33], [52]. We now move on to showing how MKL can be achieved using the FM and FI.

IV. Multiple Kernel Learning

Consider some non-linear mapping function $\phi : \mathbf{x}_i \to \phi(\mathbf{x}_i) \in \mathbb{R}^{D_K}$, where D_K is the dimensionality of the transformed feature vector $\phi(\mathbf{x}_i)$. For brevity, we will denote $\phi(\mathbf{x}_i)$ as ϕ_i, as ϕ_i can be considered the feature vector in the transformed space. With kernel algorithms, one does not need to explicitly transform $\mathbf{x}_i$, one simply needs to represent the dot product $\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j) = \kappa(\mathbf{x}_i, \mathbf{x}_j)$. The kernel function κ can take many forms, with the polynomial $\kappa(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j + 1)^p$ and *radial-basis-function* (RBF) $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma ||\mathbf{x}_i - \mathbf{x}_j||^2)$ being two of the most well known. Given a set of n feature-vectors X, one can thus construct an $n \times n$ kernel matrix $K = [K_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j)]_{n \times n}$. This kernel matrix K represents all pairwise dot products of the feature vectors in the transformed high-dimensional space $\mathcal{H}_K$—called the *Reproducing Kernel Hilbert Space* (RKHS).

There are many algorithms that use kernels to transform the input data to an appropriate and useful space; in this paper, we focus on kernel-based classification, such as the SVM [53], [54]. Multiple kernel algorithms, such as MKLGL [2], FIGA [4], and GAMKL$_p$ [6][2], take single kernel algorithms a step further by representing the feature-vector with multiple kernels and then combining them to produce a single decision output. The kernel combination can be computed in many ways, as long as the combination results in a Mercer kernel [55]. For the feature-level fusion algorithms in this paper, we will assume that the kernel $\mathcal{K}$ is composed by a weighted combination of pre-computed kernel matrices, i.e.,

$$\mathcal{K} = \sum_{k=1}^{m} \sigma_k K_k, \quad (3)$$

where there are m kernels and σ_k is the weight applied to the kth kernel. The domain of σ is very important and many MKL implementations only work for a single domain. For example, $\Delta_2 = \{\sigma \in \mathbb{R}^m : \|\sigma\|_2 = 1, \sigma_k \geq 0\}$ is the ℓ_2-norm MKL [1], [3]. MKLGL [2] uses a generalized MKL instantiation that allows for an ℓ_p-norm domain $\Delta_p = \{\sigma \in \mathbb{R}^m : \|\sigma\|_p = 1, \sigma_k \geq 0\}$, simultaneously learning σ and the parameters of an SVM on the resultant kernel $\mathcal{K}$. Similarly, the GAMKL$_p$ and DeFIMKL algorithms also use a generalized MKL implementation where σ can be restricted to the ℓ_p-norm domain.

A. The MKLGL algorithm [2]

As previously stated, MKLGL simultaneously learns the kernel weights, $\boldsymbol{\sigma}$, and the parameters of an SVM, $\boldsymbol{\alpha}$. These parameters are found by iteratively solving the generalized SVM optimization problem, which is a minimax problem. For more information on the kernel SVM formulation and its solution via MKLGL see [2], [56]. Algorithm 1 summarizes the MKLGL method.

B. The GAMKL$_p$ algorithm [6]

The GAMKL$_p$ algorithm uses a genetic algorithm to learn the optimal kernel weights, $\boldsymbol{\sigma}$, as outlined in Algorithm 2. The parameters shown in Algorithm 2 are those used in the experiments in this paper, though it may be necessary to modify them when using other datasets.

C. The DeFIMKL algorithm [6]

The formulation of the quadratic program used for the DeFIMKL algorithm, while not novel, does require considerable mathematical manipulation. The derivation of the algorithm is reviewed here.

[1] Generally, when dealing with information fusion problems it is convenient to have $h : X \to [0, 1]$, where each source is normalized to the unit-interval.

[2] Note that GAMKL$_p$ was shown to be a generalized version of FIGA in [6].

Algorithm 1: MKLGL Classifier Training

Data: $(\mathbf{x}_i, y_i)$ - feature vector and label pairs; K_k - kernel matrices; p - norm type

Result: $\boldsymbol{\alpha}$ - MKLGL classifier solution; $\boldsymbol{\sigma}$ - kernel weight vector

Initialize $\sigma_k = 1/m,\ k = 1, ..., m$ - set kernel weights equal

while *not converged* **do**

Solve unbalanced SKSVM for kernel matrix $K = \sum_{k=1}^{m} \sigma_k K_k$ for the optimal solution $\boldsymbol{\alpha}$

Update the kernel weights, σ_k using

$$\sigma_k = \frac{f_k^{2/(1+p)}}{\left(\sum_{k=1}^{m} f_k^{2p/(1+p)}\right)^{1/p}},\ k = 1, ..., m; \tag{4a}$$

$$f_k = \sigma_k^2 (\boldsymbol{\alpha} \circ \mathbf{y})^T K_k (\boldsymbol{\alpha} \circ \mathbf{y}). \tag{4b}$$

Algorithm 2: GAMKL$_p$ Classifier Training

Data: $(\mathbf{x}_i, y_i)$ - feature vector and label pairs; K_k - kernel matrices; p - norm type (regularization)

Result: $\boldsymbol{\sigma}$ - Kernel weights

Randomly initialize population of 31 kernel weight vector chromosomes.

for *25 generations* **do**

for *each chromosome* **do**

Compute the kernel using (3).

Compute the fitness as the 5-fold cross-validation kernel SVM accuracy to suppress the effects of overtraining.

Select parents via fitness proportional selection with elitism.

Generate offspring with 60% crossover rate.

Mutate offspring with 5% mutation rate.

Normalize each chromosome to lie in the ℓ_p-norm domain.

Select $\boldsymbol{\sigma}$ as the fittest chromosome in the last generation.

Let $f_k(\mathbf{x}_i)$ be the decision-value on feature-vector $\mathbf{x}_i$ produced by the kth classifier in an ensemble. The overall decision of the ensemble is computed by the Choquet integral, where the evidence h is the set of decisions by the classifier ensemble and g encodes the relative worth of each classifier in the ensemble. So, mathematically, the ensemble decision $f_g(\mathbf{x}_i)$ on feature-vector $\mathbf{x}_i$ with respect to the FM g is produced by

$$f_g(\mathbf{x}_i) = \sum_{k=1}^{m} f_{\pi(k)}(\mathbf{x}_i) \left[g(A_k) - g(A_{k-1}) \right], \tag{5}$$

where $A_k = \{f_{\pi(1)}(\mathbf{x}_i), \ldots, f_{\pi(k)}(\mathbf{x}_i)\}$, such that $f_{\pi(1)}(\mathbf{x}_i) \geq f_{\pi(2)}(\mathbf{x}_i) \geq \ldots \geq f_{\pi(m)}(\mathbf{x}_i)$. This is a generalized classifier fusion method that has been explored in many previous works [36], [48], [49], [57].

In [8], we proposed a method to learn the FM g from training data with a regularized *sum-of-squared error* (SSE) optimization, which we now briefly describe. Let the SSE be defined as

$$E^2 = \sum_{i=1}^{n} \left(f_g(\mathbf{x}_i) - y_i \right)^2. \tag{6}$$

It can be shown that (5), as a Choquet integral, can be reformulated as

$$f_g(\mathbf{x}_i) = \sum_{k=1}^{m} \left[f_{\pi(k)}(\mathbf{x}_i) - f_{\pi(k+1)}(\mathbf{x}_i) \right] g(A_k), \tag{7}$$

where $f_{\pi(m+1)} = 0$ [7]. The SSE can thus be expanded as

$$E^2 = \sum_{i=1}^{n} \left(H_{\mathbf{x}_i}^T \mathbf{u} - y_i \right)^2, \tag{8a}$$

where $\mathbf{u}$ is the lexicographically ordered FM g, i.e., $\mathbf{u} = (g(\{x_1\}), g(\{x_2\}), \ldots, g(\{x_1, x_2\}), g(\{x_1, x_3\}), \ldots, g(\{x_1, x_2, \ldots, x_m\}))$, and

$$H_{\mathbf{x}_i} = \begin{pmatrix} \vdots \\ f_{\pi(1)}(\mathbf{x}_i) - f_{\pi(2)}(\mathbf{x}_i) \\ \vdots \\ 0 \\ \vdots \\ f_{\pi(m)}(\mathbf{x}_i) - 0 \end{pmatrix}, \tag{8b}$$

where $H_{\mathbf{x}_i}$ is of size $(2^m - 1) \times 1$ and contains all the difference terms $f_{\pi(k)}(\mathbf{x}_i) - f_{\pi(k+1)}(\mathbf{x}_i)$ at the corresponding locations of A_k in $\mathbf{u}$. We can now fold out the squared term in (8a), producing

$$\begin{aligned} E^2 &= \sum_{i=1}^{n} \left(\mathbf{u}^T H_{\mathbf{x}_i} H_{\mathbf{x}_i}^T \mathbf{u} - 2 y_i H_{\mathbf{x}_i}^T \mathbf{u} + y_i^2 \right) \\ &= \mathbf{u}^T D \mathbf{u} + \mathbf{f}^T \mathbf{u} + \sum_{i=1}^{n} y_i^2, \\ D &= \sum_{i=1}^{n} H_{\mathbf{x}_i} H_{\mathbf{x}_i}^T, \quad \mathbf{f} = -\sum_{i=1}^{n} 2 y_i H_{\mathbf{x}_i}. \end{aligned} \tag{9}$$

Note that (9) is a quadratic function; hence, we can add in the constraints on $\mathbf{u}$, such that it represents a FM, producing a constrained QP. We can write the monotonicity constraint on $\mathbf{u}$, according to properties P4 and P5, as $C\mathbf{u} \leq 0$, where

$$C = \begin{pmatrix} \Psi_1^T \\ \Psi_2^T \\ \vdots \\ \Psi_{n+1}^T \\ \vdots \\ \Psi_{m(2^{m-1}-1)}^T \end{pmatrix} \tag{10}$$

and Ψ_1^T is a vector representation of the monotonicity constraint, $g\{x_1\} - g\{x_1, x_2\} \leq 0$. Hence, C is simply a matrix of $\{0, 1, -1\}$ values of size $(m(2^{m-1} - 1)) \times (2^m - 1)$. See

[8] for more details about the form of C. Thus, the full QP to learn the FM $\mathbf{u}$ is

$$\min_{\mathbf{u}} 0.5\mathbf{u}^T\hat{D}\mathbf{u} + \mathbf{f}^T\mathbf{u}, \quad C\mathbf{u} \leq \mathbf{0}, \quad (\mathbf{0}, 1)^T \leq \mathbf{u} \leq \mathbf{1}, \quad (11)$$

where $\hat{D} = 2D$. We will also test the performance of ℓ_2 and ℓ_1 regularization on the optimization at (11), i.e.,

$$\min_{\mathbf{u}} 0.5\mathbf{u}^T\hat{D}\mathbf{u} + \mathbf{f}^T\mathbf{u} + \lambda\|\mathbf{u}\|_p, \quad (12)$$

where $p = 1$ for ℓ_1 regularization and $p = 2$ for ℓ_2. Again, see [8] for more discussion on this topic. The QPs at (11) and (12) provide a method to learn the FM $\mathbf{u}$ (i.e., g) from training data. We now propose a method for using this learning method for ensemble learning with kernel SVMs.

We propose that each learner $f_k(\mathbf{x}_i)$ is a kernel classifier, each trained on a separate kernel K_k; here, we will use the SVM. The SVM classifier decision value is

$$\eta_k(\mathbf{x}) = \sum_{i=1}^{n} \alpha_{ik} y_i \kappa_k(\mathbf{x}_i, \mathbf{x}) - b_k, \quad (13)$$

which is essentially the distance of $\mathbf{x}$ from the hyperplane defined by the learned SVM model parameters, α_{ik} and b_k [53], [54]. Typically, the class label is then computed as $\text{sgn}\{\eta_k(\mathbf{x})\}$. One could use $f_k(\mathbf{x}) = \text{sgn}\{\eta_k(\mathbf{x})\}$ as the training input to the FM learning at (9), but this eliminates information about which kernel produces the largest class separation—essentially, the difference between $\eta_k(\mathbf{x})$ for classes labeled $y = +1$ and $y = -1$. Hence, we remap $\eta_k(\mathbf{x})$ onto the interval $[-1, +1]$, creating the inputs for learning by the sigmoid function,

$$f_k(\mathbf{x}) = \frac{\eta_k(\mathbf{x})}{\sqrt{1 + \eta_k^2(\mathbf{x})}}. \quad (14)$$

Thus, the training data for DeFIMKL are $(\{K_k = [\kappa_k(\mathbf{x}_i, \mathbf{x}_j)], \mathbf{f}_k(X)\}, \mathbf{y})$, $k = 1, \ldots, m$, where K_k are the kernel matrices for each kernel function κ_k, $\mathbf{f}_k(X) = (f_k(\mathbf{x}_1), \ldots, f_k(\mathbf{x}_n))^T$ are the remapped SVM decision values, and $\mathbf{y} = (y_1, \ldots, y_n)$ are the ground-truth labels of $X = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$, respectively. The output of the QP learner is the FM g. The training process is summarized in Algorithm 3.

Algorithm 3: DeFIMKL Classifier Training

Data: $(\mathbf{x}_i, y_i)$ - feature vector and label pairs; K_k - kernel matrices
Result: $\mathbf{u}$ - Lexicographically ordered fuzzy measure vector
for *each kernel matrix* **do**
  Compute the kernel SVM classifier decision values, η_k, as in (13).
  Remap the decision values onto the interval $[-1, +1]$ as f_k using (14).
Solve the minimization problem in (11) for the FM $\mathbf{u}$.

A new feature vector $\mathbf{x}$—from a test data set—can be classified by the trained algorithm with the following procedure:

1) Compute the SVM decision values $f_k(\mathbf{x})$ by using (13) and (14);
2) Apply the Choquet integral at (5) with respect to the learned FM g;
3) Compute the class label by $\text{sgn}\{f_g(\mathbf{x})\}$.

We now will apply the MKL algorithms discussed here to a pertinent defense and security problem, explosive hazard detection.

V. Explosive Hazard Detection Dataset

Our explosive hazard data set is composed of a collection of 1,955 11-dimensional feature vectors with class labels $\{-1, +1\}$, corresponding to true negatives and true positives, respectively. These feature vectors are computed by applying preprocessing and prescreener algorithms to GPR data, as discussed below.

A. Ground Penetrating Radar Data and Preprocessing

The raw data upon which this work is based was collected using a hand-held downward-looking GPR. The data acquired from this system include a time series of radar returns at a large number of discrete locations along a lane, the GPS locations of these radar returns, and a list of ground truth locations for a set of various types of explosive hazards. For more information regarding the GPR system and the data collected, see our previous work [56], [58].

Before applying the prescreening algorithm, the data collected by the GPR system was first preprocessed using *robust principal component analysis* (RPCA) [59]. The application of RPCA to GPR data has previously been shown to be fruitful since it decomposes the data into a low-rank component and a sparse component [58], [60], [61]. In this context the low-rank component corresponds to the slowly-varying background component of the radar returns, and the low-rank component represents outliers such as targets. Reference [58] contains additional information on how RPCA is applied to these data.

B. Prescreener and Feature Extraction

A simple energy-based prescreener using the RPCA sparse component was utilized to identify queue points to be investigated using the classifiers. The energy of each discrete radar return was found and a ground map was formed as shown in Figure 2. The prescreener then flags local maxima in the integrated energy ground map as queue points, and features are then extracted from those points. The features that were collected from the queue points for this experiment were based on energy and localized contrast. Specifically, for each queue point location, the features include

- the energy at the detection location;
- the energy in a disk of radius 20 cm in the integrated energy ground map;
- the ratios of energy in disks of radius 10, 20, 30, and 40 cm to the energy in a disk of radius 50 cm in the integrated energy ground map;

- the ratios of energy in circles of radius 10, 20, and 30% of total image size to total energy in the B-scan image[3].

C. Performance metric: NAUC

Results for explosive hazard detection are typically presented as *receiver operating characteristic* (ROC) curves. The horizontal axis, though typically labeled as a false alarm rate, is also directly proportional to a threshold against which the confidence of the hits is compared. As the threshold is increased, the relation of the probability of detection and *false alarm rate* (FAR) is shown. To quantify the results of a particular ROC, we find the *normalized area under the ROC* (NAUC) up to a FAR of 0.1 FA/m^2. This FAR rate was chosen to balance detection performance with practicality—a larger FAR will generally give better detection probability, however, the increased number of false alarms wastes time in the field. The NAUC equation is

$$NAUC = \frac{1}{0.1}\int_0^{0.1} P_D(FAR)dFAR, \tag{15}$$

where $P_D(FAR)$ is the probability of detection at false alarm rate, FAR. Equation (15) shows that the minimum value of the NAUC is zero if $P_D(FAR) = 0$ for $FAR \in [0, 0.1]$. It is also clear from Eq. (15) that an NAUC of 1 corresponds to perfect detection at zero FAR.

VI. Experiments and Results

Here we present the results of the GAMKL$_p$ and DeFIMKL algorithms after applying them to the GPR data set described in Section V using SVM classifiers; we use LIBSVM to implement the classifiers [62]. Their performance is presented alongside the results of the state-of-the-art MKLGL algorithm discussed in Section IV-A. Additionally, the results are compared with those of the prescreener such that the overall improvement can be evaluated.

Each experiment consists of 100 trials, where the results of these trials are statistically compared via a two-sample *t*-test at a 5% significance level. Including the standard deviation in the results highlights the sensitivity (variance) of each classifier to the selection of training data. In each trial the data set is partitioned into five partitions, each holding 20% of the data. The training/testing cycle is performed five times, where four partitions are used as training data and the remaining partition is used as the testing data; the testing results from each partition are combined to form the overall ROC for each trial and the NAUC is extracted as the performance metric.

Fifty RBF kernels are used in each algorithm with respective RBF widths σ logarithmically spaced on the interval $[10^{-2}, 10^{1.6}]$; the same RBF parameters are used for each algorithm.

[3]The B-scan image is a collection of individual radar returns surrounding the queue point location. This image essentially represents a vertical slice of earth at the queue point location. More details on these radar returns and B-scans can be found in [56], [58].

TABLE I: NAUCs and percentage improvement compared to the prescreener*.

	Results	
Algorithm	NAUC	% Improvement
Prescreener	0.204	-
MKLGL$_1$	0.438 (0.013)	115%
MKLGL$_2$	0.466 (0.013)	129%
GAMKL$_1$	**0.504 (0.012)**	**147%**
GAMKL$_2$	0.494 (0.013)	142%
DeFIMKL	0.350 (0.074)	72%
DeFIMKL$_1$ $\lambda = 1$	0.461 (0.041)	126%
DeFIMKL$_2$ $\lambda = 2$	0.486 (0.017)	138%

*Bold indicates the best performer according to a two-valued *t*-test at a 5% significance level..

A. Experiment 1

The first experiment was designed to compare the results of the MKL methods discussed in this paper with the prescreener and MKLGL algorithms. This experiment applies the different MKL classifiers to the same data partitions such that the results can be compared equally. Table I summarizes the average NAUCs from this experiment along with their improvement over the prescreener; the standard deviations are given in parentheses.

The results show that the GAMKL algorithm has superior performance when compared to the MKLGL algorithm and the regularized DeFIMKL algorithms' performance is comparable to MKLGL's performance, however, the ℓ_2-regularized DeFIMKL algorithm does beat MKLGL. The standard deviation of DeFIMKL$_2$ is marginally higher than that of MKLGL, suggesting that the DeFIMKL training is more closely dependent on the selection of training data and thus more susceptible to overtraining. This conclusion is further supported by the relatively large standard deviation exhibited by the unregularized DeFIMKL algorithm, which is to be expected since regularization was not employed to suppress the possibility of overtraining (i.e., classifier variance). GAMKL, on the other hand, has essentially equivalent classifier variance, i.e., it is just as susceptible to overtraining as MKLGL.

B. Experiment 2

A second experiment was performed with the DeFIMKL algorithm to observe the effects of the regularization parameter λ. This experiment applies the regularized DeFIMKL algorithms to the data while varying λ over the range $[0, 10]$. Figure 3 summarizes the results of this experiment.

The trend of the plot shows the importance of including regularization with the DeFIMKL algorithm, since both DeFIMKL$_1$ and DeFIMKL$_2$ benefit by using a nonzero λ. However, the average NAUC generally decreases as λ is increased. Furthermore, the standard deviation of the DeFIMKL$_2$ results increases nearly consistently with increasing λ, though the trend is opposite with the DeFIMKL$_1$ standard deviation.

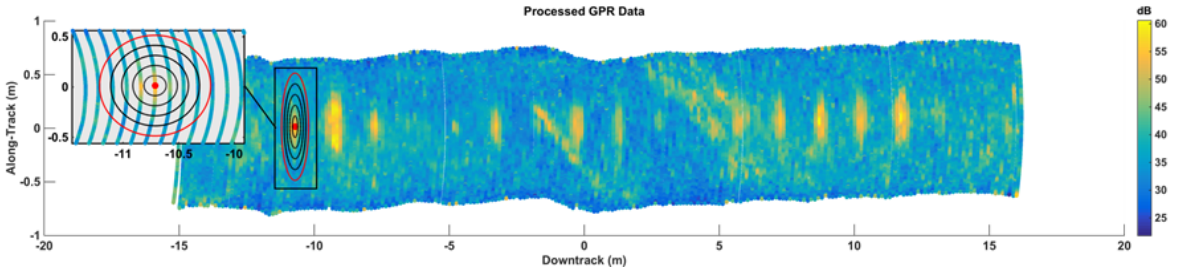

Fig. 2: Integrated energy ground map and an example queue point with 10, 20, 30, and 40 cm disks.

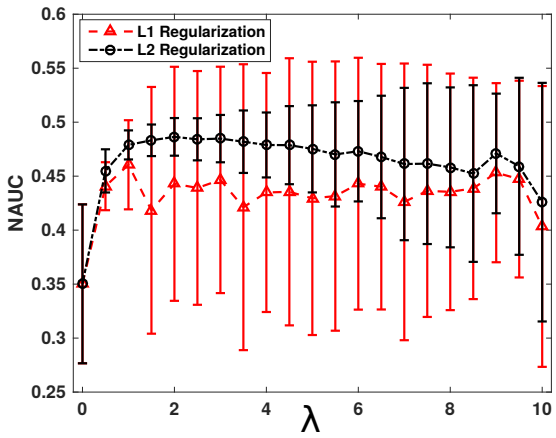

Fig. 3: DeFIMKL performance using regularization. Error bars indicate $\pm$ one standard deviation.

VII. Conclusion

This paper applies a feature-level fusion algorithm, GAMKL$_p$, and a decision-level fusion algorithm, DeFIMKL, to a dataset derived from ground penetrating radar for explosive hazard detection. GAMKL$_p$ uses a genetic algorithm to find the multiple kernel mixing coefficients, $\boldsymbol{\sigma}$, and is generalized to allow $\boldsymbol{\sigma}$ to lie in the ℓ_p-norm domain, Δ_p. The DeFIMKL algorithm aggregates kernels through the use of the Choquet fuzzy integral with respect to a fuzzy measure learned by a regularized quadratic programming approach. We use MKLGL as the benchmark MKL algorithm, and show that both GAMKL$_p$ and DeFIMKL can outperform MKLGL.

A. Future Work

We have been working on a feature-level method for aggregating the kernels K_k with a non-linear fuzzy integral. The main goal is to preserve the ability of the fuzzy integral to produce non-linear aggregations of the individual kernels, while ensuring that the result is a Mercer kernel. In order to achieve this, one must develop a way of sorting the kernel matrix terms in the Choquet integral (and not just once with the base-learner training data accuracy, as does FIGA) and still aggregate with a Mercer kernel preserving operation.

Acknowledgment

This work is funded in part by the Army Research Office (W911NF-16-1-0017). Dr. Anderson is partially funded by Army Research Office Grant W911NF-14-1-0673. Superior, a high performance computing cluster at Michigan Technological University, was used in obtaining some of the results presented in this publication.

References

[1] M. Kloft, U. Brefeld, P. Laskov, and S. Sonnenburg, "Non-sparse multiple kernel learning," 2008.

[2] Z. Xu, R. Jin, H. Yang, I. King, and M. Lyu, "Simple and efficient multiple kernel learning by group lasso," in *Proc. Int. Conf. Machine Learning*, 2010, pp. 1175–1182.

[3] C. Cortes, M. Mohri, and A. Rostamizadeh, "ℓ_2 regularization for learning kernels," in *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*. AUAI Press, 2009, pp. 109–116.

[4] L. Hu, D. T. Anderson, and T. C. Havens, "Multiple kernel aggregation using fuzzy integrals," in *IEEE International Conference on Fuzzy Systems*. IEEE, 2013, pp. 1–7.

[5] L. Hu, D. Anderson, T. Havens, and J. Keller, "Validity of different fuzzy integrals and representations for multiple kernel aggregation," in *Proc. Int. Conf. Info. Processing and Management of Uncertainty in Knowledge-Based Systems*, 2014.

[6] A. Pinar, T. C. Havens, D. T. Anderson, and L. Hu, "Feature and decision level fusion using multiple kernel learning and fuzzy integrals," in *IEEE International Conference on Fuzzy Systems*, Aug 2015, pp. 1–7.

[7] M. Sugeno, "Theory of fuzzy integrals and its applications," Ph.D. dissertation, Tokyo Institute of Technology, 1974.

[8] D. Anderson, S. Price, and T. Havens, "Regularization-based learning of the choquet integral," in *IEEE International Conference on Fuzzy Systems*, July 2014, pp. 2519–2526.

[9] D. L. Hall and J. Llinas, "An introduction to multisensor data fusion," *Proceedings of the IEEE*, vol. 85, no. 1, pp. 6–23, 1997.

[10] B. Khaleghi, A. Khamis, F. O. Karray, and S. N. Razavi, "Multisensor data fusion: A review of the state-of-the-art," *Information Fusion*, vol. 14, no. 1, pp. 28–44, 2013.

[11] F. Castanedo, "A review of data fusion techniques," *The Scientific World Journal*, vol. 2013, 2013.

[12] H. F. Durrant-Whyte, "Sensor models and multisensor integration," *The International Journal of Robotics Research*, vol. 7, no. 6, pp. 97–113, 1988.

[13] B. V. Dasarathy, "Sensor fusion potential exploitation-innovative architectures and illustrative applications," *Proceedings of the IEEE*, vol. 85, no. 1, pp. 24–38, 1997.

[14] R. C. Luo, C.-C. Yih, and K. L. Su, "Multisensor fusion and integration: approaches, applications, and future research directions," *Sensors Journal, IEEE*, vol. 2, no. 2, pp. 107–119, 2002.

[15] J. K. Aggarwal, *Multisensor fusion for computer vision*. Springer Science & Business Media, 2013, vol. 99.

[16] F. S. Khan, J. Van de Weijer, and M. Vanrell, "Top-down color attention for object recognition," in *IEEE 12th International Conference on Computer Vision*. IEEE, 2009, pp. 979–986.

[17] P. Natarajan, S. Wu, S. Vitaladevuni, X. Zhuang, S. Tsakalidis, U. Park, R. Prasad, and P. Natarajan, "Multimodal feature fusion for robust event detection in web videos," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2012, pp. 1298–1305.

[18] G. Nützi, S. Weiss, D. Scaramuzza, and R. Siegwart, "Fusion of imu and vision for absolute scale estimation in monocular slam," *Journal of intelligent & robotic systems*, vol. 61, no. 1-4, pp. 287–299, 2011.

[19] Y.-R. Yeh, T.-C. Lin, Y.-Y. Chung, and Y.-C. F. Wang, "A novel multiple kernel learning framework for heterogeneous feature fusion and variable selection," *IEEE Transactions on Multimedia*, vol. 14, no. 3, pp. 563–574, 2012.

[20] P. Gehler and S. Nowozin, "On feature combination for multiclass object classification," in *IEEE 12th International Conference on Computer Vision*. IEEE, 2009, pp. 221–228.

[21] L. Cao, J. Luo, F. Liang, and T. S. Huang, "Heterogeneous feature machines for visual recognition," in *IEEE 12th International Conference on Computer Vision*. IEEE, 2009, pp. 1095–1102.

[22] J. Yang, Y. Li, Y. Tian, L. Duan, and W. Gao, "Group-sensitive multiple kernel learning for object categorization," in *IEEE 12th International Conference on Computer Vision*. IEEE, 2009, pp. 436–443.

[23] B. Fernando, E. Fromont, D. Muselet, and M. Sebban, "Discriminative feature fusion for image classification," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2012, pp. 3434–3441.

[24] C. H. Chan, M. A. Tahir, J. Kittler, and M. Pietikainen, "Multiscale local phase quantization for robust component-based face recognition using kernel fusion of multiple descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 5, pp. 1164–1177, 2013.

[25] Y. Zhang, H. L. Yang, S. Prasad, E. Pasolli, J. Jung, and M. Crawford, "Ensemble multiple kernel active learning for classification of multisource remote sensing data," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 2, pp. 845–858, 2015.

[26] B. Bigdeli, F. Samadzadegan, and P. Reinartz, "Fusion of hyperspectral and lidar data using decision template-based fuzzy multiple classifier system," *International Journal of Applied Earth Observation and Geoinformation*, vol. 38, pp. 309–320, 2015.

[27] D. Anderson, T. Havens, C. Wagner, J. Keller, M. Anderson, and D. Wescott, "Extension of the fuzzy integral for general fuzzy set-valued information," *IEEE Transactions on Fuzzy Systems*, vol. 22, no. 6, pp. 1625–1639, Dec 2014.

[28] C. Wagner, D. Anderson, and T. Havens, "Generalization of the fuzzy integral for discontinuous interval- and non-convex interval fuzzy set-valued inputs," in *IEEE International Conference on Fuzzy Systems*, July 2013, pp. 1–8.

[29] D. T. Anderson, J. M. Keller, and T. C. Havens, "Learning fuzzy-valued fuzzy measures for the fuzzy-valued sugeno fuzzy integral," in *Computational Intelligence for Knowledge-Based Systems Design*. Springer, 2010, pp. 502–511.

[30] M. Grabisch, *Fuzzy Measures and Integrals: Theory and Applications*, M. Sugeno and T. Murofushi, Eds. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2000.

[31] D. Zhang and Z. Wang, "Fuzzy integrals of fuzzy-valued functions," *Fuzzy Sets and Systems*, vol. 54, no. 1, pp. 63–67, 1993.

[32] R. Yang, Z. Wang, P.-A. Heng, and K.-S. Leung, "Fuzzified choquet integral with a fuzzy-valued integrand and its application on temperature prediction," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 38, no. 2, pp. 367–380, 2008.

[33] M. Grabisch, H. T. Nguyen, and E. A. Walker, *Fundamentals of uncertainty calculi with applications to fuzzy inference*. Springer Science & Business Media, 2013, vol. 30.

[34] T. Havens, D. Anderson, and J. Keller, "A fuzzy choquet integral with an interval type-2 fuzzy number-valued integrand," in *IEEE International Conference on Fuzzy Systems*, July 2010, pp. 1–8.

[35] D. Anderson, T. Havens, C. Wagner, J. Keller, M. Anderson, and D. Wescott, "Sugeno fuzzy integral generalizations for sub-normal fuzzy set-valued inputs," in *Fuzzy Systems (FUZZ-IEEE), 2012 IEEE International Conference on*, June 2012, pp. 1–8.

[36] X. Wang, A. Chen, and H. Feng, "Upper integral network with extreme learning mechanism," *Neurocomputing*, vol. 74, no. 16, pp. 2520–2525, 2011.

[37] X. Liang, C. Wei, and Z. Chen, "An intuitionistic fuzzy weighted owa operator and its application," *International Journal of Machine Learning and Cybernetics*, vol. 4, no. 6, pp. 713–719, 2013.

[38] D. J. Dubois, *Fuzzy sets and systems: theory and applications*. Academic press, 1980, vol. 144.

[39] M. F. Anderson, D. T. Anderson, and D. J. Wescott, "Estimation of adult skeletal age-at-death using the sugeno fuzzy integral," *American journal of physical anthropology*, vol. 142, no. 1, pp. 30–41, 2010.

[40] A. Mendez-Vazquez and P. Gader, "Sparsity promotion models for the choquet integral," in *IEEE Symposium on Foundations of Computational Intelligence*. IEEE, 2007, pp. 454–459.

[41] J. Keller and J. Osborn, "A reward/punishment scheme to learn fuzzy densities for the fuzzy integral," in *International Fuzzy Systems Association World Congress*, 1995, pp. 97–100.

[42] ——, "Training the fuzzy integral," *International Journal of Approximate Reasoning*, vol. 15, no. 1, pp. 1–24, 1996.

[43] C. Wagner and D. T. Anderson, "Extracting meta-measures from data for fuzzy aggregation of crowd sourced information," in *IEEE International Conference on Fuzzy Systems*. IEEE, 2012, pp. 1–8.

[44] T. C. Havens, D. T. Anderson, C. Wagner, H. Deilamsalehy, and D. Wonnacott, "Fuzzy integrals of crowd-sourced intervals using a measure of generalized accord," in *IEEE International Conference on Fuzzy Systems*. IEEE, 2013, pp. 1–8.

[45] T. Havens, D. Anderson, and C. Wagner, "Data-informed fuzzy measures for fuzzy integration of intervals and fuzzy numbers," *IEEE Transactions on Fuzzy Systems*, vol. PP, no. 99, pp. 1–1, 2014.

[46] M. Grabisch, "Fuzzy integral for classification and feature extraction," in *Fuzzy Measures and Integrals: Theory and Applications*. Springer-Verlag New York, Inc., 2000, pp. 415–434.

[47] J. Keller, P. Gader, and A. Hocaoglu, "Fuzzy integral in image processing and recognition," in *Fuzzy Measures and Integrals: Theory and Applications*. Springer-Verlag New York, Inc., 2000, pp. 435–466.

[48] S. Auephanwiriyakul, J. M. Keller, and P. D. Gader, "Generalized choquet fuzzy integral fusion," *Information Fusion*, vol. 3, no. 1, pp. 69–85, 2002.

[49] H. Tahani and J. M. Keller, "Information fusion in computer vision using the fuzzy integral," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 20, no. 3, pp. 733–741, 1990.

[50] G. Choquet, "Theory of capacities," in *Annales de l'institut Fourier*, vol. 5. Institut Fourier, 1954, pp. 131–295.

[51] T. Murofushi and M. Sugeno, "An interpretation of fuzzy measures and the choquet integral as an integral with respect to a fuzzy measure," *Fuzzy sets and Systems*, vol. 29, no. 2, pp. 201–227, 1989.

[52] M. Grabisch, "Fuzzy integral in multicriteria decision making," *Fuzzy sets and Systems*, vol. 69, no. 3, pp. 279–298, 1995.

[53] C. Cortes and V. N. Vapnik, "Support-vector networks," vol. 20, no. 3, 1995, pp. 273–297.

[54] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*. ACM, 1992, pp. 144–152.

[55] J. Mercer, "Functions of positive and negative type and their connection with the theory of integral equations," vol. 209, 1909, pp. 441–458.

[56] A. Pinar, M. Masarik, T. C. Havens, J. Burns, B. Thelen, and J. Becker, "Approach to explosive hazard detection using sensor fusion and multiple kernel learning with downward-looking GPR and emi sensor data," in *Proc. SPIE*, vol. 9454, 2015, pp. 94 540B–94 540B–20.

[57] J. Zhai, H. Xu, and Y. Li, "Fusion of extreme learning machine with fuzzy integral," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 21, no. 2, pp. 23–34, 2013.

[58] M. P. Masarik, J. Burns, B. T. Thelen, J. Kelly, and T. C. Havens, "GPR anomaly detection with robust principal component analysis," in *Proc. SPIE*, vol. 9454, 2015, pp. 945 414–945 414–11.

[59] E. J. Candes, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *Journal of the ACM*, vol. 58, no. 3, May 2011.

[60] D. Kalika, M. T. Knox, L. M. Collins, P. A. Torrione, and K. D. Morton, "Leveraging robust principal component analysis to detect buried explosive threats in handheld ground-penetrating radar data," in *SPIE Defense+ Security*. International Society for Optics and Photonics, 2015, pp. 94 541D–94 541D.

[61] A. Pinar, T. C. Havens, J. Rice, M. Masarik, J. Burns, and B. Thelen, "A comparison of robust principal component analysis techniques for buried object detection in downward looking gpr sensor data," in *SPIE Defense+ Security*. International Society for Optics and Photonics, 2016, pp. 98 230T–98 230T.

[62] C. C. Chang and C. J. Lin, "LIBSVM: a library for support vector machines," *ACM Trans. Intell. Sys. Tech.*, vol. 2, no. 27, pp. 1–27, 2011.