# Multiple Kernel Aggregation Using Fuzzy Integrals


Lequn Hu,
*Student Member, IEEE*
Mississippi State University
Electrical and Computer Engineering
Mississippi State, MS 39759
Email: lh432@msstate.edu

Derek T. Anderson,
*Member, IEEE*
Mississippi State University
Electrical and Computer Engineering
Mississippi State, MS 39759
Email: anderson@ece.msstate.edu

Timothy C. Havens,
*Senior Member, IEEE*
Michigan Technological University
Electrical and Computer Engineering
Houghton, MI 49931
Email: thavens@mtu.edu



***Abstract*—The so-called *kernel-trick* is a well-known method for mapping data in a lower dimensional space into a higher dimensional space to measure the similarity (inner product) of the data elements without ever explicitly performing the mapping. The hope is to induce an improved feature space in which to carry out pattern analysis. However, important questions remain, such as i) what is the *best* kernel, and ii) do some features or sensors require different kernels? One elegant way to address these problems is *multiple kernel* (MK) aggregation. To date, the research on MKs has predominately studied linear aggregation of kernels, namely weighted sums, e.g., conic and convex sums. In this paper, we propose a new method for kernel aggregation, *fuzzy integral* aggregation of MKs (FI-MK). We study different FI formulations to determine which ensures production of an aggregated kernel that is a valid Mercer kernel. We show that the *Choquet integral* (CI) achieves this goal for matrix-wise aggregation. We leverage our theoretical results to propose a genetic algorithm-based classification scheme called FIGA. Experiments on publicly available data sets are provided that demonstrate our FIGA algorithm produces superior results in the context of *support vector machine* (SVM)-based classification.**



***Keywords*—multiple kernel learning, non-linear aggregation, Choquet fuzzy integral, fuzzy measure;**


TABLE I. LIST OF ACRONYMS AND NOTATION

| | |
|---|---|
| SVM | support vector machine |
| MK | multi-kernel |
| MKL | MK learning |
| FI | fuzzy integral |
| SI | Sugeno fuzzy integral |
| CI | Choquet fuzzy integral |
| GA | genetic algorithm |
| FM | fuzzy measure |

| | |
|---|---|
| $X$ | sources |
| $x_i$ | single source |
| $\mathcal{X}$ | feature set |
| $\mathbf{x}_i$ | feature vector of $x_i$ |
| $K$ | kernel matrix |
| $\mathcal{K}$ | aggregate kernel matrix |
| $\kappa$ | kernel function |
| $h$ | hypothesis support |
| $g$ | fuzzy measure |
| $n$ | number of sources |
| $m$ | number of kernels |
| $[n]$ | $\{1, 2, \ldots, n\}$ |

## I. INTRODUCTION

A number of modern pattern analysis techniques make use of the so-called *kernel-trick*. Examples include, to name a few, *support vector machines* (SVMs) [1] and kernel clustering [2, 3]. The basic idea is to map data in a lower dimensional space into a higher dimensional space to measure the similarity (inner product) of the data elements without ever explicitly performing the mapping. The hope is to induce an improved feature space in which to carry out pattern analysis. To date, a massive number of works have appeared regarding the design of kernel functions, with specific application to SVMs [4–6]. However, a number of important questions remain. For example, for a given task, what is the *best* kernel? If different sensors, features or sources are combined, is a single kernel well-suited to transform all or is a different kernel required for each? If some sources provide more valuable information than others, how is this modeled in terms of kernel theory?

An elegant way to address these questions is *multiple kernel* (MK) aggregation. To date, researchers in machine learning have focused predominantly on linear aggregation of base (valid) Mercer kernels, namely weighted sum forms, e.g., conic and convex sums. Limitations of current linear approaches include occasional lack of interpretability—most are well-suited for data-driven applications but not human specification—and limited capability to exploit all information available in terms of the complex ways in which sources sometimes interact.

In this article, we explore the utility of the *fuzzy integral* (FI) for MK aggregation. We prove that some FIs, e.g., the *Sugeno FI* (SI), do not produce valid MKs—viz., a multi-kernel aggregation that is a Mercer kernel—while the *Choquet FI* (CI) does indeed produce valid MKs (if sorting is addressed properly). Specifically, the difference-in-measure weighted sum CI formulation is particularly appealing as it yields the familiar linear convex sum form. This also comes with an advantage in terms of interpreting how the FI aggregation is effecting the MK mathematics. Another advantage of the FI approach is that it has a great deal of flexibility in terms of specifying, versus learning, a specific type of MK aggregation. For example, production of a number of familiar order statistics (mean, max, etc.) and linguistic concepts (e.g., *most*, *some*, *soft max*, etc.). This is elaborated on further in Section II.

The overall idea with MK approaches is that different optimal kernels exist and can be identified or learned for the different information sources. In essence, this approach will enable us to map many forms of heterogeneous data (multi-source, mixed-type and mixed-uncertainty) data into a homogeneous (implicit) kernel space. Pattern analysis, e.g., classification and clustering, can then be performed in this fused space. An advantage is that the pattern analysis algorithms do not have to address the heterogeneous data problem necessarily, that is the job of the MK framework. Figure 1 illustrates our general framework.

Of particular interest to our group is multi-sensor data fusion and its embedding in automated decision making and pattern analysis. An example is our work in the detection of explosive hazardous materials using forward-looking infrared and ground-penetrating radar sensors [7–10]. In this context,

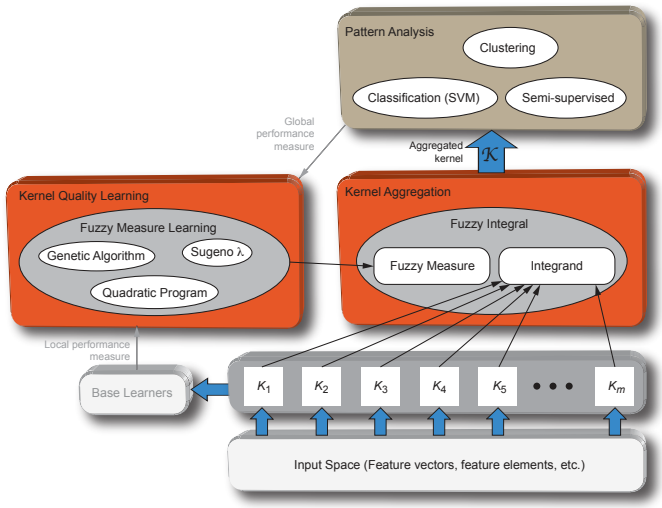


Fig. 1. Proposed FI-MKL framework. Components in red boxes are the focus of this article. Components in white ellipses are example algorithms that could be used in Pattern Analysis and MKL Weight Learning blocks. MKL Weight Learning can either use a global performance metric from the Pattern Analysis block or some local performance metric from a collection of base learners, where each base learner uses one kernel from the collection.

our goal was to fuse features within and across sensors.

This article is structured as follows. In Section II relevant fundamental concepts for the fuzzy measure and FI are summarized. The concept of kernels and MK is introduced in Section III and related work is discussed. Next, the validity of the FI, particularly the CI, for MK aggregation is studied in Section IV. We also propose a *genetic algoritm* (GA)-based classification scheme, called FIGA, that demonstrates our framework. Experimental results in the context of FIGA-learned FMs with an SVM classifier are reported in Section V and we briefly summarize in VI.

## II. Fuzzy Measure and Fuzzy Integral

The fusion of information using the *classical* FI (Sugeno or Choquet) has a rich history. Much of the theory and several applications can be found in [11, 12]. With respect to this problem, we consider a finite set of $n$ sources of information $X = \{x_1, ..., x_n\}$ and a function that maps $X$ into some domain (initially $[0, 1]$) that represents the partial support of a hypothesis from the standpoint of each source of information. Depending on the problem domain, $X$ can be a set of experts, sensors, features, pattern recognition algorithms, etc. The hypothesis is usually thought of as an alternative in a decision process or a class label in pattern recognition. Both Choquet and Sugeno integrals take partial support for the hypothesis from the standpoint of each source of information and fuse it with the (perhaps subjective) worth (or reliability) of each subset of $X$ in a non-linear fashion. This worth is encoded in a *fuzzy measure* (FM) [13]. Initially, the function $h : X \to [0, 1]$ and the FM $g : 2^X \to [0, 1]$ were designed to take real number values in $[0, 1]$. Certainly, the output range for both the support function and FM can be (and have been) defined more generally, but it is convenient to think of them on $[0, 1]$ for confidence fusion. We now review FMs and FIs.

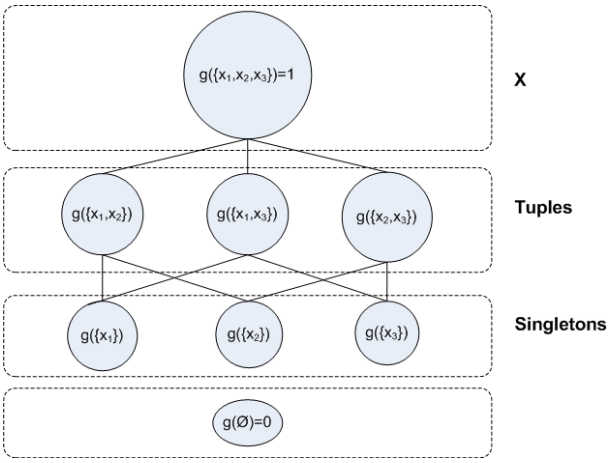


Fig. 2. Lattice of coefficients for a FM with $n = 3$. Monotonicity is presented by circle size, i.e, $g(\{x_1\}) \leq g(\{x_1, x_2\})$, as $\{x_1\} \subset \{x_1, x_2\}$.

### A. Fuzzy Measure

Measures are a fundamental concept in mathematics, especially as it relates to integrals with respect to a measure. A key property of FMs is that they require the property of monotonicity with respect to set inclusion, a far weaker property than the additive property of a probability measure. A measurable space is the tuple $(X, \Omega)$, where X is a set and $\Omega$ is a $\sigma$-algebra or set of subsets of X such that:

P1. $X \in \Omega$;

P2. Let $A \subseteq X$, if $A \in \Omega$ then $A^c \in \Omega$;

P3. If $A_i \in \Omega$, then $\bigcup_{i=1}^{\infty} A_i \in \Omega$.

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

P4. (Boundary conditions) $g(\phi) = 0$ and $g(X) = 1$;

P5. (Monotonicity) If $A, B \in \Omega$ and $A \subseteq B$, $g(A) \leq g(B)$.

Note, if $\Omega$ is an infinite set, a third condition guaranteeing continuity is required, but this is a moot point for finite $\Omega$. The value $g^i = g(\{x_i\})$ is generally referred to as a density (worth of the singletons). Figure 2 is an illustration of a FM lattice for the case of $n = 3$.

The FM can be computed or specified a number of ways. When $n$ is small, it is possible for a person to specify the measure. As $n$ grows, this become more difficult. A FM has $(2^n - 2)$ free parameters. Thus, for a relatively small number of sources, such as $n = 10$, one is confronted with finding $(2^{10} - 2)$ values! A number of work [14, 15] involves learning the measure from data. In 1974, Sugeno created a way to automatically generate the entire lattice based on just the densities, thus $(2^n - 2 - n)$ values. The Sugeno $\lambda$-fuzzy measure has the following additional property:

P6. If $A, B \in \Omega$ and $A \cap B = \phi$,

$$g(A \cup B) = g(A) + g(B) + \lambda g(A)g(B) \quad (1)$$

Sugeno proved that a unique $\lambda$ can be found by solving,

$$\lambda + 1 = \prod_{i=1}^{n}(1 + \lambda g^i), \quad \lambda > 1. \quad (2)$$

Note, when $\lambda = 0$, the measure is a probability measure. Now we turn to the formation of the FI.

### B. Fuzzy Integral

Many mathematical formulations have been proposed for the FI [13, 16–18]. These formulations have various advantages, e.g., generalizability, differentiability, and ability to address different types of uncertain data. In this paper, we stick to the conventional fuzzy SI and CI.

Given a finite set $X$, a FM, $g : 2^X \to [0, 1]$, and a partial support function, $h : X \to [0, 1]$, the finite CI of $h$ (integrand) with respect to $g$ (measure), is

$$\int_c h \circ g = \sum_{i=1}^{n} \omega_i h(x_{(i)}), \tag{3}$$

where $\omega_i = (G_{(i)} - G_{(i-1)})$, $G_{(i)} = g(\{x_{(1)}, ..., x_{(i)}\})$, $G_{(0)} = 0$, $h(x_i)$ is the strength in the hypothesis from source $x_i$, and $(i)$ is a sorting on $X$ such that $h(x_{(1)}) \geq h(x_{(2)}) \geq \ldots \geq h(x_{(n)})$. This form, $\int_c h \circ g$, is known as the difference-in-measure CI.

Another frequently encountered formulation of the FI considered here is the original discrete SI,

$$\int_s h \circ g = \bigvee_{i=1}^{n} (h(x_{(i)}) \wedge G_{(i)}), \tag{4}$$

where $\vee$ is a t-conorm (e.g., max) and $\wedge$ is a t-norm (e.g., min). Sugeno originally proposed (4) as the FI and later introduced the CI formulation at (3). The SI has the advantage that it can be interpreted as the "best pessimistic agreement" (max of mins). However, the max and min forms are often difficult to work with, i.e., differentiate. The CI has the advantage that it is differentiable. Also, if one uses a probability measure with CI then the classical Lebesgue is recovered, which is not the case for the SI. Regardless, we introduce the SI here and explore whether it can be used to aggregate MKs.

On a final (important) related note, Keller et al. [12] showed that the FI produces different well-known order statistics and functions depending on the choice of FM. Examples include, the max, min, mean, and median. When sets of equal cardinality have equal measure value in the FM, one obtains an *ordered-weighted-average* (OWA). Many linguistic schemes have also been put forth, e.g., *soft max* and *few*. The point is, the FI offers a great deal of representational power that a designer can specify or learning algorithm can compute.

Note that, herein, we will employ the FI not to combine sources, per se, but to combine kernels that describe those sources. Hence, the operations in (3) and (4) will be indexed over $m$ kernels, where each kernel defines a different feature space for the $n$ sources of information. This is unique because the FI is not being used in the conventional way, i.e. to fuse numbers, intervals, or fuzzy sets. Instead, our goal is to *fuse features spaces* for pattern analysis. Specifically, we seek to acquire a single improved mapping or facilitate a unique way to aggregate heterogeneous data. As we will show in Section IV, the form of the FI used is an important consideration in terms of whether it produces a valid MK. First, we discuss kernels and MKs.

## III. Kernels

In this section, we review basic concepts and definitions of kernels that are necessary to support proofs of how FIs can be used to combine kernels into valid MKs. Thus, it is easy to extend our procedure to show validity for any application that makes use of kernels, e.g., SVMs and kernel clustering. First, assume that from each source of information in $X$, we are measuring a feature vector $\mathbf{x}$, where $\mathbf{x}_i$ is a feature vector that describes the source $x_i$. The set of all feature vectors is $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$.[1]

**Definition 1 (kernel).** Suppose we are given a feature mapping $\phi : \mathcal{R}^d \to \mathcal{R}^{\mathcal{H}}$, where d is the dimensionality of the input space and $\mathcal{R}^{\mathcal{H}}$ is some (higher-)dimensional space, which is called the *Reproducing Kernel Hilbert Space* (RKHS). A kernel is the inner product function $\kappa : \mathcal{R}^d \times \mathcal{R}^d \to \mathcal{R}$, which represents the inner-product in $\mathcal{R}^{\mathcal{H}}$,

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle, \quad \forall \mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}. \tag{5}$$

**Definition 2 (Mercer kernel).** Assume a kernel function $\kappa$ and finite data $\mathcal{X} = \{\mathbf{x}_1, ..., \mathbf{x}_n\}$. The $n \times n$ matrix $K = [K_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j)]$, $i, j = [n]$, is a Gram matrix of inner products. Therefore, $K$ is symmetric and *positive semi-definite* (PSD), $\mathbf{x}_i^T K \mathbf{x}_i \geq 0$, $\forall \mathbf{x}_i \in \mathcal{X}$. Note, since $K$ is PSD, all eigenvalues are non-negative.

For readability, from now on we will denote features that are projected into the RKHS $\mathcal{R}^{\mathcal{H}}$ as $\phi_i$, where $\phi_i$ is equivalent to $\phi(\mathbf{x}_i)$. Furthermore, we will assume that each source in $X$ has either multiple features extracted from it or multiple kernels computed from a feature. We will use a super-script in this situation, viz., $\mathbf{x}_i^k$ is the feature vector describing source $x_i$ that is used to compute the $k$th kernel matrix $K_k$; thus, we use $\phi_i^k$ to denote $\phi(\mathbf{x}_i^k)$.

Let $K_1 = [\kappa_1(\mathbf{x}_i, \mathbf{x}_j)]$ and $K_2 = [\kappa_2(\mathbf{x}_i, \mathbf{x}_j)]$, $i, j = [n]$. The set of Mercer kernels is closed under the following (non-exhaustive) set of operations for $i, j = [n]$ [19]:

P7. (Sum)
$$\mathcal{K}_{ij} = (K_1)_{ij} + (K_2)_{ij} \tag{6}$$

P8. (Scalar Product)
$$\mathcal{K}_{ij} = c(K_1)_{ij}, \forall c > 0 \tag{7}$$

P9. (Addition by Constant)
$$\mathcal{K}_{ij} = (K_1)_{ij} + c, \forall c > 0 \tag{8}$$

P10. (Product)
$$\mathcal{K}_{ij} = (K_1)_{ij}(K_2)_{ij} \tag{9}$$

Numerous kernels have been proposed, with the most popular being the polynomial kernel and *radial basis function* (RBF) kernels defined as

$$\kappa_P(\mathbf{x}_i, \mathbf{x}_j) = (< \mathbf{x}_i, \mathbf{x}_j > + \alpha)^q, q \in \mathcal{N}, \tag{10}$$

$$\kappa_{RBF}(\mathbf{x}_i, \mathbf{x}_j) = exp(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\sigma^2}), \sigma > 0. \tag{11}$$

Kernel design takes a number of forms. In most scenarios, a single kernel is employed. Its parameters are generally specified manually or the selections of the kernel and its parameters are learned from data. However, the most widespread practice,

[1] Later, we will extend this idea to the scenario where one can have multiple feature vectors (really, multiple kernels) per source. This can manifest in one (or a combination) of two ways: i) different kernels can be computed from one feature type, or ii) a kernel can be computed from each of multiple heterogeneous feature types.

to date, is to experimentally choose from a finite number of kernels and associated kernel parameters to pick a winner.

In recent years, MK has been proposed to help address the problem of finding a "best" kernel for a data set, and more recently, for focusing on the fusion and unique transformation of different sources (e.g., sensors and/or features). Specifically, the field of MK *learning* (MKL) has emerged to find such kernel combinations automatically. Research has shown that the construction of a kernel from a number of base kernels allows for a more flexible encoding of domain knowledge from different sources or cues. In the following subsection, important MK and MKL work are reviewed in three categories, learning, aggregation, and training.

### A. Learning Method

A number of learning methods have been proposed for MKL. Generally, the methods fall into the categories of fixed rules, heuristics, and optimization [20].

*Fixed rules* are functions with no parameters and no training. Aavlidis et al. [21] proposed a simple summation of kernels for gene functional classification. Ben-Hur and Noble [22] used both summation and multiplication to compose *pairwise kernels* and *genomic kernels* to improve the classication performance of protein-protein interaction prediction. This category is the simplest and is more about the application or the specific kernels within MK in general.

*Heuristic* approaches use a parametrized aggregation function and they find the weight generally by a measure obtained from each kernel function separately. Moguerza et al. [23] proposed heuristics to estimate the aggregation weights based on conditional class probabilities and de Diego et al. [24] used this method to fuse information from several feature representations for face recognition. In [25], Qiu and Lane proposed two simple heuristics to select the kernel weights for regression problems . While more sophisticated than fixed rules, heuristic approaches are also not flexible enough to address a wide range of MKL applications.

*Optimization* approaches use a parametrized aggregation function, as in the case of heuristic approaches, but they learn the function weights by solving an optimization problem. This is the most flexible and powerful theory of the three categories discussed. The optimization can be integrated into a kernel-based learner, such as the SVM. In [26], Cortes et al. proposed the centered-kernel alignment with corresponding optimization task (which has an analytical solution). Tanabe et al. [27] optimized the kernel weights for a convex combination of kernels by minimizing a feature space-based kernel matrix evaluation measure. In [28], Lewis et al. used a latent variable generative model using the maximum entropy discrimination to learn data-dependent kernel combination weights. Kloft et al. [29] and Xu et al. [30] learned convex combinations of kernels using the $\ell_1$-norm for regularizing the kernel weights.

### B. Aggregation Method

In general, MK aggregation can be categorized as either linear or non-linear, with most methods being of type linear. Here, we review a few relevant works and formulations.

Linear aggregation is the most popular. It is based on a (potentially weighted) summation of base kernels,

$$\mathcal{K} = \sum_{k=1}^{m} \omega_k K_k \tag{12}$$

where $\omega$ are weights, and $K_k = [\kappa_k(\mathbf{x}_i^k, \mathbf{x}_j^k)]$ is the kernel matrix produced by the $k$th feature extracted from the sources $X$. Note, many simply search for a single aggregate kernel to transform $x_i$ and $x_j$, thus one kernel function $\kappa$ is applied to all features extracted from $x_i$ and $x_j$. However, better performance may be obtained by placing each feature in its own space, viz., the $k$th feature vector gets its own kernel function $\kappa_k$. Approaches to MK aggregation differ in the way that restrictions are placed on the weights $\omega$. The most common categories include the linear sum ($\omega_k \in \mathcal{R}$) [31], conic sum ($\omega_k \in \mathcal{R}^+$ ) [32], and convex sum ($\omega_k \in \mathcal{R}^+$ and $\sum_{k=1}^{m} \omega_k = 1$) [33]. Compared to linear sum, conic and convex sum are appealing because they give rise to weight (thus kernel and/or source) importance interpretation.

Non-linear aggregation methods combine kernels in non-linear ways, e.g., via multiplication, power and exponentiation. In [34], Cortes et al. proposed polynomial combinations of kernels in regression. This non-linear strategy leads to improved performance (over linear kernel aggregation), in part because the solution space had merely been enlarged. Additional non-linear methods can be found in [35, 36]. However, existing non-linear combinations usually result in non-convex optimization problems, leading to higher complexity and computational cost. Moreover, existing solutions for non-linear combinations of kernels have been difficult to interpret. One problem with the existing set of non-linear aggregation techniques is they are relatively unsophisticated in comparison to the flexibility of the FI. Our FI approach is both extremely flexible and it has the nice property that it can be realized in terms of the popular linear convex sum form. As already discussed, in [12], Keller et al. showed that for different selection of FM, the CI can produce many different order statistics, e.g., the max, min, median, etc., and linguistic terms have been defined as well, e.g., soft min, soft max, etc. The FI is unique in that it offers non-linear sophisticated potential but yet it can be embedded in the context of existing optimal linear convex sum form.

### C. Training Method

In general, MKL can be organized into either one-step or two-step techniques. One-step methods calculate both the aggregation function parameters and the parameters of the combined base learner (e.g., SVM) in a single pass. Most one-step methods use a sequential approach or a simultaneous approach. In sequential approach, the aggregation weights are determined first, then a kernel-based learner is trained using the combined kernel. In simultaneous approach, both set of parameters are learned at the same time.

Two-step methods use an iterative approach (alternating optimization). In each iteration, the aggregation weights are updated first with fixed base learner parameters. Next, the base kernel parameters are updated using fixed aggregation weights. These steps are repeated until convergence. In the next section, we discuss the mathematics of using the FI for MK. Specifically, we provide a few proofs that show some FIs can be used for MK, while others cannot.

## IV. Multiple kernel aggregation using the Fuzzy Integral

In this section, we focus on the applicability of two most popular formulations of the FI. In Equation 3 and 4, the first step is the re-permutation of the sources (in this case, the sources will be kernels). First, we discuss the validity of the FI if sorting is done per cell in K. Next, we discuss the impact

of another sorting based on a quality value with respect to the base-learner. We show that it does indeed matter which sorting method and which formulation of the FI one uses. It matters not only in terms of yielding valid MKs (viz., Mercer kernels), but also in terms of the resulting mathematical form and its connection to existing categories of linear sum, conic and convex sum.

**Proposition 1.** The maximum operator [37],

$$\mathcal{K}_{ij} = \max\{(K_1)_{ij}, (K_2)_{ij}\}, \quad i,j = [n], \tag{13}$$

is not guaranteed to produce a $\mathcal{K}$ that is a Mercer kernel.

*Proof:* This proof is centered on the use of max (and similarly min) to combine Mercer kernels. Max is easily rejected as a valid way to aggregate kernels using a simple proof by contradiction. Consider the following example for $n = 3$,

$$K_1 = \begin{bmatrix} 2 & -1 & -2 \\ -1 & 2 & 3 \\ -2 & 3 & 8 \end{bmatrix}, \quad \lambda_1 = \begin{bmatrix} 0.72 \\ 1.4 \\ 9.8 \end{bmatrix},$$

and

$$K_2 = \begin{bmatrix} 7 & 4 & -2 \\ 4 & 3 & -1 \\ -2 & -1 & 1 \end{bmatrix}, \quad \lambda_2 = \begin{bmatrix} 0.29 \\ 0.68 \\ 10.02 \end{bmatrix},$$

where $\lambda_1$ and $\lambda_2$ are the eigenvalues for $K_1$ and $K_2$ respectively. Hence,

$$\mathcal{K} = \begin{bmatrix} 7 & 4 & -2 \\ 4 & 3 & 3 \\ -2 & 3 & 8 \end{bmatrix}, \quad \lambda_{\mathcal{K}} = \begin{bmatrix} -0.92 \\ 9.34 \\ 9.57 \end{bmatrix}.$$

The presence of negative eigenvalues in this counter example shows that max does not always produce a $\mathcal{K}$ which a Mercer kernel (similar proof holds for the min operator). ■

### A. Element-Wise Sorting of Kernel Matrix Aggregation

In the FI, one sorts according to the confidences (aka the $h$ values). However, the FI is used in this article to fuse kernels, which are matrices not scalars. So, the question is how to perform the sorting. The first discussion is on sorting per kernel matrix element. This means that each $(x_i, x_j)$ combination will be sorted differently. Mathematically, there is no violation, however one needs to verify that this selection scheme will produce valid MKs. We show that neither the SI nor CI can produce a valid MK based on a sorting per cell.

**Proposition 2.** Given $m$ base Mercer kernels, $\{\kappa_1, \ldots, \kappa_m\}$, and a FM, $g$, the discrete SI kernel aggregation,

$$\mathcal{K}_{ij} = \bigvee_{k=1}^{m} \left\{(K_{(k)_{ij}})_{ij} \wedge G_{(k)_{ij}}\right\}, \quad i,j = [n], \tag{14}$$

is not guaranteed to produce a $\mathcal{K}$ that is a Mercer kernel. Note that the sort index $(k)_{ij}$ is indexed to indicate a dependence on the element-wise sort, viz., $(K_{(1)_{ij}})_{ij} \geq (K_{(2)_{ij}})_{ij} \geq \ldots \geq (K_{(m)_{ij}})_{ij}$.

*Proof:* Consider a FM in which each $g$ is 1 and a set of RBF kernels. It is easily shown that $\kappa_{RBF}(\mathbf{x}_1, \mathbf{x}_2) \leq 1$, for any $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{R}^d$. Because $G_{(k)_{ij}} = 1$, $k = [m]$, the inner min operation $(K_{(k)_{ij}})_{ij} \wedge G_{(k)_{ij}} = (K_{(k)_{ij}})_{ij}$. Hence, (14) becomes equivalent to the max operation at (13), which was proved in Proposition 1 to not always produce a Mercer kernel. ■

**Proposition 3.** Given $m$ base Mercer kernels, $\{\kappa_1, \ldots, \kappa_m\}$, and a FM, $g$, the difference-in-measure CI,

$$\mathcal{K}_{ij} = \sum_{k=1}^{m} (\omega_{ij})_k (K_{(k)_{ij}})_{ij}, \quad i,j = [n], \tag{15}$$

where $(\omega_{ij})_k = (G_{(k)_{ij}} - G_{(k-1)_{ij}})$ and $G_{(0)} = 0$, is not guaranteed to produce a $\mathcal{K}$ that is a Mercer kernel.

*Proof:* A FM can be constructed such that $G_{(k)_{ij}} = 1$ and $k = [m]$. This causes (15) to be equivalent to the max operation at (13). Hence, this proof follows directly the proof of Propositions 1 and 2. ■

### B. Matrix-Wise Sorting of Kernel Matrix Aggregation

In the previous section, we showed that sorting cannot be done on a per-element basis. Semantically, per-element also does make much sense and mathematically—viz., in terms of production of a Mercer kernel—it is not valid. Here, we propose an alternative solution based on sorting at the matrix level. Assume that each kernel matrix $K_k$ has a numeric "quality." This could be computed, for example, by computing the classification accuracy of a base-learner that uses kernel $K_k$ or, as you will see later, by a learning algorithm such as GA. Let $\nu_k \in [0, 1]$ be the quality of the $k$th kernel. These qualities can be sorted, $\nu_{(1)} \geq \nu_{(2)} \geq \ldots \geq \nu_{(m)}$. We now investigate the SI and CI with regards to this matrix-level quality measure (this is the FM) and sorting method.

**Proposition 4.** Given $m$ base Mercer kernels, $\{\kappa_1, \ldots, \kappa_m\}$, FM $g$ and a sorting $\nu_{(1)} \geq \nu_{(2)} \geq \ldots \geq \nu_{(m)}$, the SI form,

$$\mathcal{K}_{ij} = \bigvee_{k=1}^{m} \left\{(K_{(k)})_{ij} \wedge G_{(k)}\right\}, \quad i,j = [n], \tag{16}$$

is not guaranteed to produce a $\mathcal{K}$ that is a Mercer kernel.

*Proof:* This proof is nearly identical to the proof of Proposition 2. ■

**Proposition 5.** Given $m$ base Mercer kernels, $\{\kappa_1, \ldots, \kappa_m\}$, FM $g$ and a sorting $\nu_{(1)} \geq \nu_{(2)} \geq \ldots \geq \nu_{(m)}$, the difference-in-measure CI form,

$$\mathcal{K}_{ij} = \sum_{k=1}^{m} \omega_k (K_{(k)})_{ij}, \quad i,j = [n], \tag{17}$$

produces a $\mathcal{K}$ that is a Mercer kernel.

*Proof:* By properties P7 and P8. ■

**Remark 1.** While Propositions 1-5 outline scenarios where max (min), SI, and CI do and do not produce a $\mathcal{K}$ that is Mercer kernel, the more interesting question becomes, when do max (min), element-wise SI and CI, and base-learner sorted SI produce a valid Mercer kernel $\mathcal{K}$? We aim to tackle this question in future work on this topic.

Proposition 5 leads to our proposed GA solution to aggregating kernel matrices with the CI.

TABLE II. UCI DATA SETS INVESTIGATED

| Dataset | Breast | Ionosphere | Sonar |
|---|---|---|---|
| Number of instances | 286 | 351 | 208 |
| Number of features | 9 | 34 | 60 |
| Number of classes | 2 | 2 | 2 |

### C. Genetic Algorithm for FI Aggregation of MKs

In our proposed FIGA algorithm, a GA [38] is used to learn the FM, which describes the quality of combinations of kernel matrices. Specifically, the densities are learned and the Sugeno $\lambda$-FM is used to derive the rest of the measure. A GA is a biologically motivated stochastic search procedure for an optimal solution to a given task. Herein, the task is SVM-based classification using MKL. Our chromosome is of length $m$ (number of kernels), where gene $E_k$ is the value of $g^k$ (the $k$th density). Our fitness function is the accuracy of the SVM, $[0, 1]$. A population of $50$ chromosomes is used and we perform $150$ iterations. Proportional selection (roulette wheel) is used and we also select the most fit individual at each iteration and pass it along to the next iteration as well. One-point crossover (rate of $60\%$) and mutation (rate of $5\%$) are used. Note, more sophisiticated GA approaches, e.g., that proposed in [14], could be used to learn each point in the lattice versus just the densities. Other possibilities include quadratic programming approaches [39].

## V. EXPERIMENTS AND RESULTS

In this section, we demonstrate our proposed FIGA concept on the benchmark UCI data sets shown in Table II.

### A. Datasets

In order to evaluate classification performance, three UCI datasets are used (see Table II). In each experiment, $80\%$ of the data is selected at random for training and the rest is testing. We apply $m = 10$ base kernels to the entire feature vector:

- 4 RBF kernels with different $\frac{1}{\sigma^2}$ values: 1 / number of data points, 1 / number of features, the median distance of the data set, and $0.01$;
- Dot product kernel (standard Euclidean distance);
- Five polynomial kernels of degrees $2, 3, 4, 5, 6$.

### B. Comparison of FIGA to MKLGL

In this sub-section we investigate the quantitative performance of FIGA to the state-of-the-art machine learning technique MKL by *group lasso* (MKLGL) [30]. FIGA was run five times for each dataset. The experimental results are reported in Table III.

Table III tells the following story. First, FIGA outperforms MKLGL in each data set. That is, FIGA has a higher average accuracy and a lower error band. Take the Sonar dataset for example. The average accuracy of FIGA is 8% higher than MKLGL and the error band is decreased by nearly 3. The improvements on the other two data sets is not as significant as Sonar; however, the performance differences are consistent.

In addition, we would like to note the following. In MKLGL, the authors used 117, 442 and 793 kernels respectively. In FIGA, we used just 10! In MKLGL, their intent was to use 13 kernels on the full feature vector and 13 additional kernels per single feature vector element. Their intent is to use MKLGL to, in effect, perform feature selection. Regardless, their technique is more complex than ours and ultimately has inferior performance.

TABLE III. COMPARISON OF FIGA TO MKLGL

| | Method | Breast | Ionosphere | Sonar |
|---|---|---|---|---|
| Accuracy | FIGA | $98.2 \pm 0.9$ | $94.0 \pm 2.16$ | $90.15 \pm 4.13$ |
| $[0, 100]\%$ | MKLGL | $96.6 \pm 1.2$ | $92.0 \pm 2.9$ | $82.0 \pm 6.8$ |
| Number | FIGA | 10 | 10 | 10 |
| of kernels | MKLGL | 117 | 442 | 793 |

## VI. CONCLUSION AND FUTURE WORK

The results of our experiments clearly show that using fuzzy integrals to aggregate multiple kernels shows added benefit to existing MKL algorithms. The proposed FIGA algorithm produced superior results when compared to MKL group lasso [30] on three well-known datasets. Interesting questions arise when examining these results. The FIGA algorithm uses a global performance measure to generate candidate densities (qualities) of the individual kernels. We believe that these results could be expanded on by using a local performance measure which sets the individual densities according the classification accuracy of base learners (SVMs) trained on each individual kernel. An anticipated drawback to local performance measures is that kernels that show good generalized classification results will dominate the overall solution; hence, kernels that classify only a few "hard" data points (to the benefit of global performance) may be downplayed in the overall aggregation. Hybrid approaches that use both global and local performance measures may help improve the overall diversity of the kernel ensemble while maintaining good overall performance.

In the future, we propose to further explore (and exploit) the connection of fuzzy integral-based MKL to ensemble learning. We will also investigate a more efficient and direct approach to learning the fuzzy measure in a data-driven fashion.


## ACKNOWLEDGMENT

This research was supported in part by a National Institute of Justice grant (2011-DN-BX-K838).


## REFERENCES


[1] Schlkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. The MIT Press, Cambridge, MA, 2002.

[2] M. Filippone, F. Camastra, F. Masulli, and S. Rovetta, "A survey of kernel and spectral methods for clustering," *Pattern Recogn.*, vol. 41, no. 1, pp. 176–190, Jan. 2008.

[3] T. C. Havens, J. C. Bezdek, C. Leckie, L. O. Hall, and M. Palaniswami, "Fuzzy c-means algorithms for very large data," *Fuzzy Systems, IEEE Transactions on*, vol. 20, no. 6, pp. 1130 –1146, dec. 2012.

[4] W. An-na, Z. Yue, H. Yun-tao, and L. Yun-lu, "A novel construction of svm compound kernel function," in *Logistics Systems and Intelligent Management, 2010 International Conference on*, vol. 3, jan. 2010, pp. 1462 –1465.

[5] M. Lu, C. Chen, J. Huo, and X. Wang, "Optimization of combined kernel function for svm based on large margin learning theory," in *Systems, Man and Cybernetics. IEEE International Conference on*, oct. 2008, pp. 353 –358.

[6] B. Chen, L. Duan, and J. Hu, "Composite kernel based svm for hierarchical multi-label gene function classification," in *Neural Networks (IJCNN), The 2012 International Joint Conference on*, june 2012, pp. 1 –6.

[7] D. Anderson, O. Sjahputera, K. Stone, and J. Keller, "Causal cueing system for above ground anomaly detection of explosive hazards using support vector machine localized by k-nearest neighbor," in *Computational Intelligence for Security and Defence Applications (CISDA), 2012 IEEE Symposium on*, july 2012, pp. 1 –8.

[8] K. Stone, J. M. Keller, D. T. Anderson, and D. B. Barclay, "An automatic detection system for buried explosive hazards in fl-lwir and fl-gpr data," pp. 83 571E–83 571E–15, 2012.

[9] T. C. Havens, K. Stone, D. T. Anderson, J. M. Keller, K. C. Ho, T. T. Ton, D. C. Wong, and M. Soumekh, "Multiple kernel learning for explosive hazard detection in forward-looking ground-penetrating radar," pp. 83 571D–83 571D–15, 2012.

[10] D. Anderson, J. Farrell, K. Stone, J. Keller, and C. Spain, "Fusion of anomaly algorithm decision maps and spectrum features for detecting buried explosive hazards in forward looking infrared imagery," in *Applied Imagery Pattern Recognition Workshop (AIPR), 2011 IEEE*, oct. 2011, pp. 1 –8.

[11] M. Grabisch, T. Murofushi, and M. Sugeno, *Fuzzy measures and integrals: theory and applications*, ser. Studies in fuzziness and soft computing. Physica-Verlag, 2000.

[12] H. Tahani and J. Keller, "Information fusion in computer vision using the fuzzy integral," *IEEE Transactions System Man Cybernetics*, vol. 20, pp. 733–741, 1990.

[13] M. Sugeno, "Theory of fuzzy integrals and its applications," *Ph.D. thesis*, vol. Tokyo Institute of Technology, 1974.

[14] D. T. Anderson, J. M. Keller, and T. C. Havens, "Learning fuzzy-valued fuzzy measures for the fuzzy-valued sugeno fuzzy integral," in *Proceedings of the Computational intelligence for knowledge-based systems design, and 13th international conference on Information processing and management of uncertainty*, ser. IPMU'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 502–511.

[15] C. Guo, D. Zhang, and C. Wu, "Fuzzy-valued fuzzy measures and generalized fuzzy integrals," *Fuzzy Sets Syst.*, vol. 97, no. 2, pp. 255–260, Jul. 1998.

[16] M. Grabisch, H. Nguyen, and E. Walker, *Fundamentals of uncertainty calculi, with applications to fuzzy inference*. Kluwer Academic, Dordrecht, 1995.

[17] D. T. Anderson, T. C. Havens, C. Wagner, J. M. Keller, M. F. Anderson, and D. J. Wescott, "Sugeno fuzzy integral generalizations for sub-normal fuzzy set-valued inputs," in *FUZZ-IEEE*, 2012, pp. 1–8.

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

[19] R. Herbrich, *Learning Kernel Classifiers: Theory and Algorithms*. Cambridge, MA, USA: MIT Press, 2001.

[20] M. Gönen and E. Alpaydin, "Multiple kernel learning algorithms," *Journal of Machine Learning Research*, vol. 12, pp. 2211–2268, 2011.

[21] P. Pavlidis, J. Cai, J. Weston, and W. S. Noble, "Learning gene functional classifications from multiple data types," *JOURNAL OF COMPUTATIONAL BIOLOGY*, vol. 9, pp. 401–411, 2002.

[22] A. Ben-Hur and W. S. Noble, "Kernel methods for predicting protein–protein interactions," *Bioinformatics*, vol. 21, no. 1, pp. 38–46, Jan. 2005.

[23] J. M. Moguerza, A. Muñoz, and I. M. de Diego, "Improving support vector classification via the combination of multiple sources of information," in *SSPR/SPR*, 2004, pp. 592–600.

[24] I. M. Diego, A. Muñoz, and J. M. Moguerza, "Methods for the combination of kernel matrices within a support vector framework," *Mach. Learn.*, vol. 78, no. 1-2, pp. 137–174, Jan. 2010.

[25] S. Qiu and T. Lane, "A framework for multiple kernel support vector regression and its applications to sirna efficacy prediction," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 6, pp. 190–199, 2009.

[26] C. Cortes, M. Mohri, and A. Rostamizadeh, "Two-stage learning kernel algorithms," in *ICML*, 2010, pp. 239–246.

[27] H. Tanabe, T. B. Ho, C. H. Nguyen, and S. Kawasaki, "Simple but effective methods for combining kernels in computational biology." in *RIVF*. IEEE, 2008, pp. 71–78.

[28] D. P. Lewis, T. Jebara, and W. S. Noble, "Nonstationary kernel combination," in *In 23rd International Conference on Machine Learning (ICML*, 2006, pp. 553–560.

[29] M. Kloft, U. Brefeld, S. Sonnenburg, and A. Zien, "Non-sparse regularization and efficient training with multiple kernels," University of California, Berkeley, Tech. Rep. CoRR abs/1003.0079 and Technical Report UCB/EECS-2010-21, 2010.

[30] Z. Xu, R. Jin, H. Yang, I. King, and M. R. Lyu, "Simple and efficient multiple kernel learning by group lasso," in *ICML*, 2010, pp. 1175–1182.

[31] C. Igel, T. Glasmachers, B. Mersch, N. Pfeifer, and P. Meinicke, "Gradient-based optimization of kernel-target alignment for sequence kernels applied to bacterial gene start detection," *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, vol. 4, no. 2, pp. 216 –226, april-june 2007.

[32] D. Conforti and R. Guido, "Kernel based support vector machine via semidefinite programming: Application to medical diagnosis," *Computers & OR*, vol. 37, no. 8, pp. 1389–1394, 2010.

[33] C. Cortes, M. Mohri, and A. Rostamizadeh, "L2 regularization for learning kernels," *CoRR*, vol. abs/1205.2653, 2012.

[34] ——, "Learning non-linear combinations of kernels," in *Advances in Neural Information Processing Systems 22*, Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, Eds., 2009, pp. 396–404.

[35] W.-J. Lee, S. Verzakov, and R. P. W. Duin, "Kernel combination versus classifier combination," in *Proceedings of the 7th international conference on Multiple classifier systems*, ser. MCS'07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 22–31.

[36] M. Varma and B. R. Babu, "More generality in efficient multiple kernel learning," in *ICML*, 2009, p. 134.

[37] S. Boughorbel, J.-P. Tarel, and F. Fleuret, "Non-mercer kernels for svm object recognition," in *In British Machine Vision Conference (BMVC)*, 2004, pp. 137–146.

[38] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, 1st ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1989.

[39] A. Abdallah, H. Frigui, and P. Gader, "Adaptive local fusion with fuzzy integrals," *Fuzzy Systems, IEEE Transactions on*, vol. 20, no. 5, pp. 849 –864, oct. 2012.