Efficient Binary Fuzzy Measure Representation and Choquet Integral Learning

Muhammad Aminul Islam[1], Derek T. Anderson[2], Xiaoxiao Du[2], Timothy C. Havens[3], and Christian Wagner[4]

[1] Mississippi State University, Mississippi State, MS, USA,
mi160@msstate.edu
[2] University of Missouri, Columbia, MO, USA
andersondt@missouri.edu, xdy74@mail.missouri.edu
[3] Michigan Technological University, Houghton, MI, USA
thavens@mtu.edu
[4] University of Nottingham, Nottingham, UK
christian.wagner@nottingham.ac.uk

Abstract. The *Choquet integral* (ChI), a parametric function for information aggregation, is parameterized by the *fuzzy measure* (FM), which has 2^N real-valued variables for N inputs. However, the ChI incurs huge storage and computational burden due to its exponential complexity relative to N and, as a result, its calculation, storage, and learning becomes intractable for even modest sizes (e.g., $N = 15$). Inspired by empirical observations in multi-sensor fusion and the more general need to mitigate the storage, computational, and learning limitations, we previously explored the *binary ChI* (BChI) relative to the *binary fuzzy measure* (BFM). The BChI is a natural fit for many applications and can be used to approximate others. Previously, we investigated different properties of the BChI and we provided an initial representation. In this article, we propose a new efficient learning algorithm for the BChI, called EBChI, by utilizing the BFM properties that add at most one variable per training instance. Furthermore, we provide an *efficient representation of the BFM* (EBFM) scheme that further reduces the number of variables required for storage and computation, thus enabling the use of the BChI for "big N". Finally, we conduct experiments on synthetic data that demonstrate the efficiency of our proposed techniques.

Keywords: binary Choquet integral, binary fuzzy measure

1 Introduction

Data/information fusion can be described as the process of combining of multiple inputs to provide a more accurate, concise, and/or reliable result than what a single source can achieve on its own. Driven by the need for better results, countless applications in many fields, such as computer vision and remote sensing, have long been applying fusion at different "levels" (signal, feature, decision etc.). Furthermore, the daily advancement in engineering technologies like smart

cars, which operate in complex and dynamic environments using multiple sensors, are raising both the demand for and complexity of fusion.

While there is a multitude of *fuzzy integral* (FI) variants for fusion, we focus in this paper on the *Choquet Integral* (ChI), a well-known, demonstrated, and flexible aggregation function. The ChI has been used in numerous applications (mostly focused on decision level fusion), e.g., humanitarian demining [1], computer vision [2], pattern recognition [3–7], multi-criteria decision making [8, 9], control theory [10], and multiple kernel learning [1, 11–14]. The ChI is a nonlinear aggregation function parameterized by a *fuzzy measure* (FM), a normal and monotone capacity. The FM is defined on the power set of the sources, i.e., on the sets of all possible combinations of sources, and therefore has 2^N variables for N sources. With the flexibility of choosing values for these $2^N - 2$ parameters (excluding the null set and X, which have fixed values by definition) in the FM, the ChI covers a wide range of aggregation operators. However, this advantage comes at a price: the requirement to specify (by human) or learn (from data) the FM. This means that the complexity of a learning problem, both in terms of storage and computation, is on an exponential order in respect to the number of sources. Therefore, a learning problem with the full set of variables becomes intractable at a relatively small N. Different approaches exist to learn the ChI from data, e.g., *quadratic programming* (QP) [15], gradient descent [16], penalty/reward [17], Gibbs sampler [18], linear programming [19], and efficient optimization with only data-supported variables [20].

In [21], we explored the *binary fuzzy measure* (BFM). The need to investigate the BFM was driven by experimental findings in binary decision making for machine learning [22]. Specifically, multiple instance learning was used to acquire the ChI for signal processing and the learned FM had values approximately in $\{0, 1\}$ versus $[0, 1]$. This suggests that the underlying FM in some applications can be binary, which motivates the use of a BFM directly rather than the real-valued FM due to its simplicity and efficient computation. Thus, many problems are a natural fit for the BFM and others are likely approximatable.

The BFM also has nice properties and computational advantages over the FM. In [21], we showed that the ChI relative to the BFM is equivalent to the Sugeno integral. We also showed that only one variable is effectively used for the ChI computation of an observation in comparison to N variables for the real valued FM. Moreover, only one-valued variables need be stored since zero-valued variables can be discarded. These features make the ChI computation and its storage (the BFM) less expensive compared to the real valued FM/ChI.

Herein, we first put forth an efficient data-driven learning method for the BFM and subsequently the BChI, which we refer to as *efficient BChI* (EBChI). Based on the fact that only one variable contributes to the BChI computation of an instance, we can explain this for variable selection during learning. Thus, each training instance adds at most one variable and the learning problem consequently becomes scalable to the problem size. That is, the number of variables to be optimized is no longer exponential of N, but rather linear to the number of instances (in the worst case). This not only lessens the computation burden,

but it also provides a more robust and generalized solution since the number of unknowns (variables) are always fewer then the number of equations (training instances). In contrast, the learning problem with the entire set of FM variables is prone to overfitting as the number of training samples now becomes much smaller than the number of variables ($2^N - 2$) [23].

Next, we provide a representation scheme for the BFM with the minimum set of variables, which we call the *efficient BFM* (EBFM). The BFM variables can be partitioned into two groups, one-valued variables and zero-valued variables. Among the one-valued variables, some of them can deduce their values from others using the FM's monotonicity property (which we call dependent variables) while others cannot (which we call independent variables). The dependent variables can be eliminated without any loss of information and the BFM can be represented with only the independent variables, of which there can be at most $\binom{N}{N/2}$ variables. The full set of FM variables can be retrieved from these independent variables and vice-versa. Therefore, the independent variables constitute the minimal BFM or EBFM.

In Section 2 we give the preliminaries of the FM, BFM, ChI, and BChI. Section 3 describes the efficient data-driven BFM learning followed by the representation in Section 4. In Section 5, we conduct experiments on synthetic data to demonstrate the performance of our proposed learning method.

2 Fuzzy Measure and the Choquet Integral

Let $X = \{x_1, x_2, ..., x_N\}$ be a discrete set of N sources. A FM is a monotonic function defined on the power set of X, 2^X, as $\mu : 2^X \rightarrow \Re^+$ that satisfies: (i) (boundary condition:) $\mu(\emptyset) = 0$, and $\mu(X) > 0$ and (ii) (monotonicity:) if $A, B \subseteq X, A \subseteq B$, $\mu(A) \leq \mu(B)$. Often an additional constraint is imposed to limit the upper bound to 1, i.e., $\mu(X) = 1$. Let $h(x_i)$ be the data/information from the ith source. The discrete ChI (finite X) is

$$\int_C \mathbf{h} \circ \mu = C_\mu(\mathbf{h}) = \sum_{i=1}^{N} h(x_{\pi(i)}) \left[\mu(S_{\pi(i)}) - \mu(S_{\pi(i-1)})\right], \tag{1}$$

where π is a permutation of X, such that $h(x_{\pi(1)}) \geq h(x_{\pi(2)}) \geq \ldots \geq h(x_{\pi(N)})$, $S_{\pi(i)} = \{x_{\pi(1)}, \ldots, x_{\pi(i)}\}$, and $\mu(S_0) = 0$. Equation (1) is often referred to as the difference-in-measure form since the integral is represented as the sum of difference-in-measure weighted by the input-values. The ChI can equivalently be written in difference-in-inputs form as

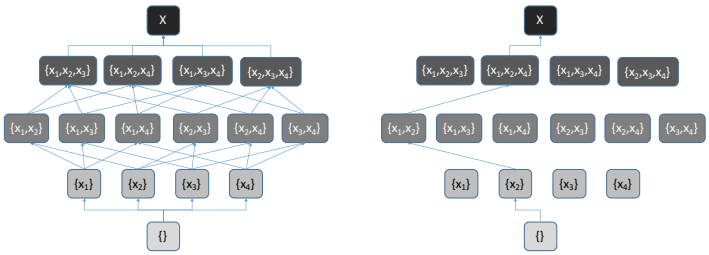
$$\int_C \mathbf{h} \circ \mu = C_\mu(\mathbf{h}) = \sum_{i=1}^{N} [h(x_{\pi(i)}) - h(x_{\pi(i+1)})]\mu(S_{\pi(i)}), \tag{2}$$

where $h(x_{\pi(N+1)}) = 0$. The latter weighted-measure form at (2) is suitable for the FM learning problem herein, where μ is unknown. Equation 2 can also be written in matrix form to facilitate optimization as

$$C_\mu(\mathbf{h}) = \mathbf{c}^T \mathbf{u}_B, \tag{3}$$

where $\mathbf{u}_B$ is the vector of all variables except $\mu(\emptyset)$ and has a length of $2^N - 1$, and $\mathbf{c}$ holds the coefficients of $\mathbf{u}_B$ for observation $\mathbf{h}$.

The FM can be visualized with respect to its underlying Hasse diagram (shown in Figure 1). Each instance yields a sort, π, which produces a walk up the lattice. The walk starts with $\mu(\emptyset)$ followed by N other variables, each of different size cardinality. For example, an observation $\mathbf{h}$ with $h(\{x_2\}) \geq h(\{x_1\}) \geq h(\{x_4\}) \geq h(\{x_3\})$ walks along the path shown in Figure 1(b) and the corresponding ChI has variables $\mu(\{x_2\}), \mu(\{x_1, x_2\}), \mu(\{x_1, x_2, x_4\})$, and $\mu(X)$.



(a) FM for N=4 (b) Walk for an example input

Fig. 1. (a) FM for four inputs. Arrows indicate monotonicity conditions on immediate subsets. (b) Path taken by observation $\mathbf{h}$ with $h(\{x_2\}) \geq h(\{x_1\}) \geq h(\{x_4\}) \geq h(\{x_3\})$. Only four variables $\mu(\{x_2\})$, $\mu(\{x_1, x_2\})$, $\mu(\{x_1, x_2, x_4\})$, and $\mu(X)$ are used for the ChI.

2.1 The Binary Fuzzy Measure

As already stated, a BFM, μ_B, is a special case of the real-valued FM, μ, that restricts μ_B to $\{0, 1\}$ instead of $[0, 1]$. Obviously, this drastically reduces the search space for an optimization problem. In article [21], we proved that the BChI and Sugeno Integral are equivalent. The BChI is simply the standard Choquet integral with respect to the BFM. Suppose the BFM values along the walk for $\mathbf{h}$ are given by $\mu_B(S_{\pi(i)}) = 0$ if $i < k$, else 1, where $\mu_B(S_{\pi(k)})$ is the first variable encountered along this path with value 1. Replacing the FM with this BFM in Eq. (1) and then expanding it, the BChI can be written as [21]

$$\int_C \mathbf{h} \circ \mu_B = C_{\mu_B}(\mathbf{h}) = \sum_{i=1}^{N} h(x_{\pi(i)}) \left[\mu_B(S_{\pi(i)}) - \mu_B(S_{\pi(i-1)})\right]$$

$$= \left(\sum_{i=1}^{k-1} h(x_{\pi(i)}) \left[\mu_B(S_{\pi(i)}) - \mu_B(S_{\pi(i-1)})\right]\right) + h(x_{\pi(k)}) \left[\mu_B(S_{\pi(k)}) - \mu_B(S_{\pi(k-1)})\right]$$

$$+ \left(\sum_{i=k+1}^{N} h(x_{\pi(i)}) \left[\mu_B(S_{\pi(i)}) - \mu_B(S_{\pi(i-1)})\right]\right) = h(x_{\pi(k)})\mu_B(S_{\pi(k)}), \tag{4}$$

since $\mu_B(S_{\pi(i)}) - \mu_B(S_{\pi(i-1)})$ is zero except for $i = k$ and $\mu_B(S_{\pi(k-1)}) = 0$. It is trivial to show with some mathematical manipulation that the BChI in difference-in-inputs form also can be written as

$$C_{\mu_B}(\mathbf{h}) = \sum_{i=1}^{N} [h(x_{\pi(i)}) - h(x_{\pi(i+1)})]\mu_B(S_{\pi(i)}) = h(x_{\pi(k)})\mu_B(S_{\pi(k)}). \quad (5)$$

According to (4) and (5), the BChI of an instance uses only one variable $\mu_B(S_{\pi(k)})$. This fact allows us to use significantly fewer variables than the standard ChI (which we will show in Section 3), thus enabling the learning of a BFM for larger number of inputs/sources problems, which otherwise would be intractable to solve on most personal computers.

3 BChI learning

Let $O = \{\mathbf{h}_j, y_j\},\ j = 1, 2, \ldots, M$, be a training data set with M instances. Here $\mathbf{h}_j$ represents the jth instance with data from N inputs and y_j is the associated label or ground-truth for $\mathbf{h}_j$. For example, $\mathbf{h}_j$ could be an image, $h_j(\{x_i\})$ could be the soft-max normalized decision of N different deep learners and $y_j = 0$ if $\mathbf{h}_j$ is not the category of interest, e.g., person, or $y_j = 1$ if it is.

The goal is to learn the BFM such that the aggregation results of the training instances optimize a criteria, which is usually specified by a function of error relative to a label, y_i, called an objective function. Common functions include the *sum of squared error* (SSE) [4,23,24] and the sum of absolute error. Without loss of generality, we focus on the SSE, which is widely used due to its continuity, differentiabiliy, and non-linearity relative to errors. The sum of squared error for training data, O, is $E(O, \mathbf{u}_B) = \sum_{j=1}^{M} (C_{\mu_B}(\mathbf{h}_j) - y_j)^2$, where $C_{\mu_B}(\mathbf{h}_j)$ is the BChI for instance $\mathbf{h}_j$ and y_j is associated label.

3.1 Learning with the full set of FM variables

Traditionally, FM learning is formulated with a full set of variables without extracting any knowledge from the training data to reduce the number of variables. In this case, the BChI for a training instance $\mathbf{h}_j$ is represented with a $2^N - 1$ dimensional vector $\mathbf{u}_B$, and, consequently, the SSE for training data, O, is

$$E(O, \mathbf{u}_B) = \sum_{j=1}^{M} e^j = \sum_{j=1}^{M} (\mathbf{c}_j^T \mathbf{u}_B - y_j)^2 = ||D\mathbf{u}_B - \mathbf{y}||_2^2,$$

where $D = [\mathbf{c}_1\ \mathbf{c}_2\ \ldots\ \mathbf{c}_M]^T$, $\mathbf{y} = [y_1\ y_2\ \ldots\ y_M]^T$, $||x||_2$ is norm-2 operation on x, and $\mathbf{u}_B$ is the vector of the $2^N - 1$ binary variables excluding the null set. Based on this, the SSE optimization problem is

$$\min_{\mathbf{u}_B} f(\mathbf{u}_B) = ||D\mathbf{u}_B - \mathbf{y}||^2,$$

$$
\begin{aligned}
&u_B(k) \leq u_B(l), \text{ if } u_B(k) = \mu_B(A), u_B(l) = \mu_B(B),\\
&\text{and } A \subset B, \forall k, l \text{ (monotonicity conditions)}\\
&u_B(p) = 1, \text{ if } u_B(p) = \mu_B(X) \text{ (normality conditions)}\\
&u_B(l) \in \{0, 1\}, \forall l \text{ (BFM value restriction)},
\end{aligned}
$$

which can be solved by any mixed-integer, integer, or binary quadratic programming library [25, 26]. It is obvious that this optimization problem does not scale well since the number of variables is exponential with respect to N regardless of the training sample size. Moreover, its complexity would be higher than the standard FM due to the use of integer-programming, which is in general costlier than the real-valued counterpart.

3.2 Efficient ChI learning

Herein we propose an efficient learning algorithm that selects variables for optimization from the training data as opposed to using the full set of variables in a standard method. The proposed method can drastically reduce the number of variables; however, the amount of savings depends on different factors such as training data volume, underlying FM, the context of the problem (noisy or no-noise), and the acceptable error in the objective function. As shown in the previous section, given a set of inputs, the BChI needs only one variable to determine the output. The converse is also true, i.e., given a set of inputs, the actual output (which has no noise or fluctuation) can help retrieve the variable responsible for the BChI. In this case, the output equals the kth sorted input, and the variable is associated with this kth input.

Suppose a training instance, $\mathbf{h}_j$, is not affected by noise, i.e., $e_j = 0$, then the output is, $y_j = h_j(x_{\pi_j(k)})$, and the variable corresponding to $h_j(x_{\pi_j(k)})$ is $\mu_B(S_{\pi_j(k)})$. Thus, just by inspecting each training instance, we can retrieve for a noise-free problem the BChI variables with their exact values without requiring any optimization. However, real world problems often–if not always–are affected by noise. When noise variance is small, we can choose to select one variable per instance by finding the sorted input k closest to the output label (which also results in minimum error, e_j), where k can be determined as

$$k = \arg\min_i ||y_j - h_j(x_{\pi_j(i)})||^2. \tag{6}$$

Since the noise can be random, training instances with the same walk in the lattice can be affected by varying magnitude of noise. Consequently, they can pick different k's and hence more than one variable for the same walk as opposed to a single variable for an ideal case. Suppose, the instances $\mathbf{h}_l, l \in \{t_1, \ldots, t_p\} \subseteq \{1, \ldots, M\}$ have the same permutation, π, for their sorting order and the set of variables picked by them using Eq. (6) are $\mu_B(S_{\pi(k)}), k = \{k_1, k_2, \ldots, k_Q\}$ with $S_{\pi(k_1)} \subset S_{\pi(k_2)} \cdots \subset S_{\pi(k_Q)}$. Then the BChI for $\mathbf{h}_l$ w.r.t. these variables is

$$C_{\mu_B}(\mathbf{h}_l) = \sum_{q=1}^{Q} [h_l(x_{\pi(k_q)}) - h_l(x_{\pi(k_{q+1})})] \mu_B(S_{\pi(k_q)}), \tag{7}$$

where $h_l(x_{\pi(k_{Q+1})}) = 0$. The number of selected variables is bounded in $[1, N]$.

For a highly noisy system, the user can choose to select multiple variables vs. one variable per training instance and use different criteria instead of squared of Euclidean distance as in Eq. (6), e.g., selecting variables associated with those inputs that fall within a certain threshold (or standard deviation) of the training label y_j. While increasing the number of variables will have no impact for a system with little noise with sufficient training samples, it can lower the SSE for a highly noisy system with limited data. However, there is an optimum balance between the complexity and the error as increasing variables even for high noise case has diminishing results.

Let the set of variables selected by all the training instances (using Eq. (6)) be represented in vector form as $\mathbf{v}_B$. Then the EBChI of $\mathbf{h}_j$ can be written in matrix form as $C_{\mu_B}(\mathbf{h}_j) = \mathbf{a}_j^T \mathbf{v}_B$, where $\mathbf{a}_j$ be the coefficient of $\mathbf{v}_B$ calculated according to Eq. (7). Based on this, the SSE minimization problem becomes

$$\min_{\mathbf{v}_B} f(\mathbf{v}_B) = ||W\mathbf{v}_B - \mathbf{y}||_2^2,$$

$$\begin{aligned}
&v_B(i) \leq v_B(j), \text{ if } v_B(i) = \mu(A), v_B(j) = \mu(B) \\
&\text{and } A \subset B, \forall i, j \in \{1, 2, \ldots, Q\} \text{ (monotonicity conditions)} \\
&v_B(j) = 1, \text{ if } v_B(j) = \mu_B(X) \text{ (normality conditions)} \\
&v_B(i) \in \{0, 1\}, \ \forall i \text{ (BFM value restriction)}
\end{aligned} \tag{8}$$

$$W = [\mathbf{a}_1 \ \mathbf{a}_2 \ \ldots \ \mathbf{a}_M]^T.$$

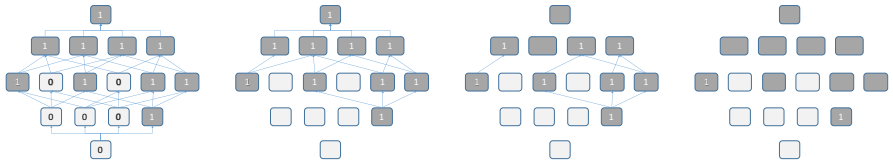
4 Efficient BFM data structure

In the last section, we provided an algorithm to efficiently learn a BFM that uses fewer variables. Anderson et al. introduced a simple approach to represent the BFM that can be applied here to further reduce the learned variables for efficient storage and representation. In that method, the variable elimination process considers only the values of the variables and does not take into account the monotonicity property of the FM, which can greatly enhance the representation technique. By taking into consideration both values and properties, herein we propose a new way to efficiently represent the full-fledged BFM and then we provide an upper bound on the minimum number of variables required.

4.1 Representation

Since the variables of a BFM are binary valued, the variables can take either zero or one values. The zero-valued variables do not contribute to the BChI, so they can be discarded. Thus, only one-valued variables can be considered as candidates for representation. Due to the monotonicity property, if a variable

$\mu_B(A)$ is one-valued, then all variables that are supersets of A are also one-valued. As such, the one-valued variables can be divided into two parts: (i) independent variables whose values cannot be derived from another one-valued variable using monotonicity condition and (ii) dependent variables whose values can be retrieved using the independent variables and monotonicity condition, and therefore, can be discarded. Consequently, only the one-valued independent variables are necessary to represent a BFM, which we refer to as EBFM. Figure 2 illustrates the EBFM representation technique for an example with $N = 4$.



(a) Full set of FM variables (b) Only one-valued variables (c) Training variables from an arbitrary data set (d) Independent variables

Fig. 2. An example of the BFM representation for four inputs case. Light gray nodes with zeros represent zero-valued variables while dark-grey nodes with one's denote one-valued variables. Empty nodes are for placeholders only, and indicate that their variables are removed. (a) shows the full set of FM variables, (b) only one-valued variables, (c) EBChI variables selected from a noise free training data set, and (d) EBFM represented with independent variables. The full FM lattice (a) can be simply derived from (d) using the FM's monotonicity property.

From the EBFM with independent variables that correspond to sets $B = \{B_1, B_2, \ldots, B_l\}$, the BChI of an observation $\mathbf{h}_j$ can be computed as follows:

1. Sort inputs in descending order. Let the sorting order be $x_{\pi_j(i)}, i = 1, 2, \ldots, N$, and the associated variables for the BChI are $\mu_B(S_{\pi_j(i)}), i = 1, 2, \ldots, N$.
2. Find the minimum k for which $S_{\pi_j(k)} \supseteq B_l \in B, \forall B_l$.
3. Return $\mathbf{h}_j(x_{\pi_j(k)})$ as the output.

4.2 Upper bound

To determine the upper bound on the number of variables in EBFM, we use a theorem by E. Sperner [27]. The theorem proves that if $B_1, B_2, \ldots, B_t$ are subsets of an N-element set B, such that no B_i is a subset of any other B_j, then

$$t \leq \binom{N}{[N/2]}, \tag{9}$$

where $[x]$ denotes the rounded integer value. As the independent variables in EBFM have the same definitions as the B_is above, Eq. (9) also gives the upper

bound on the number of variables in the EBFM. For 20 inputs, there can be at most 184, 756 independent variables (in median-like aggregation case), thus using only 17.62% of the total variables in the worst case. However, the actual usage (or saving) depends on the specific BFM; for example, min and max aggregation operators require 1 and N variables respectively.

5 Experiments

Experiments are conducted on synthetic data set for the following reasons. First and foremost, we know the true underlying BFM, which facilitates the comparison and investigation of the proposed method's behaviour, whereas in real world applications/data the true FM may never be known. Moreover, it is quite challenging to find real world examples of varying complexity while it is far easier to create a FM in synthetic data with different complexity. Synthetic experiments also give us insight into how the learning method will behave in different noisy contexts. The experiments are designed to compare the computational complexity as well as to measure the performance in terms of MSE from the predicted test labels using the learned BFM in a no noise as well as in a noisy environment.

5.1 No noise scenario

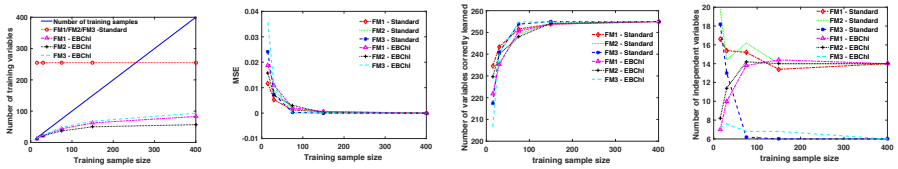
First, a training data set of $M = 500$ and $N = 8$ is generated pseudo-randomly from a uniform distribution, which is then partitioned to create five data-sets for five fold cross-validation. Each cross-validation data-set contains 400 training samples and 100 test samples. Then we created training data-sets of sample sizes 150, 75, 30, and 15 via random selection of instances from those of 400, 150, 75, and 30 respectively. Test data for all sample sizes remains the same. We specified three BFMs–BFM1, BFM2 and BFM3–in Table 1 using the EBFM representation with independent variables. In the table, each independent variable is denoted with the inputs' indices in the set, e.g., 12 stands for independent variable $\mu_B(\{x_1, x_2\})$. The independent variables in BFM1 lie in the lower part of the lattice (hence largest number of one valued variables) while those of BFM3 reside on the upper part. The BFM2 independent variables spread across the lattice from top to bottom. The labels for these BFMs are created without adding any noise, which also serve as the ground-truth for noisy system.

Table 1. The FMs used in the experiment

BFM	Independent variables														#var	# 1-var
1	8	56	67	57	345	346	347	1234	1235	1245	1236	1246	1237	1247	255	211
2	3568	3578	3678	4568	4578	4678	5678	12568	12578	12678	123458	123468	123478	1234567	255	59
3	67	68	78	123456	123457	123458									255	131

Figure 3 shows the results for different sample sizes. As can be seen, the number of variables increases linearly at small M, then remains constant for

large sample sizes, which is still far fewer than the standard optimization method. The average of the MSE as well as the number of variables correctly learned are approximately the same for both standard and EBChI (Figure 3(b) and (c)). An interesting observation from Figure 3(d) is that the EBChI has far fewer independent variables for 15 training samples, meaning the learned FM from the EBChI is less complex than the standard one.



(a) Average number of training variables over five iterations (b) Average mean squared error on test data (c) Average number of variables correctly learned (d) Average number of independent variables

Fig. 3. Noise-free training data of different sizes, $M = 15,\ 30,\ 75,\ 150,$ and 400.

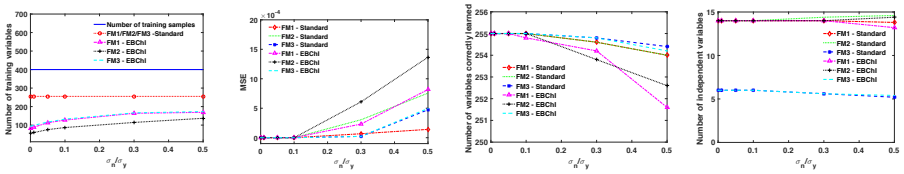
5.2 Noisy scenario

In most analysis of noisy systems, noise is modeled as Gaussian distribution, which provides a good approximation in many scenarios. Herein, we model the output as $y = C_{\mu_B}(\mathbf{h}) + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma_n^2)$. The observations in the training data are the same as those for the noise-free system; however, the labels are created by adding randomly generated values from a normal distribution of variance σ_n^2. We conducted experiments with five different variances, $\sigma_n/\sigma_y = \{0, 0.01, 0.05, 0.1, 0.3, 0.5\}$, where σ_y^2 is the variance of the true labels (actual label without noise). The standard deviations for FM1, FM2, and FM3 are 0.184, 0.1804, and 0.2055. The MSE was measured with respect to the true test labels.

Figure 4 compares the results for noisy data with 400 training samples. More noise means more variations around the true value, which results in selection of multiple variables for instances with the same sorting order. Thus, the number of training variables in EBChI increases with the noise level. The presence of noise equally affects both the EBChI and the standard BChI. When the FMs are learned with sufficient number of samples, the error is minimal–on the order of 10^-3 (Figure 4(c))–and there is a mismatch of only 4 out of 255 variables in the worst scenario; see Figure 4(c). As the result shows, both the standard method and EBChI are resilient to relatively moderate level of noise ($\sigma_n/\sigma_y = 0.3$).

6 Conclusion

In this paper, we proposed an efficient method to learn the BFM. Variable selection in the EBChI is driven by the observed instances whereas the standard



(a) (b) (c) (d)

Fig. 4. Results for data-driven learning with noise, with standard deviation, $\sigma_n = \{0.0\sigma_y, \}$. (a) average number of training variables over five iterations, (b) average mean squared error on test data, (c) average number of variables correctly learned, (d) corresponding independent variables

learning method uses full set of variables. This makes the EBChI tractable for a relatively large problem (large N) in contrast to the standard approach. As demonstrated by the results, learning with the EBChI is approximately equivalent to the standard BChI learning method for noisy and noise-free scenarios; therefore, it provides an efficient alternative for data-driven learning of the BFM. Moreover, we introduced a representation technique called the EBFM to describe a BFM minimally via independent variables. In future work, we will apply our technique to real world problems. Additionally, we will study which problems can be natural fit to BFM and which problems can be approximated by a BFM.

References

1. Price, S.R., Murray, B., Hu, L., Anderson, D.T., Havens, T.C., Luke, R.H., Keller, J.M.: Multiple kernel based feature and decision level fusion of ieco individuals for explosive hazard detection in flir imagery. In: SPIE. Volume 9823. (2016) 98231G–98231G–11
2. Tahani, H., Keller, J.: Information fusion in computer vision using the fuzzy integral. IEEE Transactions System Man Cybernetics **20** (1990) 733–741
3. Grabisch, M., Nicolas, J.M.: Classification by fuzzy integral: Performance and tests. Fuzzy sets and systems **65**(2-3) (1994) 255–271
4. Grabisch, M., Sugeno, M.: Multi-attribute classification using fuzzy integral. In: Fuzzy Systems, 1992., IEEE International Conference on, IEEE (1992) 47–54
5. Mendez-Vazquez, A., Gader, P., Keller, J.M., Chamberlin, K.: Minimum classification error training for Choquet integrals with applications to landmine detection. IEEE Transactions on Fuzzy Systems **16**(1) (2008) 225–238
6. Keller, J.M., Gader, P., Tahani, H., Chiang, J., Mohamed, M.: Advances in fuzzy integration for pattern recognition. Fuzzy sets and systems **65**(2-3) (1994) 273–283
7. Gader, P.D., Keller, J.M., Nelson, B.N.: Recognition technology for the detection of buried land mines. IEEE Transactions on Fuzzy Systems **9**(1) (2001) 31–43
8. Grabisch, M.: The application of fuzzy integrals in multicriteria decision making. European journal of operational research **89**(3) (1996) 445–456
9. Labreuche, C.: Construction of a Choquet integral and the value functions without any commensurateness assumption in multi-criteria decision making. In: EUSFLAT Conf. (2011) 90–97

10. Tomlin, L., Anderson, D.T., Wagner, C., Havens, T.C., Keller, J.M. In: Fuzzy Integral for Rule Aggregation in Fuzzy Inference Systems. Springer International Publishing (2016) 78–90
11. Pinar, A.J., Rice, J., Hu, L., Anderson, D.T., Havens, T.C.: Efficient multiple kernel classification using feature and decision level fusion. IEEE Transactions on Fuzzy Systems **PP**(99) (2016) 1–1
12. Pinar, A., Havens, T.C., Anderson, D.T., Hu, L.: Feature and decision level fusion using multiple kernel learning and fuzzy integrals. In: 2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE). (Aug 2015) 1–7
13. Hu, L., Anderson, D.T., Havens, T.C., Keller, J.M. In: Efficient and Scalable Nonlinear Multiple Kernel Aggregation Using the Choquet Integral. Springer International Publishing (2014) 206–215
14. Hu, L., Anderson, D.T., Havens, T.C.: Multiple kernel aggregation using fuzzy integrals. In: 2013 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE). (July 2013) 1–7
15. Grabisch, M., Nguyen, H.T., Walker, E.A.: Fundamentals of uncertainty calculi with applications to fuzzy inference. Volume 30. Springer Science & Business Media (2013)
16. Keller, J.M., Osborn, J.: Training the fuzzy integral. International Journal of Approximate Reasoning **15**(1) (1996) 1–24
17. Keller, J.M., Osborn, J.: A reward/punishment scheme to learn fuzzy densities for the fuzzy integral. Proc. Int. Fuzzy Sys. Assoc. World Cong (1995) 97–100
18. Mendez-Vazquez, A., Gader, P.: Sparsity promotion models for the Choquet integral. In: Foundations of Computational Intelligence, 2007. FOCI 2007. IEEE Symposium on, IEEE (2007) 454–459
19. Beliakov, G.: Construction of aggregation functions from data using linear programming. Fuzzy Sets and Systems **160**(1) (2009) 65–75
20. Islam, M.A., Anderson, D.T., Pinar, A.J., Havens, T.C.: Data-driven compression and efficient learning of the Choquet integral. IEEE Transactions on Fuzzy Systems **PP**(99) (2017) 1–1
21. Anderson, D.T., Islam, M., King, R., Younan, N.H., Fairley, J.R., Howington, S., Petry, F., Elmore, P., Zare, A.: Binary fuzzy measures and Choquet integration for multi-source fusion. In: 6th International Conference on Military Technologies. (May 2017)
22. Du, X., Zare, A., Keller, J.M., Anderson, D.T.: Multiple instance Choquet integral for classifier fusion. In: 2016 IEEE Congress on Evolutionary Computation (CEC). (July 2016) 1054–1061
23. Anderson, D.T., Price, S.R., Havens, T.C.: Regularization-based learning of the Choquet integral. In: 2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE). (July 2014) 2519–2526
24. Cho, S.B., Kim, J.H.: Combining multiple neural networks by fuzzy integral for robust classification. IEEE Transactions on Systems, Man, and Cybernetics **25**(2) (1995) 380–384
25. Olsson, C., Eriksson, A.P., Kahl, F.: Solving large scale binary quadratic problems: Spectral methods vs. semidefinite programming. In: Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on, IEEE (2007) 1–8
26. Glover, F., Kochenberger, G.A., Alidaee, B.: Adaptive memory tabu search for binary quadratic programs. Management Science **44**(3) (1998) 336–345
27. Sperner, E.: Ein satz über untermengen einer endlichen menge. Mathematische Zeitschrift **27**(1) (1928) 544–548