Fuzzy Choquet Integration of Deep Convolutional Neural Networks for Remote Sensing

Derek T. Anderson, Grant J. Scott, Muhammad Aminul Islam, Bryce Murray and Richard Marcum

Abstract What deep learning lacks at the moment is the heterogeneous and dynamic capabilities of the human system. In part, this is because a single architecture is not currently capable of the level of modeling and representation of the complex human system. Therefore, a heterogeneous set of pathways from sensory stimulus to cognitive function needs to be developed in a richer computational model. Herein, we explore the learning of multiple pathways–as different deep neural network architectures–coupled with appropriate data/information fusion. Specifically, we explore the advantage of data-driven optimization of fusing different deep nets–GoogleNet, CaffeNet and ResNet–at a per class (neuron) or shared weight (single data fusion across classes) fashion. In addition, we explore indices that tell us the importance of each network, how they interact and what aggregation was learned. Experiments are provided in the context of remote sensing on the UC Merced and WHU-RS19 data sets. In particular, we show that fusion is the top performer, each network is needed across the various target classes, and unique aggregations (i.e., not common operators) are learned.

Keywords Fuzzy integral · Convolutional neural network · Remote sensing

D. T. Anderson (✉) · G. J. Scott · R. Marcum
Electrical Engineering and Computer Science, University of Missouri, Columbia, MO, USA
e-mail: andersondt@missouri.edu

G. J. Scott
e-mail: grantscott@missouri.edu

R. Marcum
e-mail: ram7cd@missouri.edu

M. A. Islam · B. Murray
Electrical and Computer Engineering, Mississippi State University, Starkville, MS, USA
e-mail: mi160@msstate.edu

B. Murray
e-mail: bjm260@msstate.edu

W. Pedrycz and S.-M. Chen (eds.), *Computational Intelligence for Pattern Recognition*, Studies in Computational Intelligence 777,
https://doi.org/10.1007/978-3-319-89629-8_1

1 Introduction

We humans excel at many robust pattern recognition tasks in which computational systems can only perform well when limited in scope and constrained in operating environment. The human visual system is no exception. Humans develop at an early age a comprehensive visual processing and pattern recognition ability. Our vision allows us to process our physical environment (navigation) and facilitates many higher-level cognitive functions such as object classification and entity resolution. We accomplish this via a complex multistage visual system that begins with basic lightness and color receptors, then builds upon the perceived edges to derive shapes, spatial relationships, and eventually to organization of components into objects of interest – and this is before any higher level cognitive processing.

Deep neural network models follow a similar paradigm conceptually, extracting first edges and other simple geometric primitives in the lowest levels, then later mid-level assemblies of these primitives into visual concepts, which are then combined in higher-level layers as object components (blobs), that are eventually agglomerated into objects. These visual objects are agglomerated within fully connected neural layers for eventual classifications, which is an informational (cognitive) output. What deep architectures lack at the moment is the heterogeneous and dynamic capabilities of the human system, which is in part because a single architecture is not capable of the level of modeling and representation of the complex human system. Therefore, a heterogeneous set of pathways from sensory stimulus to cognitive function needs to be developed in a richer computational model. The model proposed in this chapter represents the learning of multiple pathways–as deep neural networks–coupled with appropriate information fusion. We feel fusion of the cognitive outputs (information) from multiple heterogeneous models (pathways) is the next step towards robust computational cognitive processing of visual, and visual-like, sensory data.

In general, *computational intelligence* (CI) is a branch of mathematics inspired by nature. Specifically, CI is associated with *neural networks* (NNs), *evolutionary algorithms* (EA) and *fuzzy set theory* (FST). NNs were established in 1943 by McCulloch and Pitts [1], FST was established in 1965 by Zadeh [2] and EAs were made popular by Holland in the early 1970s [3] (but arguably have roots going back as far as Turing in 1950). The point is, CI has existed in one form or another since the advent of *artificial intelligence* (AI). In this chapter, we focus on the intersection of NNs and FST for pattern recognition. In the last decade, substantial interest and effort has gone into *deep learning* (DL), a re-branding of NNs. This shift has forced us to re-address fundamental questions like; should humans design features (the classical approach to pattern recognition) or is a machine better at this task? Empirically, DL has more-or-less unanimously topped the charts in many domains (e.g., natural language processing [4, 5], vision [6–10], remote sensing [11–14]). However, while DL has generated great excitement, much remains to be explored and explained. In this chapter, we focus on the specific question of how to perform decision-level fusion of DL networks.

DL can be viewed as a generalization of the classical pattern recognition pipeline–e.g., pre-processing, feature extraction (selection and/or reduction), classification and post-processing. In some settings this is now being called *shallow learning* because there are only a few "layers" in the pattern recognition pipeline. In the context of computer vision, DL can also be decomposed into levels; "low" (e.g., signal/image analysis via convolution), "mid" and "high" (more AI than signal processing, e.g., MLP classification). In the extreme, DL is nothing more than a series of operations that transform data to decisions. The point is, fusion can (and often does) take place at different levels in pattern recognition/DL. For example, keeping with the fusion nomenclature of the *Joint Directors of Laboratories* (JDL) [15], some fusion algorithms do *signal-in-signal-out* (SISO), whereas others do *feature-in-feature-out* (FIFO) and *decision-in-decision-out* (DIDO). If we regard DL as a SIDO process (e.g., SI = image and DO = class label), then it can be decomposed into its corresponding SISO, SIFO, FIFO, FIDO, DIDO (and combinations therein). In summary, fusion is not as simple as "cram data into a DL and let it do its thing".

Herein, we restrict our analysis to *deep convolutional neural networks* (DCNNs) [6–10, 16, 17], versus *auto encoders* (AEs) [18–22], *deep belief nets* (DBNs) [23, 24], etc., for sake of discussion tractability. The reality is, we still know little-to-nothing about fundamental DL fusion questions such as; (i) how/where is fusion currently happening, (ii) based on our current set of neurons/transformations, what is mathematically expressible and what is not (but should be), (iii) how should we be performing fusion at different levels, (iv) how do we address heterogeneity with respect to semantics and/or uncertainty across data/information sources, and (v) how do we explain what fusion is doing (aka *explainable AI* (XAI)), to list a few. Independent of DL, fusion is a complicated topic that often means different things to different people in different fields (and even within the same field). Fusion is a wealth of challenges wrapped up into one term. Fusion ranges from data association (e.g., finding a one-to-one mapping between pixels in one sensor to pixels in another) to the mathematics of aggregation (specific functions/operators). In general, the idea of fusion is to obtain a "better" result than if we only used the individual inputs. However, better is not a well defined concept. In some applications, better might mean taking a set of inputs and reducing them into a single result that can be more efficiently or effectively used for visualization. Better could also refer to obtaining more desirable properties such as higher information content or lower conflict. In areas like pattern recognition, better often refers to some desirable property like more robust and generalizable solutions (e.g., classifiers). Regardless of the task at hand or the particular application, fusion is a core tool at the heart of numerous modern scientific thrusts.

In this chapter, we make the following contributions. First, we discuss two approaches for heterogeneous DCNN architecture fusion; density-based imputation and full *Choquet integral* (ChI) learning (per neuron and "shared weight"). Second, we outline indices for introspection and information theoretic indices to understand the capacity and integral (moving us closer to a so-called XAI solution versus black

box solution). Third, we demonstrate and analyze these ideas on remotely sensed data. Last, we provide open source code at www.derektanderson.com/FuzzyLibrary and www.github.com/scottgs/fi_library.

2 Deep Convolutional Neural Networks

To date, the AE [18–20], CNN [6–10, 16, 17], DBN [23, 24] and *recurrent NNs* (RNNs) [25, 26] are the most mainstream DLs. However, other DL approaches exist, e.g., deep inference nets ([27] Verma et al. Takagi-Sugeno-Kang deep net), deconvolution CNNs (specifically transpose matrix convolution) [28–30] and morphological shared weight neural networks [31, 32]. Herein, we focus on the CNN, which is by far the most employed and often the highest performer. With respect to the CNN, a number of architectures have been explored to date, e.g., AlexNet [7], GoogLeNet [17], VGGNet [33] and their derivatives. These architectures can be downloaded and extended (training, evaluation, visualization) via open source libraries like TensorFlow [34], CaffeNet [35], and MatConvNet [36]. The fundamental challenges of which architecture, how deep versus wide, hyperparameter tuning, what neuron types, how to transfer a DL from one domain to another (transfer learning [37]), and other questions are unanswered. Also, numerous challenges exist; e.g., lack of training data volume (and variety), class imbalance, dimensionality (spatial, temporal and spectral), explainable DL (what did the DL learn, versus a black box), to name a few. While DL has sparked a revolution in computer vision, pattern recognition and AI in general, an overwhelming number of theoretical and applied questions remain ripe for exploration.

In general, most CNNs consist of combinations of the following operations (see Fig. 1). First, let the input to the system, O_0, be a three dimensional data cube of size $N_0 \times M_0 \times D_0$; where N_0 and M_0 are spatial dimensions and D_0 is the temporal or spectral dimensionality (e.g., RGB imagery has $D_0 = 3$). **(Convolution)** The backbone of a CNN is filtering via convolution. Filtering can take a number of meanings, e.g., enhancement, denoising or detection. Convolution specifics include factors like (i) stride (spatial and/or spectral/temporal "jumps") and (ii) padding (if no padding is used then the spatial dimension shrinks). **(Pooling)** Pooling is often applied to reduce spatial dimensionality–and combat challenges related to affine variation, noise, etc. Most often, average and max pooling are used. **(Activation)** Nonlinearity is also typically applied, in the form of a function like hyperbolic tangent (*tanh*), sigmoid, or ReLU ($ReLU(x) = \max(0, x)$). **(Training Techniques)** In order to combat factors like sensitivity to parameter selection and overtraining, methods like dropout [38], regularization [39] and/or batch normalization [40] (addresses internal covariate shift and vanishing gradients) are often used. Beyond architecture, there are factors like GPU acceleration [41], training (e.g., *stochastic gradient decent* (SGD) [42], SGD with momentum [43, 44], AdaGrad [45], RMSProp [46] and ADAM [47]). The reader can refer to [39] for additional mathematical and algorithm

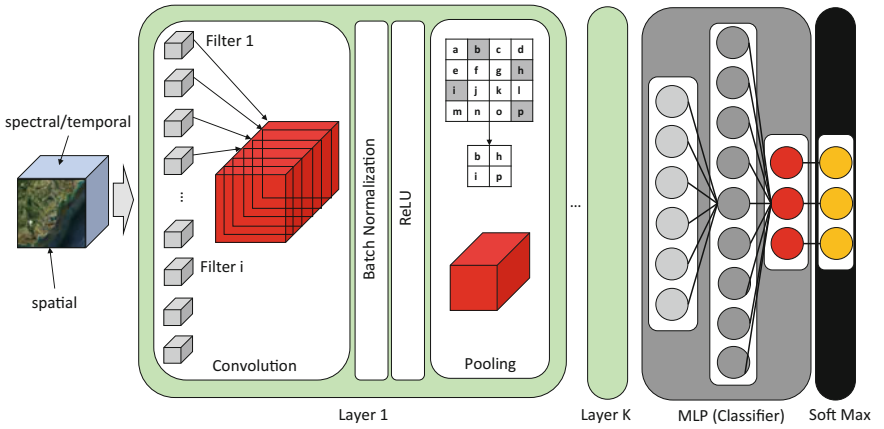

Fig. 1 Example CNN. Input is a 3D cube (x-y spatial, z spectral), green layers consist of subset of convolution (morphology, etc.), pooling (average, max, etc.), batch normalization (or other method to mitigate overfitting, vanishing gradients, internal covariate shift, etc.) and nonlinear function (e.g., ReLU activation). The output of the green layers are typically fed to a MLP and optional post-processing steps (e.g., soft max normalization)

details related to CNNs. The reader can also refer to [48] for a recent survey of DL in remote sensing (theory, applications and open challenges).

The idea of FST in NNs is not new. The reader can refer to the work of Pal and Mitra [49] for neuro-fuzzy pattern recognition. Pal, Mitra, and others (e.g., Keller and the fuzzy perceptron [50]), explored a variety of topics such as fuzzy min-max networks, fuzzy MLPs, and fuzzy Kohonen networks. In terms of aggregation, a few FST works have been explored to date. In 1992 [51], Yager put forth the *ordered weighted average* (OWA) [52]–which technically is a *linear combination of order statistics* (LCOS) since the weights are real-valued numbers (versus sets)–neuron. In 1995, Sung-Bae utilized the OWA for NN aggregation (at the decision/output level) [53]. In 1995, Sung-Bae et al. also explored the fuzzy integral, the Sugeno fuzzy integral not Sugeno's fuzzy ChI, for NN aggregation [54]. Specifically, they used the Sugeno λ-*fuzzy measure* (FM) and the densities were derived using their respective accuracy rates on training data. In 2017 [55], we (Scott et al.) used the Sugeno and ChIs for DCNN fusion. Specifically, Scott et al. used transfer learning to adapt GoogLeNet, AlexNet and ResNet50 from camera imagery to remote sensing imagery. Scott then applied different aggregations–fuzzy integral, voting, arrogance, and weighted sum–to these DCNNs. Scott's fusion was based on the Sugeno λ FM and the densities were (i) set to the normalized classifier accuracies and (ii) a GA learned the densities (which led to higher performance).

3 Fuzzy Measure and Fuzzy Integrals

The ChI has been successfully demonstrated in numerous applications; e.g., explosive hazard detection [56–58], computer vision [59], pattern recognition [60–64], remote sensing [65], multi-criteria decision making [66, 67], forensic anthropology [68–70], control [71], multiple kernel learning [56, 72–75], multiple instance learning [76], ontologies [77], missing data [78], and most relevant to the current chapter, DL [55]. The ChI is a nonlinear aggregation function that is parameterized by the FM (aka capacity). Countless mathematical variations of the fuzzy integral have been put forward for different reasons; e.g., address different types (i.e., real-valued, interval-valued, set-valued) of uncertainty in the integrand and/or measure, limit the number of input interactions for tractability, etc. Herein, we focus on and succinctly review just the real-valued discrete (finite X) ChI for DCNN fusion.

3.1 Discrete (Finite X) Fuzzy Measure

Let $X = \{x_1, x_2, \ldots, x_N\}$ be N sources, e.g., experts, sensors, or in the case of this chapter, DCNNs. The first action we face is how to assign "worth/utility" to different subsets of DCNNs. For example, the well-known backbone of calculus on real-valued domains is the Lebesgue measure; which coincides with length, area and hypervolume. However, when X is a discrete domain, e.g., set of DCNNs, what is the corresponding "measure"? In [59], Keller et al. first investigated the idea of using the fuzzy integral for pattern recognition. A FM is a function, μ, on the power set of X, 2^X, which satisfies (1) (boundary condition) $\mu(\emptyset) = 0$ (often $\mu(X) = 1$[1]) and (2) (monotonicity) if $A, B \subseteq X$ and $A \subseteq B$, then $\mu(A) \leq \mu(B)$.

3.2 Discrete (Finite X) Fuzzy Integral

The FM models important "interactions" (e.g., subjective worth, statistical correlation, etc.) between different source subsets. The input provided by our sources is $\{h(\{x_1\}), h(\{x_2\}), \ldots, h(\{x_N\})\}$. The fuzzy integral is a way to combine the integrand (h) information relative to the FM (μ). Let $h(\{x_i\}) \in \Re^{\geq 0}$ be the data from source i. The discrete (finite X) Sugeno FI is[2]

[1] If $\mu(X) < 1$, properties like idempotency and boundedness are not guaranteed.

[2] Due to the maximum (t-conorm) and minimum operators (t-norm), the Sugeno FI does not actually generate any possible number between the minimum and maximum of the inputs. Instead, it selects one of the FM or input values, i.e., at most one of $2^N + N$ values.

$$\int_S h \circ \mu = S_\mu(h) = \bigvee_{i=1}^{N} \left(h(\{x_{\pi(i)}\}) \wedge \mu(A_i) \right), \tag{1}$$

where π is the permutation $h(\{x_{\pi(1)}\}) \geq h(\{x_{\pi(2)}\}), \ldots, \geq h(\{x_{\pi(N)}\})$, $A_i = \{x_{\pi(1)}, \ldots, x_{\pi(i)}\}$ and $\mu(A_0) = 0$. The discrete (finite X) ChI is[3]

$$\int_C h \circ \mu = C_\mu(h) = \sum_{i=1}^{N} h(\{x_{\pi(i)}\}) \left[\mu(A_i) - \mu(A_{i-1}) \right]. \tag{2}$$

Since the ChI is a parametric aggregation function, once the FM is determined the ChI turns into a specific operator. For example: if $\mu(A) = 1, \forall A \in 2^X \setminus \emptyset$, the ChI becomes the maximum operator; if $\mu(A) = 0, \forall A \in 2^X \setminus X$, we recover the minimum; and if $\mu(A) = \frac{|A|}{N}$, we recover the mean. Each of these cases can be viewed as constraints or simplifications on the FM (and therefore the ChI). In general, the discrete ChI has $N!$ unique input sortings and each yields a linear convex sum operator.

3.3 Data-Driven Optimization

The first challenge we must confront is, where do we get the FM (μ) from? One option is to have an expert specify it. However, this is not practical (assuming the expert could even meaningfully assign values to the interactions) as the number of inputs (e.g., DLs) increases. Another option is we can specify or try to learn the worth of just the singletons (the densities). From there, a number of formulas can be used to impute (fill in) the missing variable values. Popular approaches include the Sugeno λ-FM and the S-Decomposable FM [79]. However, while convenient, most often we do not obtain the desired values for variables that we need. With respect to pattern recognition, the focus of this chapter, another route is to learn it from data. Next, we review one way to learn the FM, and therefore the ChI, in the context of DIDO for DL. However, the reader can refer to [80] for an efficient learning method with only data-supported variables and [81] for a review of alternative FM/ChI learning methods.

We quickly summarize one way to learn the full FM/ChI (see [82] for full mathematical explanation). Let $O = \{\mathbf{h}_j, y_j\},\ j = 1, 2, \ldots, M$, be M training examples; where $\mathbf{h}_j$ is the j-th instance with data from N inputs and y_j is the ground-truth for $\mathbf{h}_j$. The sum of squared error for training dataset O is

[3] The ChI is used frequently for various reasons; e.g., it is differentiable [62], for an additive (probability) measure it recovers the Lebesgue integral, it yields a wider spectrum of values between the minimum and maximum (versus the discrete and relatively small number of values that the Sugeno FI selects from), etc.

$$E(O, \mathbf{u}) = \sum_{j=1}^{M} e^j = \sum_{j=1}^{M} (\mathbf{c}_j^T \mathbf{u} - y_j)^2 = ||D\mathbf{u} - \mathbf{y}||_2^2, \tag{3}$$

where $\mathbf{u} = [\mu(\{x_1\}), \ldots, \mu(\{x_N\}), \mu(\{x_1, x_2\}), \mu(\{x_1, x_3\}), \ldots, \mu(X)]$ (lexicographic vector of size $2^N - 1$), $\mathbf{c}_j$ holds the coefficients of $\mathbf{u}$ for observation $\mathbf{h}_j$, e.g., for $N = 3$ and $h(\{x_2\}) \geq h(\{x_1\}) \geq h(\{x_3\})$,

$$c = [0, h(\{x_2\}) - h(\{x_1\}), 0, h(\{x_1\}) - h(\{x_3\}), 0, 0, h(\{x_3\})]\,,$$

$D = [\mathbf{c}_1 \ \mathbf{c}_2 \ \ldots \ \mathbf{c}_M]^T$ (full dataset), $\mathbf{y} = [y_1 \ y_2 \ \ldots \ y_M]^T$, and $||\cdot||_2$ is norm-2 operation, $\mathbf{u}$. The regularized SSE optimization problem is

$$\min_{\mathbf{u}} f(\mathbf{u}) = ||D\mathbf{u} - \mathbf{y}||^2 + \beta v(\mathbf{u}), \tag{4}$$

where $\beta \in \Re^{\geq 0}$ (regularization constant, which balances the "cost" (or penalty) of obtaining minimum function error relative to our desire to have minimal model complexity) and $v(\mathbf{u})$ is an index of model complexity (e.g., k-additive and Mobius, Gini-Simpson, ℓ_p-norm, etc. [83]), subject to the FM boundary and monotonicity conditions (see [82] for how to pack the constraints into a linear algebra expression), which can be solved via quadratic programming. Full code (including how to build C) can be found at www.derektanderson.com/FuzzyLibrary and www.github.com/scottgs/fi_library.

3.4 Explainable AI (XAI) Fusion

It is one thing to train a network and another to understand it! In this subsection, we highlight FM and ChI indices for the purpose of *explainable AI* (XAI).[4] XAI is an attempt to explain the inner operations of pattern recognition for purposes like describing it to others for domain knowledge transfer, trust, etc. The Shapley index addresses the importance or worth of each input (aka DL),

$$\Phi_\mu(i) = \sum_{K \subseteq X \setminus \{i\}} \zeta_{X,1}(K) \left(\mu(K \cup \{i\}) - \mu(K)\right), \tag{5}$$

where $\zeta_{X,1}(K) = \frac{(|X|-|K|-1)!|K|!}{|X|!}$, $K \subseteq X \setminus \{i\}$ denotes all proper subsets from X that do not include source i. The Shapley value of μ is the vector $\Phi_\mu = (\Phi_\mu(1), \ldots, \Phi_\mu(N))^t$ and $\sum_{i=1}^{N} \Phi_\mu(i) = 1$. The Shapley index can be interpreted as the average amount of *contribution* of source i across all coalitions. The next index informs us

[4] Go to www.derektanderson.com/FuzzyLibrary and www.github.com/scottgs/fi_library.

about how two inputs interact with one another (aka what advantage is there in combining DLs). The interaction index (Murofushi and Soneda [84]) between i and j is

$$\mathcal{I}_\mu(i,j) = \sum_{K \subseteq X \setminus \{i,j\}} \zeta_{X,2}(K)(\mu(K \cup \{i,j\}) - \mu(K \cup \{i\}) - \mu(K \cup \{j\}) + \mu(K)), \tag{6}$$

where $\zeta_{X,2}(K) = \frac{(|X|-|K|-2)!|K|!}{(|X|-1)!}$, $\mathcal{I}_\mu(i,j) \in [-1,1], \forall i,j \in \{1,2,\ldots,N\}$. A value of 1 (respectively, -1) represents the maximum complementary (respective redundancy) between i and j. Refer to [85] for further details about the interaction index, its connections to game theory and interpretations. Grabisch later extended the interaction index to the general case of any coalition [86],

$$\mathcal{I}_\mu(A) = \sum_{K \subseteq X \setminus A} \zeta_{X,3}(K,A) \sum_{C \subseteq A} (-1)^{|A \setminus C|} \mu(C \cup K), \quad i = 1,\ldots,N, \tag{7}$$

where $\zeta_{X,3}(K,A) = \frac{(|X|-|K|-|A|)!|K|!}{(|X|-|A|+1)!}$. Equation (7) is a generalization of both the Shapley index and Murofushi and Soneda's interaction index as $\Phi_\mu(i)$ corresponds with $\mathcal{I}_\mu(\{i\})$ and $\mathcal{I}_\mu(i,j)$ with $\mathcal{I}_\mu(\{i,j\})$.

The above indices are focused strictly on the FM. A different fundamental type of question is what "type" of aggregation is the ChI performing? Answering this question helps us understand how the DL information is being combined (e.g., in an optimistic, pessimistic, expected value like fashion, etc.). In [87], we established an index (D_1) to measure the degree to which a given FM/ChI is an maximum operator. In the following, we discuss the FM in terms of its underlying lattice structure. Let "layer k" (measure defined on sets of cardinality k) be denoted by $L(k)$, e.g., $L(1) = \{\mu(\{x_1\}), \mu(\{x_2\}), \mu(\{x_3\})\}$ for $N = 3$. Next, let $\mathbf{W} = \frac{[\frac{1}{N},\ldots,1]}{\sum_{i=1}^{N} \frac{i}{N}}$ be weights (penalty or costs) for each layer and

$$D_1(\mu) = \sum_{k=1}^{1} \frac{\mathbf{W}(k)}{2}(T_1 + T_4) + \left[\sum_{k=2}^{N} \frac{\mathbf{W}(k)}{3}(T_1 + T_2 + T_4)\right], \tag{8}$$

$T_1 = 1 - \left(\frac{\sum_{I \in L(k)} \mu(I)}{|L(k)|}\right)$, $T_2 = \left(\frac{\sum_{I \in L(i)} \mu(I)}{|L(k)|} - \frac{\sum_{J \in L(k-1)} \mu(J)}{|L(k-1)|}\right)$, $T_3 = \frac{\sum_{I \in L(k)} \mu(I)}{|L(k)|}$ and $T_4 = \frac{\sum_{I \in L(k)} (\mu(I) - T_3)^2}{|L(k)|-1}$. A value of $D_1 = 0$ means the ChI is the maximum operator. The distance of a learned capacity to a minimum operator (D_2), mean (D_3) and LCOS (D_4), for $\mathbf{W}_2 = \frac{[1,\ldots,\frac{1}{N-1}]}{\sum_{i=1}^{N-1} \frac{i}{N-1}}$, is

$$D_2(\mu) = \sum_{k=1}^{1} \frac{\mathbf{W}_2(k)}{2}(T_3 + T_4) + \left[\sum_{k=2}^{N-1} \frac{\mathbf{W}_2(i)}{3}(T_3 + T_2 + T_4)\right], \tag{9}$$

$$D_3(\mu) = \frac{1}{2^N - 2} \sum_{k=1}^{N-1} \sum_{I \in L(k)} \left| \mu(I) - \frac{k}{N} \right|, \tag{10}$$

$$D_4(\mu) = \frac{1}{N-1} \sum_{k=1}^{N-1} \sqrt{T_4}. \tag{11}$$

4 DCNN Fusion Based on Fuzzy Integration

The focus of this chapter is fusing different state-of-the-art DCNN architectures. However, the procedures outlined are applicable to other neural inputs (see Fig. 2).

4.1 DCNN Architectures Used for Fusion

The first NN used herein for fusion is CaffeNet [35], which is a derivative of AlexNet with similar structure, except that the order of pooling and normalization is reversed to

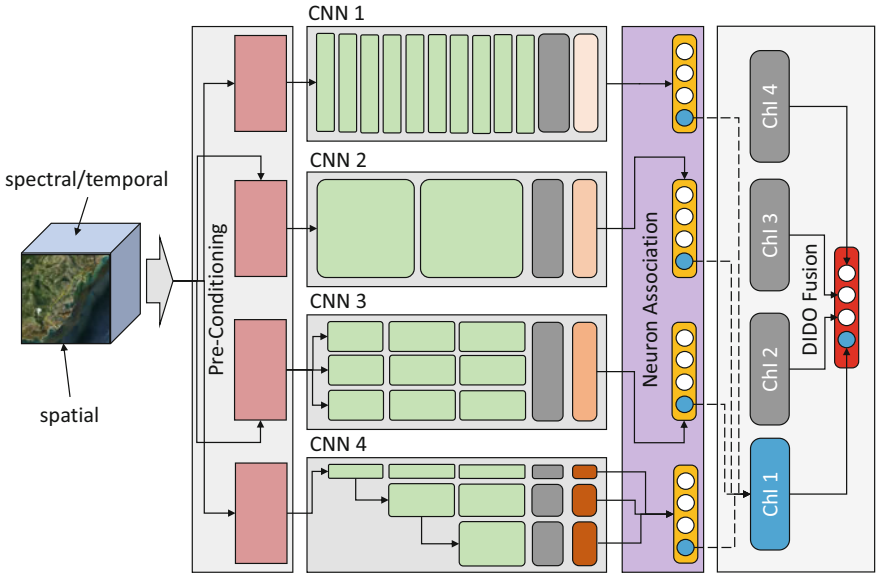

Fig. 2 Illustration of DIDO DCNN fusion. Note, many possibilities exist; e.g., variations in architecture, pre-conditioning/transforms (e.g., conversion to frequency analysis versus spatial domain, band selection or grouping, etc.), training data, etc. Next, neuron mapping/association is required followed by aggregation. Herein, a different fusion operator is learned per output neuron (versus shared fusions/weights)

reduce learnable parameters. CaffeNet contains five convolutional feature extraction steps and three fully connected layers for classification. Classification is performed with two fully connected inner product layers and a final soft-max layer for the network output. The output of soft-max layer is effectively a classification vector. CaffeNet represents the most simple and shallow of our DL investigated herein.

GoogLeNet [17] is a much deeper NN than CaffeNet–it has 27 parameterized layers. Because of this network depth, GoogLeNet has three classification outputs at various stages of the network to facilitate error back propagation. GoogLeNet's novel *inception layer* processes the input with max-pooling, 1×1, 3×3, and 5×5 convolutions simultaneously in a feature extraction step, and the outputs are concatenated as the layer output to achieve a multi-scale feature extraction. Using multiple convolutions at each stage follows the intuition that features from different kernel scales can be extracted and processed at the same time, thereby extracting multi-scale visual features. GoogLeNet is from a family of networks commonly referred to as *inception networks*.

ResNet [88] is a collection of DCNN architectures inspired by VGGNet [33]. In both ResNet and VGGNet, the primary kernels used to construct the convolution layers are 3×3. The architecture design incorporates the following rules to govern their structure. First, if the output of the feature map is the same, then the same number of 3×3 convolutional layers will be used. Second, if the output of the feature map is halved, then it will use twice as many 3×3 convolutional kernels The ResNet architectures employ residual connections that bypass two or more convolution layers at a time, allowing error to better propagate backward through the network. These are commonly referred to as *residual networks*, and here the ResNet50 and ResNet101 architectures are used within our experimental design. These networks have 50 and 101 feature extraction steps, respectively.

4.2 Transfer Learning, Neuron Association and Conditioning

If we design a set of custom DCNNs then it is trivial to ensure a bijection (one-to-one and onto) output neuron mapping. However, if existing community pretrained DCNNs (GoogLeNet, AlexNet, etc.) are leveraged–a task encountered frequently in practice–then this is not guaranteed. One way to resolve the one-to-one mapping task is to replace and retrain the DCNN classification layers per the labels for the task at hand. This is a type of transfer learning that keeps the feature layers intact. In [89], we (i) replaced and retrained the classification layers and we also (ii) updated the feature weights (e.g., convolution layers). Thus, we built custom classifiers for remote sensing of aerial imagery based on a network initialized by ground-perspective RGB imagery. In addition, data augmentation via rotation and image flipping was applied as well. However, we remark that other avenues exist; e.g., one could manually resolve the mapping or use an automated method based on an ontology. Regardless, using multiple custom or pretrained networks of different architectures raises another

question; are the outputs numerically (e.g., all in [0, 1]) and semantically "to scale" (e.g., does a (e.g., $a = 0.5$) in domain i map to a in the other domains). One way to mitigate this issue in practice is to add a soft max normalization (aka normalized exponent function) layer after the raw neuronal output layer. For example, if η is the soft max output for neuron o_j then the soft max function is $\eta(o_j) = \frac{e^{o_j}}{\sum_{n=1}^{N} e^{o_n}}$. Thereby, we bound the domain of input for the subsequent fusion layer of our pattern recognition system, ensure the data across networks and neurons is well conditioned.

4.3 Imputation: λ-FM ChI

The first fusion approach explored here is to exploit our knowledge about the performance of the individual DCNNs on training data [54, 55]. A classical approach to obtaining the remaining $2^N - 2 - N$ FM values (beyond the densities) is the Sugeno λ-measure. For sets $A, B \subseteq X$, such that $A \cap B = \phi$,

$$\mu_\lambda(A \cup B) = \mu_\lambda(A) + \mu_\lambda(B) + \lambda\mu_\lambda(A)\mu_\lambda(B), \tag{12}$$

for some $\lambda > -1$. In particular, Sugeno showed that λ can be found by solving

$$\lambda + 1 = \prod_{i=1}^{N} (1 + \lambda\mu(x_i)), \lambda > -1, \tag{13}$$

where there exists exactly one real solution such that $\lambda > -1$. Some advantages of the Sugeno λ-measure include its simplicity, the N densities can be more tractable to acquire, fewer number of parameters can help address overfitting (versus using the full 2^N variables), and it is a probability measure when $\lambda = 0$. However, there is no guarantee in practice that the values that it imputes are what we actually need. Simply speaking, more information or a different imputation formula may be required; e.g., the S-Decomposable imputation formula, $\mu(A) = \bigvee_{i \in A}(\mu(x_i))$ (where $\bigvee$ is a t-conorm). Algorithm (1) describes how to use the Sugeno λ-measure to fuse a set of pretrained DLs based on individual performance for density.

Algorithm 1 λ-FM Based Imputation of ChI from Pre-Trained DCNNs

INPUT: DL_i - N DCNNs (B neurons each); $\bar{O}$ - labeled training data
1. Run each DCNN on $\bar{O}$, get *overall accuries* (OA); $a_{b,i} \in [0, 1]$ (i.e., performance of DL i on class b)
2. Assign the ith density its corresponding OA; i.e., $\mu_{\lambda_b}(x_i) = a_{b,i}$
3. Find λ_b (using $\{\mu_{\lambda_b}(\{x_1\}), ..., \mu_{\lambda_b}(\{x_N\})\}$) for Sugeno λ-FM (solve Eq. (13))
4. Recursively calculate $\mu_{\lambda_b}(A), \forall A \in 2^X \setminus \{\{x_1\}, ..., \{x_N\}\}$, using the densities and λ_b (Eq. (12))
OUTPUT: B full fuzzy measures - $\{\mu_{\lambda_1}, ..., \mu_{\lambda_B}\}$

4.4 Optimization Approach: Learning the Full ChI

As stated in Sect. 4.3, there is no guarantee that imputation from densities results in the input interactions that we desire (and thus results in an appropriate aggregation operator). Algorithm (2) shows how to use quadratic programming for acquisition of the full FM for DIDO fusion of DCNNs (Algorithm (3) is how to learn a single "shared" FM to be applied to all neurons). Thus, training data is directly used to learn these crucial interactions–which means better selection of appropriate aggregation operator. However, as we discuss in [80], this process can lead to a big boost in performance but it is not without flaw. Specifically, in [80] we show that training data only typically supports a subset of FM variables. In return, we put forth an extended optimization of the ChI by (1) identifying which variables are supported by data, (2) optimizing just those variables and then (3) looking at imputation methods to infer the value of data unsupported variables based on application specific criteria. We do not have space to go into depth about the extension here, the reader can refer to [80] for full details.

Algorithm 2 Learn a Full FM/ChI Per Class for a Set of Pre-Trained DCNNs

INPUT: DL_i - N DCNNs (B neurons each); $\bar{O}$ - training data; β - regularization
1. Per class/output neuron (b), run each instance ($1 \leq j \leq |\bar{O}|$) through each DCNN (i); get $h_j^b(x_i)$ terms
2. Per neuron (b), construct the individual D_b from the $h_j^b(x_i)$ terms
3. Run B independent QPs (on the D_b respectively); yielding $\{\mu_1, ..., \mu_B\}$
OUTPUT: B full fuzzy measures - $\{\mu_1, ..., \mu_B\}$

Algorithm 3 Learn a Single "Shared Weight" Full FM/ChI for Pre-Trained DCNNs

INPUT: DL_i - N DCNNs (B neurons each); $\bar{O}$ - training data; β - regularization
1. Per class/output neuron (b), run each instance ($1 \leq j \leq |\bar{O}|$) through each DCNN (i); get $h_j^b(x_i)$ terms
2. Per neuron (b), construct the individual D_b from the $h_j^b(x_i)$ terms
3. Use QP to solve $\left(||D_1\mathbf{u} - \mathbf{y_1}||_2^2 + ... + ||D_B\mathbf{u} - \mathbf{y_B}||_2^2 + \beta v(\mathbf{u})\right)$; yields μ
OUTPUT: Full fuzzy measure - μ

5 Experiments

In this chapter, two benchmark remote sensing datasets suitable for classification tasks of objects or land-cover/land-use are used. Remote sensing data represents a significant pattern recognition challenge. As can be seen in Figs. 3 and 9 below, the variability and complexity of overhead imagery is immense. The visual cues exist at multiple levels: fine-scale (e.g. airplane shapes, vehicle presence, etc.) to

Fig. 3 Sample image chips from the 21 class UCM benchmark dataset, each 256×256 pixels approximately 0.3m *ground sampling distance* (GSD) spatial resolution. Classes in left-to-right, top-down order: 1 *agricultural*, 2 *airplane*, 3 *baseball diamond*, 4 *beach*, 5 *buildings*, 6 *chaparral*, 7 *dense residential*, 8 *forest*, 9 *freeway*, 10 *golf course*, 11 *harbor*, 12 *intersection*, 13 *medium residential*, 14 *mobile home park*, 15 *overpass*, 16 *parking lot*, 17 *river*, 18 *runway*, 19 *sparse residential*, 20 *storage tanks*, and 21 *tennis court*. In Sect. 5.1, neuron indices are used instead of text descriptions for sake of compactness

large-scale (e.g., road way configurations in overpasses versus intersections versus freeway). In fact the entire field of photo-interpretation revolves around developing human expertise in this pattern recognition task. For each of the datasets herein DCNNs were trained using the techniques in [89], including transfer learning and data augmentation via rotation and flipping. The trained DCNNs are then used in a locked state, i.e., no further learning happens in DL during the fusion stage. The training of the DCNNs are done in five-fold, cross validation manner; such that we have 5 sets of 80% training and 20% testing for both datasets. Per DCNN fold, three-fold CV fusion is used (due to limited data).

5.1 UC Merced (UCM) Dataset

The UC Merced (UCM) benchmark dataset [90, 91] has been used in a wide range of remote sensing research, including prior work in classification of objects and land-cover such as [55, 89, 92]. Figure 3 shows exemplar image chips from each class of the UCM dataset. The dataset includes 21 classes that are a mix of objects (airplane, baseball diamond, etc.) and landcover (beach, chaparral, etc.). We see that some classes, e.g., harbor and parking lot, are complex compositions of sub-entities (boats and vehicles); while others are general structural patterns of shapes (e.g., intersection and baseball diamonds).

Table 1 is the result of fusion on the UCM dataset. First, we see that aggregation outperforms no aggregation (i.e., the individual DCNNs) in four out of five folds.

Table 1 Fusion results for the UCM dataset

	Method									
	ChI Per Neuron	ChI Shared	SLFM Shared	CNet	GNet	RNet 50	RNet 101	Max	Avg	Min
Fold 1	0.979	0.977	0.984	0.957	0.957	0.985	0.973	0.978	0.981	0.976
Fold 2	0.991	0.994	0.993	0.964	0.983	0.978	0.988	0.993	0.994	0.993
Fold 3	0.994	0.990	0.996	0.971	0.985	0.992	0.988	0.996	0.996	0.998
Fold 4	0.992	0.996	0.996	0.988	0.980	0.983	0.988	0.996	0.992	0.998
Fold 5	0.989	0.985	0.989	0.976	0.973	0.983	0.980	0.989	0.989	0.986

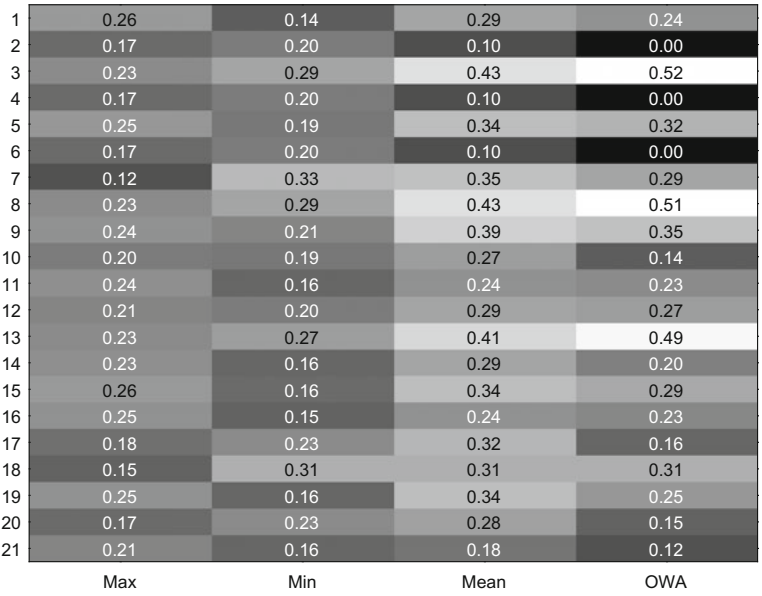

Fig. 4 Color coded matrix showing the distances obtained using the four reported indices of introspection ($D_1(\mu)$ to $D_4(\mu)$) relative to the learned full ChI per neuron on fold 1 of the UCM dataset. y-axis is the neuron index (see Fig. 3) and x-axis is the distance measure. Neurons two, four and six are OWA operators (but not min, max or mean like)

Second, we see that min, max and average (basic aggregation operators) do well in comparison to the ChI. However, these three operators are specific instances of the ChI, which informs us that there are challenges with variety and thus generalizability of this particular data set (otherwise they should have been selected). Next, it is interesting to see that the shared weight fusion solutions do as well as they do. It is our suspicion–something to be explored in future work–that a shared FM for the ChI helps combat overfitting. It is also our suspicion–again, subject of future work–that while the Sugeno λ-FM would not be our first choice, it might also help combat overfitting as it has just N parameters versus the otherwise $2^N - 1$. However, the performance of the individual DCNNs (which were used as the densities) are so high that ultimately this forces the Sugeno λ-FM to more-or-less be the maximum operator.

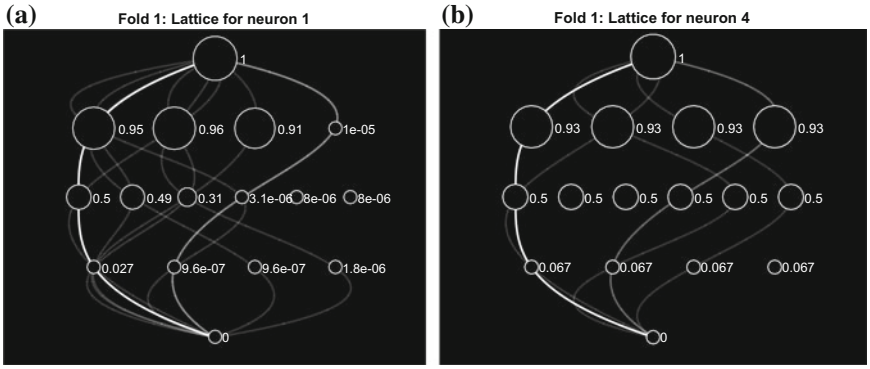

Fig. 5 Example of two full FMs for the **a** first and **b** fourth neuron in fold 1 of the UCM dataset. "Layer" l (from bottom to top) in the image denotes FM variables with cardinally l. Thus, layer 0 (bottom node) is the empty set, the next layer is the singletons, top is $\mu(X)$, etc. Each variable is presented in lexicographic order, i.e., layer 2 is $\{x_1, x_2\}$, $\{x_1, x_3\}$, $\{x_1, x_4\}$, $\{x_2, x_3\}$, $\{x_2, x_4\}$ and $\{x_3, x_4\}$. The nodes are also drawn size-wise proportional to their value (a minimum size and maximum was specified to make them still show up for 0 valued variables). In addition, the "paths" drawn indicate the visitation frequency (the brighter the line, the higher the visitation) for the test data in fold 1. Furthermore, the fourth neuron learned an OWA with weights $(0.067, 0.433, 0.43, 0.07)^t$ – a trimmed mean operator. Conversely, neuron one is more complex to decode. It does not reduce into a single compact description like an OWA. However, we can view it in terms of the $N!$ walks (possible sorts). Since the $h(\{x_1\}) \geq h(\{x_2\}) \geq h(\{x_3\}) \geq h(\{x_4\})$ is encountered frequently, we decode and analyze its weights. The linear convex sum weights for the ChI of this walk (sorting) are $(0.027, 0.473, 0.45, 0.05)$ respectively. Thus, it is a weighted average of GoogLeNet and ResNet50. This analytic process can be repeated for the other $N! - 1$ walks if desired

Next, Fig. 4 gives us a feel for what type of aggregation strategy is being used for the 21 classes. Again, the max, min and mean are all OWAs, so we can start first with analyzing column four. There are three neurons (2, 4 and 6–i.e., airplane, beach and chaparral) that learned an OWA. The other neurons have learned something more unique, which helps justify the inclusion of the ChI versus say a simpler operator (see Fig. 5(a)). At that, none of the learned OWAs are that similar to our extreme markers of max (a t-conorm or union like operator), min (a t-conorm or intersection like operator) or average (an expected value like operator). For example, Fig. 5(b) shows one of these OWA operators, which breaks down into a trimmed mean operator.

Last, Fig. 6 shows the FM and Shapley values. While it is more-or-less impossible to read individual values in these plots, they show that there is no consensus in values nor importance of DCNNs. Meaning, different output neurons (classes) appears to use these different DCNNs in different ways. Furthermore, Fig. 7 shows the corresponding interaction index values. These values also reinforce the complex interplay and back-and-forth exchange of complementary, independent and redundant information between DCNNs across output neurons (classes). In total, the combination of

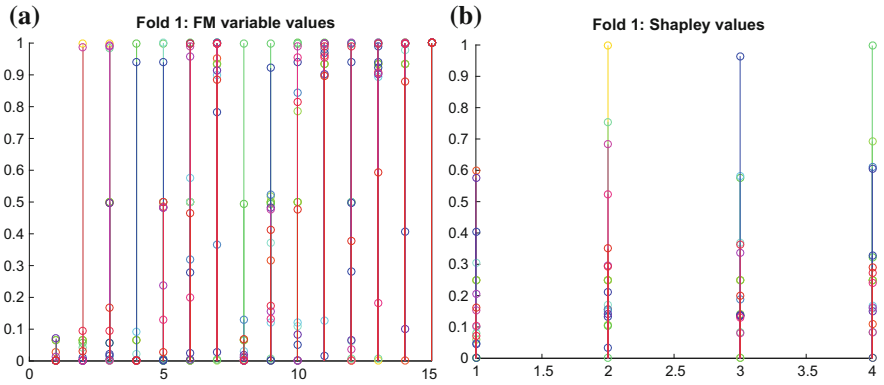

Fig. 6 Learned full ChI per neuron on fold 1 of the UCM dataset. **a** Plot of the $2^4 - 1$ binary encoded FM variables, i.e., for $N = 3$ the order is ($x1$, $x2$, $x12$, $x3$, $x13$, $x23$, $x123$, $x4$, $x14$, $x24$, $x124$, $x34$, $x134$, $x234$, $x1234$). This plot shows the agreement/disagreement of variable values across the 21 neurons. If all neurons required the same fusion then each x-axis location would have a single convergent set of circles (FM variable values). However, we can see that each x-axis location (FM variable) has for the most part significant variation (outside the CaffeNet density). **b** Plot of the 4 neuron Shapley index values across the 21 neurons. Again, this plot shows the variety of values learned. With respect to individual output neurons, some NNs could be eliminated. However, across the 21 neurons, no NN can be eliminated (we would expect to see approximately all zero values for that Shapley if so)

analysis of underlying aggregation function, importance of individual DCNNs and their pair-wise interaction behavior help the claim that performance appears to be improving due to diversity in the way these DCNNs operate. This is in line with our intuition about these DCNNs based on the ways their architectures were created.

Last, Fig. 8 shows example images *missed* by our fusion approach. As the reader can visually verify, these examples are extreme and represent incorrectly labeled or fundamentally ambiguous labels. We would not expect fusion to be able to fix this type of problem. At that, it is hard to say that the DCNNs should have got these, as a human might just as well mistaken them.

5.2 WHU-RS19 (RSD) Dataset

The WHU-RS19 (RSD) dataset is composed of 600×600 pixel, JPEG compressed images [93]. This class includes 19 classes, and approximately 50 chips per class. This imagery was screen scrapped from Google Earth, and therefore they are of variable spatial resolutions. Figure 9 shows exemplar image chips from each class of the RSD benchmark dataset. Similar to the UCM dataset, this dataset is a mixture of landcover and objects within the image chips. Table 2 shows the result of fusion, Fig. 10 are the indices of introspection, Fig. 11 are example lattices, Fig. 12 are the FM and the Shapley indices and Fig. 13 are example interaction indices. Overall, we see

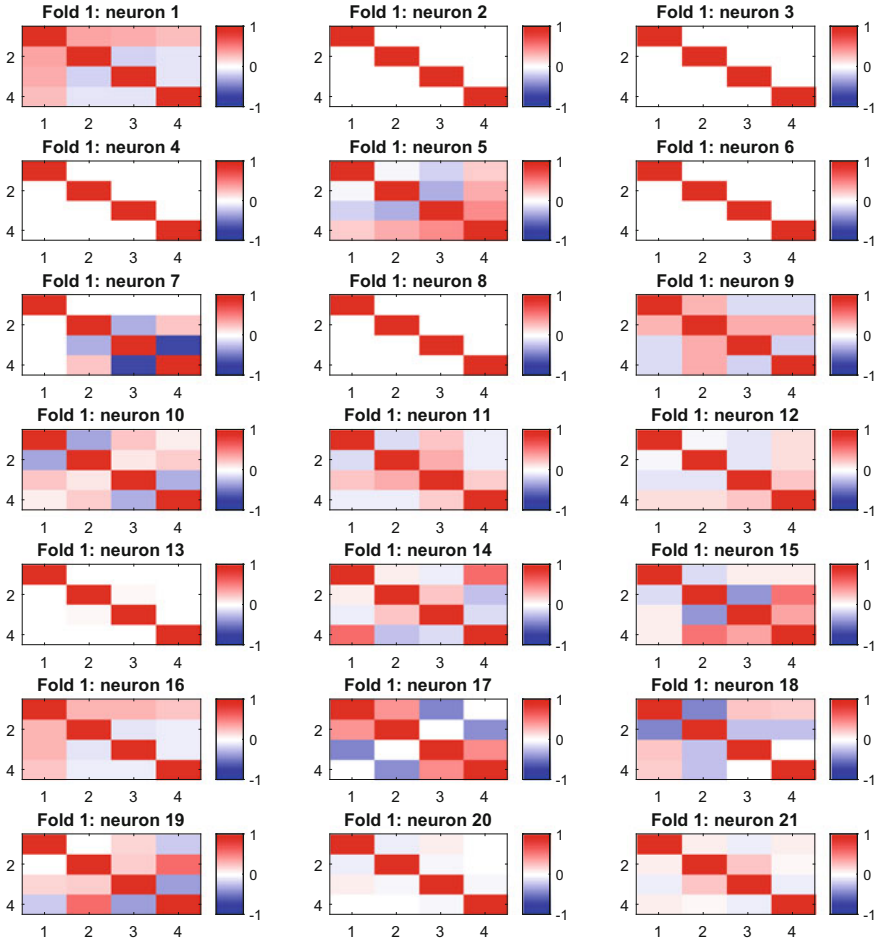

Fig. 7 Interaction index values for the learned full ChI per neuron on fold 1 of the UCM dataset. Index 1 is CaffeNet, 2 is GoogLeNet, 3 is ResNet50 and 4 is ResNet101. Consider neuron 1. CaffeNet has positive interactions (complementary information) with the other three NNs (0.37, 0.34 and 0.3 respectively). On the other hand, GoogLeNet has negative interaction values (redundancy) with the ResNet NNs (–0.19 and –0.1 respectively). The two ResNet NNs have a negative interaction index of –0.12. Also, in neuron 7, CaffeNet has approximately a zero interaction index with the other NNs (independence), whereas GoogLeNet has a value of –0.29 with ResNet50 and a positive interaction value of 0.22 with ResNet101. Last, ResNet50 and ResNet101 have a large negative interaction index of –0.72 with each other

Misclassified Image	Other images from correct class	Predicted Exemplar
 **(a)**	 Dense Residential	 Mobile Home Park
 **(b)**	 Intersection	 Overpass
 **(c)**	 Medium Residential	 Dense Residential
 **(d)**	 Golf Course	 Forest
 **(e)**	 Dense Residential	 Medium Residential

Fig. 8 Five images *missed* by the fusion framework; **a** dense residential misclassified as mobile home park, **b** (incorrectly labeled) intersection misclassified as overpass (correct label), **c** medium residential misclassified as dense residential, **d** (incorrectly labeled) golf course misclassified as forest, and **e** dense residential misclassified as medium residential

Fig. 9 Sample image chips from the 19 class RSD benchmark dataset, each 600×600 pixels of various spatial resolution. Classes in left-to-right, top-down order: 1 *airport*, 2 *beach*, 3 *bridge*, 4 *commercial area*, 5 *desert*, 6 *farmland*, 7 *football field*, 8 *forest*, 9 *industrial area*, 10 *meadow*, 11 *mountain*, 12 *park*, 13 *parking lot*, 14 *pond*, 15 *port*, 16 *railway station*, 17 *residential area*, 18 *river*, and 19 *viaduct*. In Sect. 5.2, neuron indices are used instead of text descriptions for sake of compactness

Table 2 Fusion results for the RSD dataset

	Method								
	ChI Per Neuron	ChI Shared	SLFM Shared	CNet	GNet	RNet50	Max	Avg	Min
Fold 1	0.989	0.991	0.991	0.982	0.977	0.988	0.991	0.991	0.991
Fold 2	0.992	0.984	0.992	0.978	0.994	0.989	0.987	0.992	0.987
Fold 3	0.984	0.992	0.979	0.955	0.988	0.966	0.979	0.979	0.979
Fold4	0.983	0.983	0.983	0.983	0.960	0.971	0.983	0.988	0.987
Fold 5	0.998	1.00	1.00	0.977	0.994	0.994	1.00	1.00	1.00

the same general trend (as the UCM dataset). Namely, (i) aggregation outperforms no aggregation in general and (ii) there are challenges with variety (and therefore generalizability) in the RSD data set as well.

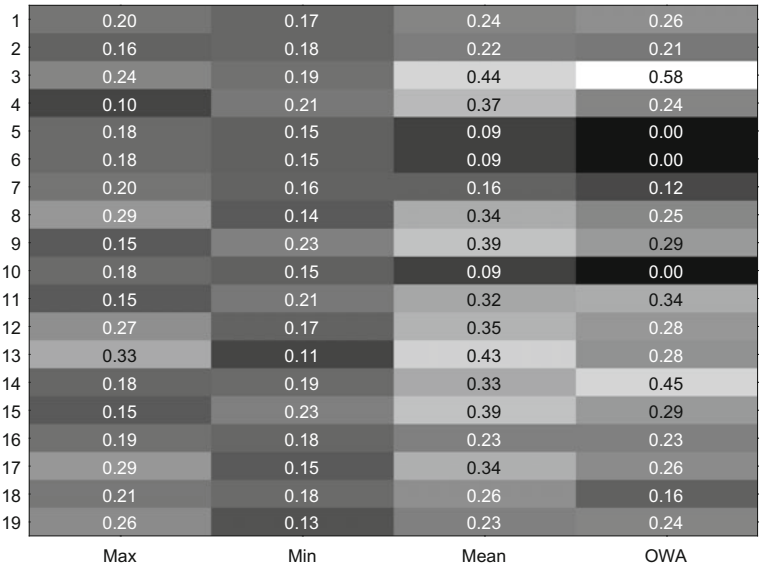

Fig. 10 Color coded matrix showing the distances obtained using the four reported indices of introspection ($D_1(\mu)$ to $D_4(\mu)$) relative to the learned full ChI per neuron on fold 1 of the RSD dataset. y-axis is the neuron index (see Fig. 9) and x-axis is the distance measure

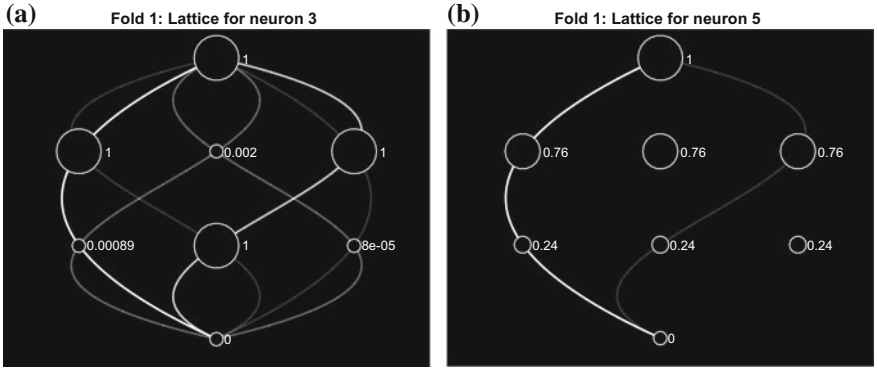

Fig. 11 Example FMs for fold 1 of the RSD dataset. Neuron three is for all intents and purposes a binary FM (see [94] for a formal characterization of binary FMs, the resultant FI and efficient ways of representing and learning such a function). For binary FMs, the Sugeno FI and the ChI are mathematically equivalent [94]. The FI is acting like a "dynamic maximum operator" with respect to FM variables that have a value one–or conversely a "dynamic minimum" with respect to zero valued FM variables. For example, if $h(\{x_1\}) \geq h(\{x_2\}) \geq h(\{x_3\})$ (aka CaffeNet is more confident than GoogLeNet followed by ResNet) then we take the output of GoogLeNet. However, if $h(\{x_2\})$ (GoogLeNet) is the most confident then we take its input. This reasoning can be followed to get similar stories for the other $N! - 2$ walks. Another interesting observation of neuron 3, versus neuron 5, is a slightly more diverse visitation (walk) pattern

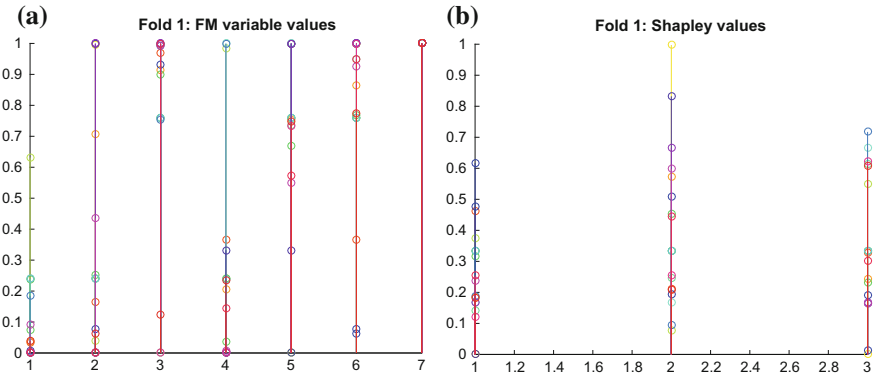

Fig. 12 Binary encoded $2^3 - 1$ FM variables and the 3 Shapley index values for the nineteen output neurons in the RSD dataset. As demonstrated in the UCM dataset, great variability exists in FM variable and Shapley values for these nineteen output neurons

6 Conclusion and Future Work

In summary, this chapter outlined a data-driven method for optimizing Choquet integral-based fusion of heterogeneous deep convolutional neural networks for pattern recognition in remotely sensed data. To the best of our knowledge, no one has previously learned the full fuzzy Choquet integral for fusing neural networks, just density-based fuzzy measures. This chapter brought together state-of-the-art advancements in two important parts of computational intelligence; fuzzy set theory and neural networks. Specifically, CaffeNet, GoogLeNet, ResNet50 and ResNet101 were fused at the per-output-neuron and with respect to a single "shared weight" solution. In a strive for explainable AI, versus black box solutions, different indices of introspection of the Choquet integral and information theoretic indices of the fuzzy measure were highlighted for analysis of the final deep learning solution. These indices showed us that there does appear to be diversity in these different heterogeneous DCNNs. Two benchmark remote sensing datasets were used, UCM and RSD, and our fused results showed improvement over the individual deep learners. However, this data set and DCNNs were saturated and therefore limited data (both volume and variety) existed for training fusion. Last, analysis of mislabeled imagery from fusion revealed incorrectly labeled data and ambiguous image chips that would lead to a human mislabeling imagery.

While encouraging, more research (theory and application) is needed. In future work, we will migrate our Choquet integral solution into a strictly neural representation for optimization and speed. Furthermore, we will move away from DIDO and explore fusion neurons at various layers in the network. We will also investigate what types of neural inputs should be fed to DIDO fusion; e.g., combinations of deep and shallow, different convolution map scales, etc. Future work will also include simultaneously learning the DCNNs and our fusion operators (they are learned inde-

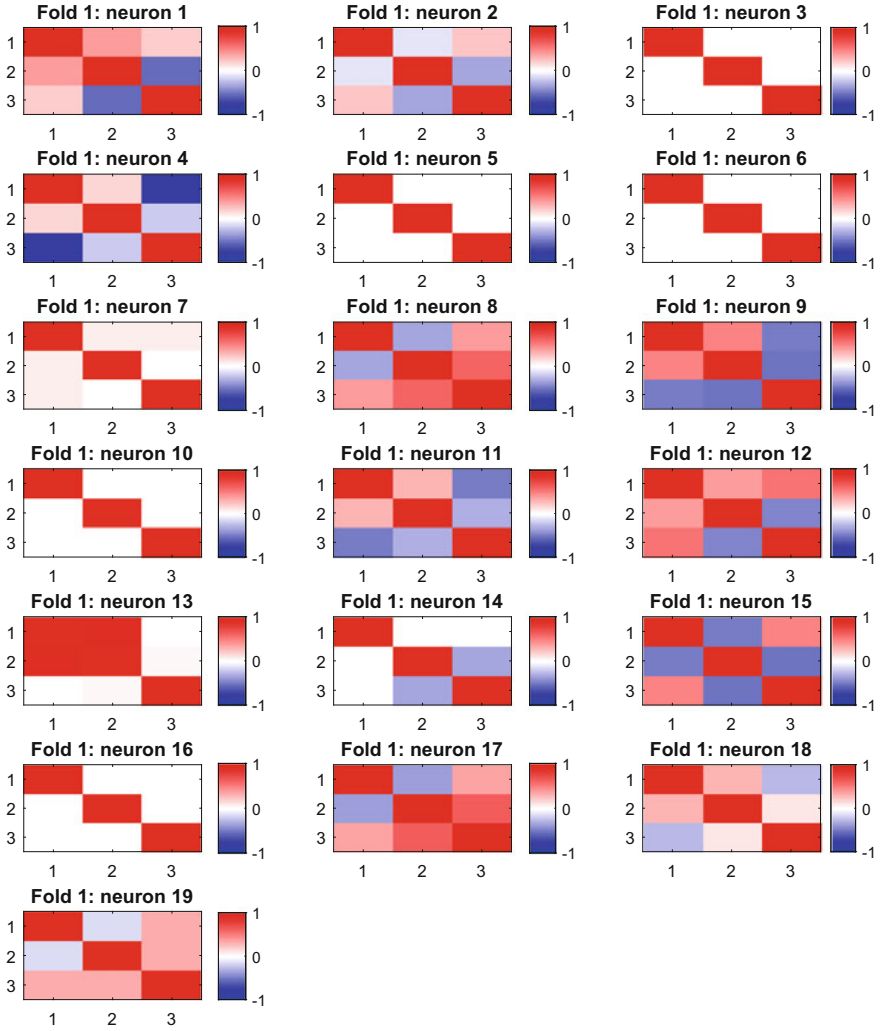

Fig. 13 Interaction index values for nineteen outputs in the RSD dataset. Index 1 is CaffeNet, 2 is GoogLeNet and 3 is ResNet50

pendently herein). Last, we will look to use our explainable AI methods to make improvements to the fusion and DCNNs, manually as well as possibly using them directly computationally to promote diversity and/or aid in the design of our networks.

References

1. W.S. McCulloch, W. Pitts, A logical calculus of the ideas immanent in nervous activity. Bull. Math. Biophys. **5**(4), 115–133 (1943)
2. L.A. Zadeh, Fuzzy sets. Inf. Control **8**(3), 338–353 (1965)
3. J.H. Holland, *Adaptation in Natural and Artificial Systems, 1992* (Ann Arbor, University of Michigan Press, MI, 1975)
4. R. Collobert, J. Weston, A unified architecture for natural language processing: deep neural networks with multitask learning, in *Proceedings of the 25th International Conference on Machine Learning* (ACM, New York, 2008), pp. 160–167
5. R. Socher, C.C. Lin, C. Manning, A.Y. Ng, Parsing natural scenes and natural language with recursive neural networks, in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)* (2011), pp. 129–136
6. K. Fukushima, S. Miyake, Neocognitron: a self-organizing neural network model for a mechanism of visual pattern recognition, in *Competition and Cooperation in Neural Nets* (Springer, Berlin, 1982), pp. 267–285
7. A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in *Advances in Neural Information Processing Systems* (2012), pp. 1097–1105
8. D. Ciregan, U. Meier, J. Schmidhuber, Multi-column deep neural networks for image classification, in *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, New York, 2012), pp. 3642–3649
9. C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 2818–2826
10. D.C. Ciresan, U. Meier, L.M. Gambardella, J. Schmidhuber, Deep, big, simple neural nets for handwritten digit recognition. Neural Comput. **22**(12), 3207–3220 (2010)
11. C. Bentes, D. Velotto, S. Lehner, Target classification in oceanographic sar images with deep neural networks: architecture and initial results, in *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)* IEEE, New York, 2015), pp. 3703–3706
12. W. Huang, L. Xiao, Z. Wei, H. Liu, S. Tang, A new pan-sharpening method with deep neural networks. IEEE Geosci. Remote Sens. Lett. **12**(5), 1037–1041 (2015)
13. X. Chen, S. Xiang, C.L. Liu, C.H. Pan, Vehicle detection in satellite images by hybrid deep convolutional neural networks. IEEE Geosci. Remote Sens. Lett. **11**(10), 1797–1801 (2014)
14. J. Yue, W. Zhao, S. Mao, H. Liu, Spectral-spatial classification of hyperspectral images using deep convolutional neural networks. Remote Sens. Lett. **6**(6), 468–477 (2015)
15. A.N. Steinberg, C.L. Bowman, F.E. White, Revisions to the JDL data fusion model, in *Handbook of Data Fusion* (1999)
16. M.D. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, in *European Conference on Computer Vision* (Springer, Berlin, 2014), pp. 818–833
17. C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 1–9
18. G.E. Hinton, R.R. Salakhutdinov, Reducing the dimensionality of data with neural networks. Science **313**(5786), 504–507 (2006)
19. P. Vincent, H. Larochelle, Y. Bengio, P.-A. Manzagol, Extracting and composing robust features with denoising autoencoders, in *Proceedings of the 25th International Conference on Machine learning* (ACM, New York, 2008), pp. 1096–1103
20. M. Chen, Z. Xu, K. Weinberger, F. Sha, *Marginalized denoising autoencoders for domain adaptation* (2012). arXiv preprint arXiv:1206.4683
21. Q. Fu, X. Yu, X. Wei, Z. Xue, Semi-supervised classification of hyperspectral imagery based on stacked autoencoders, in *Eighth International Conference on Digital Image Processing (ICDIP 2016)*, 100332B-100332B. International Society for Optics and Photonics (2016)

22. J. Geng, J. Fan, H. Wang, X. Ma, B. Li, F. Chen, High-resolution sar image classification via deep convolutional autoencoders. IEEE Geosci. Remote Sens. Lett. **12**(11), 2351–2355 (2015)
23. G.E. Hinton, Deep belief networks. Scholarpedia **4**(5), 5947 (2009)
24. H. Lee, R. Grosse, R. Ranganath, A.Y. Ng, Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations, in *Proceedings of the 26th Annual International Conference on Machine Learning* (ACM, New York, 2009), pp. 609–616
25. T. Mikolov, M. Karafiát, L. Burget, J. Cernock'y, S. Khudanpur, Recurrent neural network based language model, in *Interspeech*, vol. 2 (2010), 3 p
26. K. Funahashi, Y. Nakamura, Approximation of dynamical systems by continuous time recurrent neural networks. Neural Netw. **6**(6), 801–806 (1993)
27. S. Rajurkar, N.K. Verma, Developing deep fuzzy network with takagi sugeno fuzzy inference system, in *2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)* (2017), pp. 1–6. https://doi.org/10.1109/FUZZ-IEEE.2017.8015718
28. L. Xu, J.S. Ren, C. Liu, J. Jia, Deep convolutional neural network for image deconvolution, in *Advances in Neural Information Processing Systems* (2014), pp. 1790–1798
29. M.D. Zeiler, D. Krishnan, G.W. Taylor, R. Fergus, Deconvolutional networks, in *2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, New York, 2010), pp. 2528–2535
30. M.D. Zeiler, G.W. Taylor, R. Fergus, Adaptive deconvolutional networks for mid and high level feature learning, in *2011 IEEE International Conference on Computer Vision (ICCV)* (IEEE, New York, 2011), pp. 2018–2025
31. Y. Won, P.D. Gader, P.C. Coffield, Morphological shared-weight networks with applications to automatic target recognition. IEEE Trans. Neural Netw. **8**(5), 1195–1203 (1997)
32. X. Jin, C.H. Davis, Vehicle detection from high-resolution satellite imagery using morphological shared-weight neural networks. Image Vis. Comput. **25**(9), 1422–1431 (2007)
33. K. Simonyan, A. Zisserman, *Very Deep Convolutional Networks for Large-scale Image Recognition* (2014). arXiv preprint arXiv:1409.1556
34. M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G.S. Corrado, A. Davis, J. Dean, M. Devin, et al., *Tensorflow: Large-scale Machine Learning on Heterogeneous Distributed Systems* (2016). arXiv preprint arXiv:1603.04467
35. Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell, Caffe: convolutional architecture for fast feature embedding, in *Proceedings of the 22nd ACM International Conference on Multimedia* (ACM, New York, 2014), pp. 675–678
36. A. Vedaldi, K. Lenc, Matconvnet: convolutional neural networks for matlab, in *Proceedings of the 23rd ACM International Conference on Multimedia* (ACM, New York, 2015), pp. 689–692
37. J. Yosinski, J. Clune, Y. Bengio, H. Lipson, How transferable are features in deep neural networks? in *Advances in Neural Information Processing Systems* (2014), pp. 3320–3328
38. N. Srivastava, G.E. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
39. I. Goodfellow, Y. Bengio, A. Courville, *Deep Learning* (MIT Press, 2016)
40. S. Ioffe, C. Szegedy, Batch normalization: accelerating deep network training by reducing internal covariate shift, in *International Conference on Machine Learning* (2015), pp. 448–456
41. L. Brown, *Deep Learning with GPUs*, http://www.nvidia.com/content/events/geoInt2015/
42. L. Bottou, Stochastic gradient learning in neural networks. Proc. Neuro-Names **91**(8) (1991)
43. B.T. Polyak, Some methods of speeding up the convergence of iteration methods. USSR Comput. Math. Math. Phys. **4**(5), 1–17 (1964)
44. I. Sutskever, J. Martens, G. Dahl, G. Hinton, On the importance of initialization and momentum in deep learning, in *International Conference on Machine Learning* (2013), pp. 1139–1147
45. J. Duchi, E. Hazan, Y. Singer, Adaptive subgradient methods for online learning and stochastic optimization. J. Mach. Learn. Res. **12**(Jul), 2121–2159 (2011)
46. T. Tieleman, G. Hinton, Lecture 6.5-rmsprop: divide the gradient by a running average of its recent magnitude. Coursera: Neural Netw. Mach. Learn. **4**(2), 26–31 (2012)
47. D. Kingma, J. Ba, Adam: a method for stochastic optimization, in *3rd International Conference for Learning Representations* (2015)

48. J.E. Ball, D.T. Anderson, C.S. Chan, A comprehensive survey of deep learning in remote sensing: theories, tools and challenges for the community. J. Appl. Remote Sens. (2017)
49. S.K. Pal, S. Mitra, *Neuro-fuzzy Pattern Recognition: Methods in Soft Computing* (Wiley Inc, New Jersey, 1999)
50. J.M. Keller, D.J. Hunt, Incorporating fuzzy membership functions into the perceptron algorithm. IEEE Trans. Pattern Anal. Mach. Intell. **6**, 693–699 (1985)
51. R.R. Yager, Applications and extensions of owa aggregations. Int. J. Man Mach. Stud. **37**(1), 103–122 (1992)
52. R.R. Yager, On ordered weighted averaging aggregation operators in multicriteria decision-making. IEEE Trans. Syst. Man Cybern. **18**(1), 183–190 (1988)
53. C. Sung-Bae, Fuzzy aggregation of modular neural networks with ordered weighted averaging operators. English. Int. J. Approx. Reas. **13**(4), 359–375 (1995)
54. S.B. Cho, J.H. Kim, Combining multiple neural networks by fuzzy integral for robust classification. IEEE Trans. Syst. Man Cybern. **25**(2), 380–384 (1995)
55. G.J. Scott, R.A. Marcum, C.H. Davis, T.W. Nivin, Fusion of deep convolutional neural networks for land cover classification of high-resolution imagery. IEEE Geosci. Remote Sens. Lett. (2017)
56. S.R. Price, B. Murray, L. Hu, D.T. Anderson, T.C. Havens, R.H. Luke, J.M. Keller, Multiple kernel based feature and decision level fusion of IECO individuals for explosive hazard detection in flir imagery, in *SPIE*, vol. 9823 (2016), pp. 98231G-98231G-11. https://doi.org/10.1117/12.2223297
57. R.E. Smith, D.T. Anderson, A. Zare, J.E. Ball, B. Alvey, J.R. Fairley, S.E. Howington, Genetic programming based Choquet integral for multi-source fusion, in *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE* (2017)
58. R.E. Smith, D.T. Anerson, J.E. Ball, A. Zare, B. Alvey, Aggregation of Choquet integrals in GPR and EMI for handheld platform-based explosive hazard detection, in *Proceedings of the SPIE 10182, Detection and Sensing of Mines, Explosive Objects, and Obscured Targets XXII* (2017)
59. H. Tahani, J. Keller, Information fusion in computer vision using the fuzzy integral. IEEE Trans. Syst. Man Cybern. **20**, 733–741 (1990)
60. M. Grabisch, J.-M. Nicolas, Classification by fuzzy integral: performance and tests. Fuzzy Sets Syst. **65**(2–3), 255–271 (1994)
61. M. Grabisch, M. Sugeno, Multi-attribute classification using fuzzy integral, in *IEEE International Conference on Fuzzy Systems, 1992* (IEEE, New York, 1992), pp. 47–54
62. A. Mendez-Vazquez, P. Gader, J.M. Keller, K. Chamberlin, Minimum classification error training for Choquet integrals with applications to landmine detection. IEEE Trans. Fuzzy Syst. **16**(1), 225–238 (2008). https://doi.org/10.1109/TFUZZ.2007.902024. ISSN: 1063-6706
63. J.M. Keller, P. Gader, H. Tahani, J. Chiang, M. Mohamed, Advances in fuzzy integration for pattern recognition. Fuzzy Sets Syst. **65**(2–3), 273–283 (1994)
64. P.D. Gader, J.M. Keller, B.N. Nelson, Recognition technology for the detection of buried land mines **9**(1), 31–43 (2001)
65. G.J. Scott, D.T. Anderson, Importance-weighted multi-scale texture and shape descriptor for object recognition in satellite imagery, in *2012 IEEE International Geoscience and Remote Sensing Symposium* (2012), pp. 79–82. https://doi.org/10.1109/IGARSS.2012.6351632
66. M. Grabisch, The application of fuzzy integrals in multicriteria decision making. Eur. J. Oper. Res. **89**(3), 445–456 (1996)
67. C. Labreuche, Construction of a Choquet integral and the value functions without any commensurateness assumption in multi-criteria decision making, in *EUSFLAT Conference* (2011), pp. 90–97
68. D.T. Anderson, P. Elmore, F. Petry, T.C. Havens, Fuzzy Choquet integration of homogeneous possibility and probability distributions. Inf. Sci. **363**, 24–39, (2016). https://doi.org/10.1016/j.ins.2016.04.043. http://www.sciencedirect.com/science/article/pii/S0020025516302961. ISSN: 0020-0255

69. D.T. Anderson, T.C. Havens, C. Wagner, J.M. Keller, M.F. Anderson, D.J. Wescott, Extension of the fuzzy integral for general fuzzy set-valued information **22**(6), 1625–1639, (2014). https://doi.org/10.1109/TFUZZ.2014.2302479. ISSN: 1063-6706
70. M. Anderson, D.T. Anderson, D.J. Wescott, Estimation of adult skeletal age-at-death using the sugeno fuzzy integral. Am. J. Phys. Anthropol. **142**(1), 30–41 (2010)
71. L. Tomlin, D.T. Anderson, C. Wagner, T.C. Havens, J.M. Keller, *Fuzzy integral for rule aggregation in fuzzy inference systems* (Springer International Publishing, Berlin, 2016), pp. 78–90. https://doi.org/10.1007/978-3-319-40596-4_8
72. A.J. Pinar, J. Rice, L. Hu, D.T. Anderson, T.C. Havens, Efficient multiple kernel classification using feature and decision level fusion. PP(99), 1 (2016). ISSN: 1063-6706. https://doi.org/10.1109/TFUZZ.2016.2633372
73. A. Pinar, T.C. Havens, D.T. Anderson, L. Hu, Feature and decision level fusion using multiple kernel learning and fuzzy integrals, in *2015 IEEE International Conference on Fuzzy Systems (FUZZIEEE)* (2015), pp. 1–7. https://doi.org/10.1109/FUZZ-IEEE.2015.7337934
74. L. Hu, D.T. Anderson, T.C. Havens, J.M. Keller, Efficient and scalable nonlinear multiple kernel aggregation using the choquet integral, in *Information Processing and Management of Uncertainty in Knowledge-Based Systems: 15th International Conference, IPMU, Montpellier, France, July 15–19, 2014, Proceedings. Part I* (Springer International Publishing, Berlin, 2014), pp. 206–215
75. L. Hu, D.T. Anderson, T.C. Havens, Multiple kernel aggregation using fuzzy integrals, in *2013 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)* (2013), pp. 1–7. https://doi.org/10.1109/FUZZ-IEEE.2013.6622312
76. X. Du, A. Zare, J.M. Keller, D.T. Anderson, Multiple instance Choquet integral for classifier fusion, in *2016 IEEE Congress on Evolutionary Computation (CEC)* (2016), pp. 1054–1061. https://doi.org/10.1109/CEC.2016.7743905
77. M. Al Boni, D.T. Anderson, R.L. King, Hybrid measure of agreement and expertise for ontology matching in lieu of a reference ontology. Int. J. Intell. Syst. **31**(5), 502–525 (2016). https://doi.org/10.1002/int.21792. ISSN: 1098-111X
78. M.A. Islam, D.T. Anderson, F. Petry, D. Smith, P. Elmore, The fuzzy integral for missing data, in *2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)* (2017), pp. 1–8. https://doi.org/10.1109/FUZZ-IEEE.2017.8015475
79. M. Sugeno, Theory of fuzzy integrals and its applications. Ph.D. thesis, Tokyo Institute of Technology, (1974)
80. M.A. Islam, D.T. Anderson, A.J. Pinar, T.C. Havens, Data-driven compression and efficient learning of the Choquet Integral. IEEE Trans. Fuzzy Syst. PP(99), 1 (2017). https://doi.org/10.1109/TFUZZ.2017.2755002. ISSN: 1063-6706
81. J.M. Keller, J. Osborn, Training the fuzzy integral. Int. J. Approx. Reas. **15**(1), 1–24 (1996)
82. D.T. Anderson, S.R. Price, T.C. Havens, Regularization-based learning of the Choquet integral, in *2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)* (2014), pp. 2519–2526. https://doi.org/10.1109/FUZZ-IEEE.2014.6891630
83. A.J. Pinar, D.T. Anderson, T.C. Havens, A. Zare, T. Adeyeba, Measures of the shapley index for learning lower complexity fuzzy integrals. Granul. Comput. 1–17 (2017)
84. T. Murofushi, S. Soneda, Techniques for reading fuzzy measures (iii): interaction index, in *9th Fuzzy System Symposium* (Sapporo, Japan, 1993)
85. M. Grabisch, M. Roubens, An axiomatic approach to the concept of interaction among players in cooperative games. Int. J. Game Theory **28**(4), 547–565 (1999)
86. M. Grabisch, An axiomatization of the shapley value and interaction index for games on lattices, in *SCIS-ISIS* (2004)
87. S.R. Price, D.T. Anderson, C. Wagner, T.C. Havens, J.M. Keller, *Indices for introspection on the Choquet integral, in Advance Trends in Soft Computing* (Springer, Berlin, 2014), pp. 261–271
88. K. He, X. Zhang, S. Ren, J. Sun, *Deep Residual Learning for Image Recognition* (2015). arXiv preprint arXiv:1512.03385
89. G.J. Scott, M.R. England, W.A. Starms, R.A. Marcum, C.H. Davis, Training deep convolutional neural networks for land-cover classification of high-resolution imagery. IEEE Geosci. Remote Sens. Lett. **14**(4), 549–553 (2017)

90. S.D. Newsam, *UC Merced Land Use Dataset* (2010), http://vision.ucmerced.edu/datasets/landuse.html
91. Y. Yang, S. Newsam, Bag-of-visual-words and spatial extensions for land-use classification, in *ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (ACM GIS)* (2010), 666 p
92. C. Chen, B. Zhang, H. Su, W. Li, L. Wang, Land-use scene classification using multi-scale completed local binary patterns. Signal Image Video Proc. **10**(4), 745–752 (2016)
93. D. Dai, W. Yang, Satellite image classification via two-layer sparse coding with biased image representation. IEEE Geosci. Remote Sens. Lett. **8**(1), 173–176 (2011)
94. D.T. Anderson, M. Islam, R. King, N.H. Younan, J.R. Fairley, S. Howington, F. Petry, P. Elmore, A. Zare, Binary fuzzy measures and Choquet integration for multi-source fusion, in *6th International Conference on Military Technologies* (2017)