Received July 24, 2020, accepted August 10, 2020, date of publication August 14, 2020, date of current version August 27, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3016784

A Robust Decision-Making Framework Based on Collaborative Agents

JOHANA M. FLOREZ-LOZANO[1], (Student Member, IEEE), FABIO CARAFFINI[2], (Member, IEEE), CARLOS PARRA[1], (Member, IEEE), AND MARIO GONGORA[2]

[1]Electronic Department, School of Engineering, Pontificia Universidad Javeriana, Bogotá 110231, Colombia
[2]School of Computer Science and Informatics, Institute of Artificial Intelligence, De Montfort University, Leicester LE1 9BH, U.K.

Corresponding author: Fabio Caraffini (fabio.caraffini@dmu.ac.uk)

This work was supported by the De Montfort University through the De Montfort University Global Challenges Research Fund (GCRF) 2020 Internally Funded Project [1], Leicester, U.K.

ABSTRACT Making decisions under uncertainty is very challenging but necessary as most real-world scenarios are plagued by disturbances that can be generated internally, by the hardware itself, or externally, by the environment. Hence, we propose a general decision-making framework which can be adapted to optimally address the most heterogeneous real-world domains without being significantly affected by undesired disturbances. Our paper presents a multi-agent based structure in which agents are capable of individual decision-making but also interact to perform subsequent, and more robust, collaborative decision-making processes. The complexity of each software agent can be kept quite low without a deterioration of the performance since an intelligent and robust-to-uncertainty decision-making behaviour arises when their locally produced measure of support are shared and exploited collaboratively. We show that by equipping agents with classic computational intelligence techniques, to extract features and generate measure of supports, complex hybrid multi-agent software structures capable of handling uncertainty can be easily designed. The resulting multi-agent systems generated with this approach are based on a two-phases decision-making methodology which first runs parallel local decision making processes to then aggregate the corresponding outputs to improve upon the accuracy of the system. To highlight the potential of this approach, we provided multiple implementations of the general framework and compared them over four different application scenarios. Results are promising and show that having a second collaborative decision-making process is always beneficial.

INDEX TERMS Collaborative systems, decision-making, distributed systems, intelligent decision-making, fuzzy systems, multi-agent systems, neural systems, neuroevolution.

I. INTRODUCTION

When facing Decision-Making (DM) problems using data or sensors in real-world adverse environments, performance is significantly affected by numerous environmental factors such as sunlight, humidity, temperature, atmospheric pressure and many others. For this reason, in application domains such as remote sensing, precision agriculture and landmine detection [2]–[6], robust systems need to be based on multiple and varied sensors to mitigate the effect of environmental disturbances [7]–[10]. However, regardless of the number of sensors used, data will have high levels of noise and uncertainty even in less adverse environments. For example, wireless body sensor networks used to acquire information from patients in hospitals can easily be affected by disturbances and introduce uncertainty in medical data [11]. In these cases, as in most real-world applications, designing a robust DM system capable of aggregating multiple sources of information and make a final decision is not an easy tasks. Issues encountered while designing DM systems capable of handling multiple sources of information include the sensitivity of each source to external factors that might impact with different strengths on different sub-systems. This will bias the final decision.

The literature offers examples of advanced DM systems meant to deal with the aforementioned problem. Recently,

The associate editor coordinating the review of this manuscript and approving it for publication was Zeshui Xu.

good results were achieved with ''ensemble classifiers'' as e.g. the one used in [11] to predict heart diseases at early stages. More complex DM structures, based on ''intelligent'' computational methods, are also currently being under investigation. In the humanitarian demining field, it is worth mentioning the work done under the TIRAMISU project, with a particular reference to the ''fuzzy logic'' system proposed in [12] to detect mine-fields by considering multiple information sources from an aerial data acquisition module. In similar application domains, e.g. detecting human bodies trapped under debris, also genetic fuzzy systems are being proposed to optimise the accuracy of the classification outcome [13], as well as neural systems and other Machine Learning (ML) methods to extract relevant features leading to a very robust and accurate final decision [14]–[18]. In this light, the study in [6] has shown that DM in environments with high levels of uncertainty and disturbances, is indeed possible with acceptable accuracy. This has been achieved by coordinating multiple ML-based DM systems around a decentralised and asynchronous logic allowing for real-time classifications made locally, i.e. individually by each DM subsystem, and globally, i.e. in a collaborative manner. The collaborative DM takes into consideration measures of support exchanged among subsystems. This builds upon previous decentralised DM systems [19]–[22], whose structure was proven to reduce delays due to data processing, be more resilient to external disturbances and alleviate the sensitivity to external disturbance factors [23]; the resulting DM system in [6] was designed to be embedded in a multi-sensors platform and was successfully used to detect buried Improvised Explosive Devices (IEDs).

A practical and efficient approach to implement such a decentralised systems, is to describe and design them as Multi-Agent System (MAS). These systems can have a varying number of independent software/hardware units, i.e. the agent, operating asynchronously and concurrently thus increasing the execution speed of several tasks required to achieve a final common goal [24], [25]. If this goal is to make a final decision, based on several other measures of support calculated by each agent, then a MAS can be turned in a much more suitable scheme for robust DM than classic centralised ones [26]. This resulted in a very effective approach for social decision making in a computer mediated environment [27], where the collaborative DM capabilities of the agents was achieved by taking into consideration social parameters as ''trust'', etc., but also in other fields such as malware detection [28], network clock synchronisation and mobile modular robotics [29].

Motivated by this research problem, we propose an original multi-agent DM framework which is easily extendable with a generic number n of agents in order to adapt to several real-world scenarios and 1) operate on multiple sources of information as most appropriate or required (i.e. one agent per source); 2) alternatively, work on the same source of information, e.g. a single data set, but simultaneously using n different techniques for exacting features and generating individual measures of support for the final DM process. Hence, the proposed framework is capable of performing independent and real-time ''local'' DM by each agent but also allows for a robust ''collaborative'' DM process in which each agent aggregates the measure of support coming from the other agents to minimise the uncertainty in making the final classification decision. Unlike other studies in the field, we propose to use a combination of state-of-the-art ML methods (i.e. neural networks, fuzzy systems, etc.) to implement each agent's ''brain''. The structure of the selected ML method for implementing the local and the collaborative DM systems is not fixed but rather evolved during the training process to optimally adapt to the available data. Consequently, the proposed software infrastructure is dynamic and self-adaptive. In this article, the presented framework is also evaluated with varying number of agents, from one to six, to show its performances over four popular data sets and highlight its potential. The data sets were chosen to showcase the adaptability of the proposed DM approach to different scenarios (i.e. Medical, object classification and synthetic data set).

More in detail, the remainder of the article is structured as follows:

- Section II presents the propose multi-agent decision-making (RDM_{CA}) system and describes all its components and working mechanisms;
- Section III specifies the criteria to select the best performance of each of the decision-making systems.
- Section IV details the implementation and the test system used for four data-sets available on-line.
- Section V display and comments on the obtained results;
- Section VI draws the conclusion of this piece of research and speculate on potential ways forward.

II. THE MULTI-AGENT DECISION-MAKING FRAMEWORK

To achieve our goal, as stated in the introduction, we design a flexible and modular software architecture so that it can be easily adapted and extended to tackle a variety of application domains. The basic component of such a structure is the agent, a software unit defined by four attributes, namely

- A Local Decision-Making (LDM) system;
- A Collaborative Decision-Making (CDM) system;
- A Source of information (e.g. a set S of data);
- A Communication Link (CL);

allowing for the agent to make decisions independently, but also collaboratively by communicating with its peers. Hence, no restrictions in the number of agents is given, i.e. as many agent as required for the specific problem can be used, as well as in the nature of the source, i.e. usually this is a data set but can also be the output signal from a sensor, a simulator, etc., and no limitations are posed to the implementation of the DM systems, i.e. they can be either classic DM models or modern ML-based systems as more appropriate. A compact graphical representation of the n potential agents, $A_1, A_2, A_3, \ldots, A_n$, is available in Figure 1 to conceptualise the idea behind the

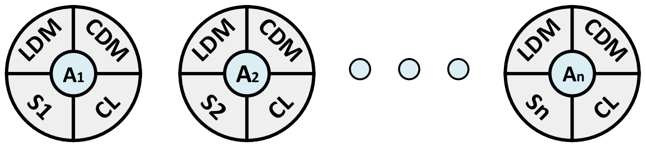


FIGURE 1. **Graphical representation of the agents in the proposed *RDM_{CA}* system.**

proposed framework. It must be noted that S_i $(i = 1, \dots, n)$ refers to the source of information of the i^{th} agent and not necessarily to a i^{th} source of information. According to the specific need, multiple agent can operate on the same source of information, e.g. the same data set S (i.e. $S_1 = S_2 = \cdots = S$). Obviously, all scenarios are possible, i.e. those in which the totality of the agents equipped with diverse local and collaborative DM systems are fed with the same source of information, those in which each agent deals with a different source of information (e.g. this could be the case in which different frequency bands are considered for taking photographs of the same object [5], [6]) and any combination in which some agents work on the same source while other use different types or sensors or look at different sources. Decision made at a local level, i.e. individually by each agent via the implemented LDM system, need to be shared so that each agent is capable of collaboratively make a more robust decision via its CDM system. For this reason, each agent is equipped with a CL which is the software component responsible for sharing measures of support. An overview of the working mechanism of the proposed system is schematised in Figure 2, in which the whole process is depicted in a sequence of five steps. Details on each step are given in the following sections.

A. DATA SOURCE AND ACQUISITION

As previously discussed, raw data to be used by any generic agent can be from different sources or have different formats. Hence, before initiating the DM process, ad-hoc acquisition processes are required to store relevant data to be fed to each i^{th} agent A_i. In this light, the very first step shown in Figure 2 represents the acquisition process to be performed to obtain raw data. This is obviously problem dependent.

In some scenarios, for the sake of robustness, multiple agents might be equipped with heterogeneous feature extractions systems and let operate on the same raw-data. For this reason, even if raw-data comes from a unique sensor, a number of sources S_i equal the number of agents A_i is defined in this step. The reason behind this is that those data will require different pre-processing phases to be usable by the corresponding agent. Therefore, even if they are acquired with the same sensor, each i^{th} agent will have to pre-process those raw-data into a meaningful data set S_i to use it for DM purposes.

B. AGENT CONCEPT ASSOCIATION (ACA)

The variables of the n acquired data sets S_i are then associated to an agent A_i during this process, indicated with the Agent Concept Association (ACA) block in Figure 2, each agent A_i $(i = 1, \dots, n)$ is

- initialised with a number of candidate DM methods (see Section II-C and II-D);
- linked to its S_i data set so that data can be captured and processed;
- prepared to undergo training and return a LDM measure of support (as graphically depicted in Figure 2);
- connected with the other $n - 1$ agents via its CL to exchange measures of support and perform CDM (as graphically depicted in Figure 2).

After being linked to a data set S_i, raw-data might need pre-processing to obtain exploitable information. Hence, while capturing data (see II-A), agents must employ digital filters (to reduce noise) and features extraction techniques that have to be chosen in advance by considering the nature of the available data. Once raw-data are transformed into meaningful features, the LDM process takes place and its output is sent (“Send LDM”) through the CL to be used as measure of support by the other agents. This means that the proposed system must be coordinated synchronously. At this point, all agents are capable of performing CDM by using the exchanged measures of support (“Collaborate DM”). To implement this coordination logic in the proposed system, each agent is executed as a sub-process on the same machine and their tasks are run in parallel simultaneously. Hence, to perform the CDM process, the “fastest” agent has to wait for the “slowest” to send its measure of support. For the sake of clarity, the implementation code for handling communication across agents and their coordination is made available in the repository in [30].

C. LOCAL DECISION-MAKING (LDM)

To have a robust and optimised LDM process, several state-of-the-art DM methods (DMMs) can be tested before choosing the most appropriate one. This process is meant to return a LDM method tailored to the scenario at hand and consists of a trial-and-error process in which the candidate DMMs are selected from the pool “Initial DMMs” to then being trained and, subsequently, assessed by means of the performance metrics explained in Section III. Suggested DM methods to be considered in the pool are

- Rule-based or Fuzzy systems, as those in [31], [32];
- Decision Tree, as those in [33], [34];
- Support Vector Machine (SVM) models, as those in [31], [33], [34];

Neural systems as e.g. [33], [35], can also be used. Amongst them, the neuroevolution approach introduced in [36]–[38], and successfully used in real-world application such as [6], appear to be the most promising choice, as shown in each case study of Section IV, since capable of optimising the structure of neural networks to best perform on the available data set. Obviously, these methods require dividing the processed data sets S_i into a training (S_T) and a validation (S_V) sets, as indicated in Figure 2. These two sets are obtained

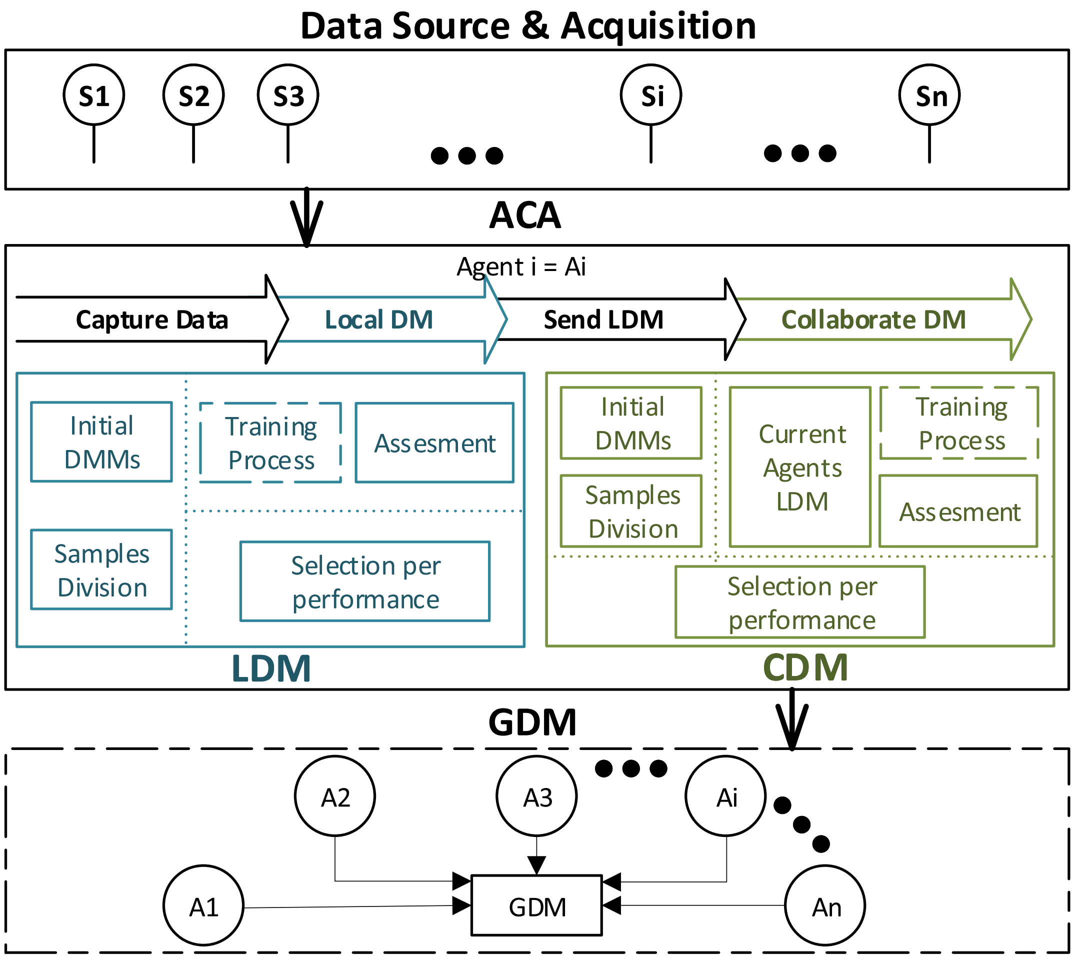


FIGURE 2. **Flow chart of the proposed *RDM_{CA}* framework. Process flow direction is from top to bottom, corresponding to the sections explained in the text: Data sources and acquisition, Agent Process Association(ACA), Local and Collaborative decision-making stages (LDM, CDM), and Global Decision Making (GDM, when applicable, see section II-E.)**

by following the results from [6], which suggest allocating 70% of the available samples to S_T and the remaining 30% to S_V to avoid biases in the learning process. The assessment process can be preformed manually to select the best optimiser, or automated with the neuroevolution approach. The latter makes use of a Genetic Algorithm (GA) [39] to evolve and compare neural networks having different architectures, activation functions, number of layers and nodes. Details of these steps are given in Section IV-C2 and IV-C3.

D. COLLABORATIVE DECISION-MAKING (CDM)

The CDM step is based on the same logic of the previously described LDM process. Hence, potential CDM methods are either be selected from the DMMs pool or evolved, trained, validated and assessed to find the optimal DM system. However, unlike the LDM module, CDM requires two input vectors formed by

- the features extracted and assigned per agent (see section II-B) from the acquired raw-data, as indicated in section II-A, and shown in the Data Source & Acquisition step of Figure 2;
- the decisions support coming from the partner agents and itself.

Every agent knows the last local decisions of the other agents and considers these when executing its collaborative decision-making process. It worth noting then, that the local decision vector (e.g. all β's on the left of Figure 3) is the same for all agents at any given time.

Also in this case, several CDM methods can be taken from the DMMs pool, trained, validated and assessed through the same approach explained in Section II-C. Hence, the available data must be split so that 70% of the samples go to S_T, while the remaining 30% go to S_V, before training the selected DM methods. The one showing the best performance over the validation set S_V is that chosen for the CDM block.

E. GLOBAL DECISION-MAKING (GDM)

Previous studies confirmed that CDM tends to improve upon the accuracy of individual DM processes in MASs [6]. In the generic case where the outcome of all CDMs is not unique or unanimous, the "global" decision is made as shown in the last block of Figure 2, the outputs generated via individuals

CDM systems are aggregated in the so-called Global Decision making (GDM) block. This block is activated when all CDM measures of support, from each agent, are available and communicated through their CLs.

Our GDM strategy for the general case involves deciding at the validation stage which agent has the best performance and use this as the chosen GDM, we can further chose a second best, etc... so that we can guarantee that the system will have a global decision, even if the best collaborative decision is not available, for example due to a fault in the sensor or agent. This will be shown in detail in Section V-D.

III. EVALUATION METRICS

To assess the proposed system we made use of three different evaluation methods capable of fairly comparing all the possible cases resulting from having different DM systems, two or more agents.

The first evaluation metric is the Root Means Square Error as defined in Equation 1,

$$RMSE = \sqrt{\frac{1}{|S|}\sum_{k=1}^{|S|}(x_k - \widehat{x_k})^2}, \tag{1}$$

where x_k is the expected output of the system, $\widehat{x_k}$ is the predicted output, and k is a generic sample from the set of available samples S. The RMSE allows for the comparison of dissimilar DM systems, being normalised by the number samples, and therefore is very established for assessing classification performances. Ideally, a perfect classifier would display a null RMSE. In practical terms, the smaller is the RMSE value, the more reliable is the classification process at hand.

The second evaluation metric is the so-called accuracy, referred to as ACC, which is defined by means of the following ratio

$$\mathrm{ACC} \triangleq \frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{FN}+\mathrm{TN}+\mathrm{FP}}, \tag{2}$$

where the terms in the numerator, i.e. True Positive (TP) and the True Negative (TN), as well as the remaining term in the denominator, i.e. False Positive (FP) and False Negative (FN), are well-known in binary classification and indicate the number of correct predictions for the positive and the negative classes, for the first case, and the number of classifications for the positive and the negative class, in the second case. A schematic description of these terms is given in the so-called confusion matrix represented in Table 1. Therefore, the ACC value is a non-negative scalar in [0, 1] which can be seen as a normalised index of correct classifications, for both the negative and positive classes. Ideally, a perfect classifier has an accuracy equal to 1, which results into a 100% correct classification rate with a null RMSE. Hence, in a real-scenario, the higher the ACC value the more effective is the classifier.

The last evaluation method makes use of the Receiver Operating Characteristic (ROC) curve to calculate the Area Under the Curve (AUC) metric [40]. The ROC curve is obtained by plotting tuples (FPR, TPR), with

TABLE 1. Example of a generic confusion matrix for binary classification.

		Class: positive	Class: negative
Prediction:	positive	TP	FP
	negative	FN	TN

$$\mathrm{FPR} \triangleq \frac{\mathrm{FP}}{\mathrm{FP}+\mathrm{TN}} \tag{3}$$

and

$$\mathrm{TPR} \triangleq \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}, \tag{4}$$

at various threshold settings, i.e. from the minimum to the maximum allowed value according to the boundaries of the output range. Usually, FPR (aka miss rate) values are represented in the x axis while TPR (aka sensitivity, recall or hit rate) values are represented in y axis. Obviously, admissible values for the two rates lies in the [0, 1] range. The corresponding AUC scalar values is somehow able to give indications on the shape of the ROC curve: if higher than 0.5, ROC is concave up and its graph is above the $y = x$ curve. In this configuration one can conclude that the performance of the DM classifier under consideration is clearly superior than a "random guess" whereas a AUC inferior to 0.5 indicates that the considered approach is making a random decision. In this light, a meaningful DM strategy must have an AUC value in the rage [0.5, 1]. While an ideal classifier would obviously display a unitary AUC value, in general terms it can be stated that the higher the its value more robust is the decision made.

The three proposed evaluation metrics are established and represent a standard for benchmarking DM methods [41], [42].

IV. IMPLEMENTING AND TESTING RDM$_{CA}$

In this section we evaluate our model using four different data sets, which will be presented in the following section, IV-A. We start by specifying the agent concept association (section IV-B) and detail the final decision-making system in section IV-C.

A. DATA SOURCES

The data sets used for evaluation are from the KEEL database [43], from which we selected four suitable data sets for supervised binary classification. Brief details of these are:

- The "DB0" data sets by Gorman and Sejnowski [44] contains 208 samples acquired with a sonar while sensing the land in the attempt to discriminate between rocks and military grade mines. Each sample has 60 features obtained by changing the orientation (i.e. angle) of the sonar while acquiring information to detect a given object. This is obviously a binary classification task in which objects belonging to the class (0) are classified as

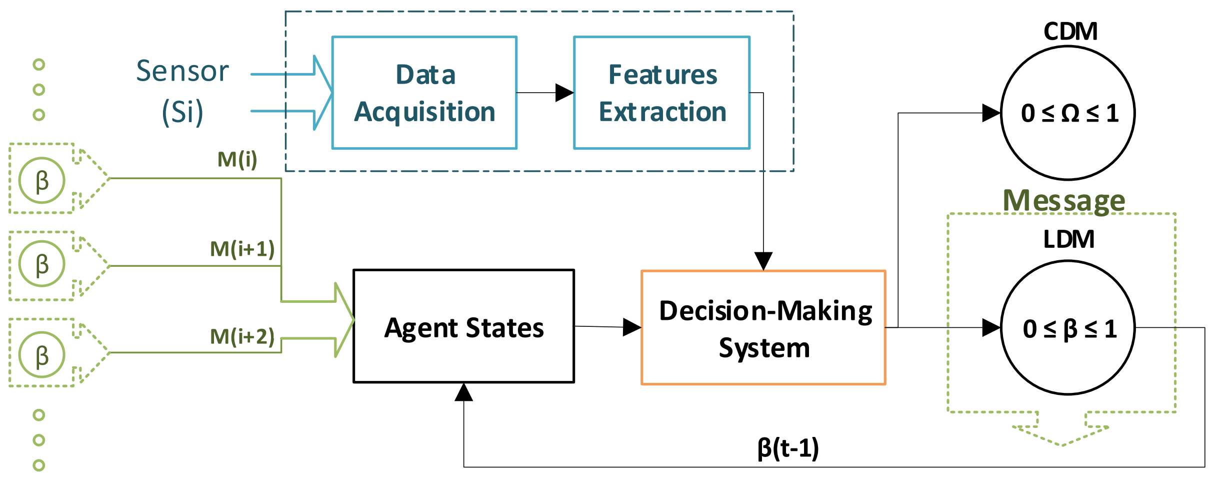


FIGURE 3. RDM_{CA} **algorithm structure per agent on the system. Where Si is the sensor/Data features input and β's are the messages containing the states of each of the other agents in the decision-making system. The outputs at the right of the diagram are the Collaborative Decision and the message from this agent, to the others, containing the Local Decision.**

rock while those belonging to the class (1) are metallic cylinders (i.e. a potential mine).

- The "DB1" data set by Kurgan *et al.* [45] consists of 267 samples acquired through Single Proton Emission Computed Tomography (SPECT), each one having 44 relevant features for cardiac diagnosis. In detail, each image can classified into two classes, (0) and (1), with the first referring to the healthy state while the second to an abnormal state for the heart. These data have been also donated and deposited to be available online from [46].
- The "DB2" breast cancer Wisconsin (diagnostic) data set [47] contains 569 samples obtained through Fine Needle Aspirate (FNA) biopsies to determine malignancy or benignity of breast cancer. Each sample displays 30 usable features computed from the corresponding digitised image of the cells nuclei extracted with the biopsy test. Benign cases are classified in the (0) class while malign cases in the (1) class.
- The last data set, i.e. "DB3", is a synthetic data set produced by the Department of Statistics of the University of California and disseminated through the technical report in [48]. It consists of 7400 samples having 20 features each. Every sample refer to a multivariate normal distribution to be classified into two possible classes. Class (0) contains those distributions displaying null mean values and co-variance matrix equal to $4 \cdot \mathbf{I}$ (where $\mathbf{I}$ is the identity matrix) while class (1) those having a mean values equal to $2/sqrt(20)$ co-variance matrix equal to $\mathbf{I}$.

Hence, three out the four selected data sets belong to real-world application domains. Even though we focus our attention on real world applications, the last (synthetic) data set was selected to test our proposed system with a relatively large data set as well.

B. AGENT CONCEPT ASSOCIATION

The second stage of the software model includes the architecture of the agents in RDM_{CA} system. The multi-agent system should have at least two agents within this architecture, and every agent should have the states information of its peers at any given time. Figure 3 illustrates as a block diagram the processes executed per agent.

The agent has two sets of inputs, the data acquisition and the messages with the local decisions from the other agents. The data acquisition block controls the timing of the sensors samples and filters this data. Then, the features block extracts the relevant characteristics and send them to the decision-making system. Every type of sensor data requires a specific features extraction process. Finally, the decision-making system (DMS) processes the extracted features and the states of the other agents to determine a local and a collaborative decision.

When a prepossessed data-set of features (rather than raw sensor data) is available for the RDM_{CA} system, the data acquisition and features extraction blocks are not required. The features come directly from the databases and can be processed directly by the decision-making block. Moreover, here a specific local decision-making method can be chosen and fixed; and use a variety of methods to explore for the collaborative decision making and evaluate the advantages of the collaborative process.

To test our model, we have six RDM_{CA} tests architectures per data set. First, there is an architecture with a single agent ($n = 1$), where the output is the single (local) decision. The other architectures use multiple agents (n), from two to six agents. Each agent with a ffANN as local decision-making method to obtain a local decision (β) and a set of decision-making methods to test different alternatives for the collaborative decision (Ω).

C. DECISION-MAKING SYSTEM

1) ARCHITECTURE DESCRIPTION

The detail of the decision-making system per agent shows the internal configuration of the DMS block (Figure 4). It has a dedicated decision-making method each to obtain a local and a collaborative decision. The local decision-making method only requires features; in contrast, the collaborative decision-making uses the local decision from all the agents, alongside the features of the sensor data.

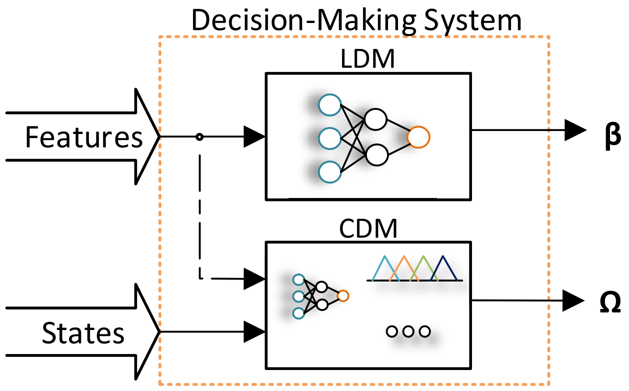


FIGURE 4. Decision-making system detail on a typical agent as in Figure 3. The features (extracted form the sensor or obtained from data) feed both decision making processes; the states of the other agents received as β's in Figure 3, feeds the collaborative decision making block as well.

In general, each DMS in the system has two outputs per agent, the local (β) and collaborative (Ω) decisions. These decisions take a value in a range (e.g.[0,1]). When the application has a binary output, a threshold is used.

Our DMS per agent has two possible architectures depending on the inputs used by CDM. The first case is when the CDM system uses only the LDM from all agents. The second case is when the CDM uses the LDM from all agents and the features from its sensor data. When the agents use the first architecture, the CDM system is identical for all. In contrast, for the second case, each agent has a specific CDM system as it uses specific information according to its own sensor (or data source).

To test the performance of the RDM_{CA} model, we define that each agent will have the same decision-making method for the local decision (a feed-forward artificial neural network ffANN structured using NEAT). On the other hand, we will test a number of collaborative decision-making methods: a choice of ''intelligent'' ones (a ffANN and a fuzzy decision support system FDSS), and a variety of aggregation and statistical methods (mean value, median value, maximum value, and a voting mode).

We selected the FDSS, the arithmetic mean value, the median value, the maximum value, and the voting mode to work under the first case of DMS architecture, when the CDM system has only the LDM from all the agents. For the case where CDM uses its sensor features as well, we will test the ffANN, in which case the GDM block will be required to become explicit as will be discussed in section V-D.

2) DECISION-MAKING METHODS

In the previous section we described the choices of decision-making methods for RDM_{CA} model to select the best performance from a tuple of LDM and CDM per agent. In this section we will describe in detail each of these.

For the collaborative decision we selected a ffANN designed with NEAT (N_{CDM}). NEAT (neuroevolution of augmenting topologies) is a state-of-the-art neuroevolution algorithm that evolves both the weights and topology of the network, allowing the behaviour of evolved neural networks to become increasingly sophisticated over generations [20]. To implement and train the neural networks with neuroevolution [36], the algorithm designed in [37] was used, from the Python library in [49].

The genome in the genetic algorithm required by NEAT includes a list of connection genes, each gene specifies the connection among two node genes, the in-node, the out-node, the weight of the connection $W_{i,j}$, the activation function of the node $f\ (x_i)$, an enable bit of the connection and an innovation number, which allows finding corresponding genes during crossover. Mutation in NEAT adjust connection weights, network structures and activation functions P_{NMAF}. Structural mutations, which expand the network, occur in two ways, adding a new connection between nodes P_{NMC+} or adding a node P_{NMN+} with initial connections. There are as well structural mutations which reduce the network by removing an existing connection P_{NMC-} or an existing node P_{NMN-} [36].

Some more elaborate versions of the library allow the original NEAT algorithm to increase the performance for specific cases. However, the data sets selected in the current project have few features in contrast with applications that require *hyperNEAT* and our system adjustment is offline. Then, the variants of NEAT such as *rtNEAT* are not required for our case, and we will use the standard library.

The FDSS model, similarly to NEAT, is an integration of two methods, genetic algorithms, and fuzzy logic. Baron presents in [50] the general structure of the FDSS algorithm. However, we implement a new operator of crossover to increase the variety of individuals.

Finally, we have the aggregation methods. The arithmetic mean value (P), the median value (D), the maximum value (M) and the voting method (V). This last one consists of collecting the local decisions from all the agents and applying a voting process, the CDM outcome depends on the quorum (e.g., if RDM_{CA} system has three agents, a quorum means that two or more agents have the same decision).

In order to easily refer to each of the decision-making methods presented in this section from here on, we have used the abbreviations shown in Table 2 where (N_{LDM}) and (N_{CDM}) are the local and collaborative decision making NEAT methods.

TABLE 2. Abbreviations list for the decision-making methods.

Method	N_{LDM}	N_{CDM}	FDSS	P	D	M	V
Abbreviation	L	N	F	Ω_P	Ω_D	Ω_M	Ω_V

3) TRAINING CONFIGURATION

In this section we will describe in detail the training of the decision making methods, namely the ffANN for the LDM, and the ffANN and FDSS, as the chosen intelligent decision-making methods for the CDM.

The training of each ffANN has two phases, the configuration and tuning steps. The initial phase of the training (configuration step) has the purpose of comparing and selecting the best configuration to improve evolution. In this phase, we defined a population of 600 individuals for NEAT to evolve during 70 epochs. Also, there are 216 tests of training configuration of NEAT per every network. The configuration parameters are the number of the hidden neurons at the beginning of the training process (0,1 or 2), the initial connections among the neurons (see Table 3), and the probabilities of the mutation operators (see Table 4).

TABLE 3. Options given by NEAT algorithm for initial connections among inputs, hidden neurons and outputs.

CN	**Type of initial connection**	**Description**
0	unconnected	None connection
1	full no direct	There are connections among inputs and hidden neurons, and hidden neurons and outputs.
2	full direct	There are connections among inputs and hidden neurons, and inputs with outputs.

TABLE 4. Probability distributions among the mutation operators of the networks structure of the ffANN used by the genetic algorithm of NEAT.

T_P	0	1	2	3	4	5
P_{NMC+}	0.1	0.4	0.7	0.3	0.2	0.1
P_{NMC-}	0.3	0.2	0.1	0.1	0.4	0.7
P_{NMN+}	0.3	0.2	0.1	0.3	0.2	0.1
P_{NMN-}	0.3	0.2	0.1	0.3	0.2	0.1

T_P	6	7	8	9	10	11
P_{NMC+}	0.3	0.2	0.1	0.3	0.2	0.1
P_{NMC-}	0.3	0.2	0.1	0.3	0.2	0.1
P_{NMN+}	0.1	0.4	0.7	0.3	0.2	0.1
P_{NMN-}	0.3	0.2	0.1	0.1	0.4	0.7

T_P	12	13	14	15	16	17
P_{NMC+}	0.1	0.1	0.1	0.4	0.4	0.4
P_{NMC-}	0.1	0.4	0.4	0.1	0.1	0.4
P_{NMN+}	0.4	0.1	0.4	0.1	0.4	0.1
P_{NMN-}	0.4	0.4	0.1	0.4	0.1	0.1

T_P	18	19	20	21	22	23
P_{NMC+}	0.2	0.2	0.2	0.3	0.3	0.3
P_{NMC-}	0.2	0.3	0.3	0.2	0.2	0.3
P_{NMN+}	0.3	0.2	0.3	0.2	0.3	0.2
P_{NMN-}	0.3	0.3	0.3	0.3	0.2	0.2

The second phase of the training process (tuning step) takes the best set of parameter configuration in terms of evolution performance from the previous step. The purpose of this stage is to attain the best network using the same training parameters. We run ten independent training process with random seeds. For each process, we defined a population of 600 individuals which train during 100 epochs. The chosen ffANN of the tuning step is that which performs the best from all training cases.

TABLE 5. Summary of the best ffANN training configurations per data set and per test (from one to six agents in the system). Nomenclature used in the table: *n*: agents in the system, *ID*: identification of the agent in the system, *TP*: mutation parameters (see Table 4), *NH*: initial neurons in the hidden layer and *CN*: initial connection among neurons and inputs (see Table 3).

	(TP, NH, CN)			
$N_{n\&ID}$	**DB0**	**DB1**	**DB2**	**DB3**
N_{11}	(6, 2, 2)	(9, 0, 2)	(3, 2, 0)	(14, 1, 2)
N_{21}	(3, 1, 0)	(6, 2, 1)	(0, 2, 2)	(6, 1, 1)
N_{22}	(3, 0, 2)	(5, 1, 2)	(21, 1, 2)	(15, 2, 1)
N_{31}	(10, 0, 0)	(19, 1, 1)	(13, 2, 2)	(0, 1, 1)
N_{32}	(23, 2, 1)	(4, 2, 1)	(8, 0, 0)	(1, 2, 1)
N_{33}	(10, 2, 0)	(15, 0, 2)	(12, 0, 2)	(2, 2, 1)
N_{41}	(0, 2, 0)	(20, 2, 1)	(23, 2, 2)	(1, 2, 1)
N_{42}	(6, 2, 1)	(10, 1, 0)	(8, 1, 0)	(12, 1, 0)
N_{43}	(11, 2, 2)	(3, 1, 1)	(3, 2, 0)	(7, 1, 1)
N_{44}	(1, 1, 0)	(5, 2, 2)	(2, 0, 2)	(13, 2, 1)
N_{51}	(7, 1, 0)	(9, 1, 1)	(23, 0, 0)	(12, 2, 1)
N_{52}	(7, 0, 0)	(20, 1, 1)	(9, 0, 0)	(6, 2, 1)
N_{53}	(7, 2, 1)	(19, 2, 1)	(3, 0, 1)	(2, 1, 1)
N_{54}	(12, 2, 1)	(4, 2, 1)	(10, 1, 0)	(2, 2, 1)
N_{55}	(22, 2, 1)	(4, 2, 1)	(6, 0, 2)	(16, 2, 1)
N_{61}	(8, 0, 0)	(16, 1, 1)	(14, 1, 2)	(14, 0, 1)
N_{62}	(23, 1, 0)	(5, 2, 0)	(8, 0, 0)	(22, 2, 1)
N_{63}	(1, 1, 1)	(23, 2, 0)	(18, 2, 0)	(12, 1, 1)
N_{64}	(15, 2, 1)	(3, 2, 1)	(3, 1, 1)	(6, 1, 1)
N_{65}	(16, 1, 1)	(16, 2, 1)	(17, 0, 2)	(7, 1, 0)
N_{66}	(0, 1, 1)	(22, 1, 1)	(10, 1, 0)	(23, 2, 1)

Table 5 shows the best performance specifications of the training process per data set and per number of agents, in terms of initial connection (see Table 3), mutation configuration (see Table 4) and number of neurons in the hidden layer (NH). This table shows that the configuration parameters for each agent's training algorithm is different and specific to the task, data set, the particular data features for the agent and number of agents in the system.

Another configuration parameter that the NEAT algorithm allows to configure is the probability of the mutation operation (P_{NMAF}) to change the activation function of the neuron. The genetic algorithm searches the most accurate activation function between a sigmoid, an absolute value, or a Gaussian function. We configure a value of $P_{NMAF} = 0.5$ for both cases L and N.

In general, Table 5 shows that for the data sets DB0, DB1, and DB3, evolution worked better with two initial conditions: The full no direct connection among neurons, meaning that the best initial network has two links at the start between inputs and hidden neurons, and hidden neurons and outputs. And, two hidden neurons (NH) at the beginning. For DB3, the networks perform better by starting with no hidden neurons and no initial connections.

Both initial configurations of the networks have their advantages. The first one uses a classical net to improve its performance; this implies less time in the construction of the net and more preference in adjusting the weights of the connections in the net. The second one searches new types of relationships and number of hidden neurons; it means that the net construction will differ from the classical net obtaining specific output functions.

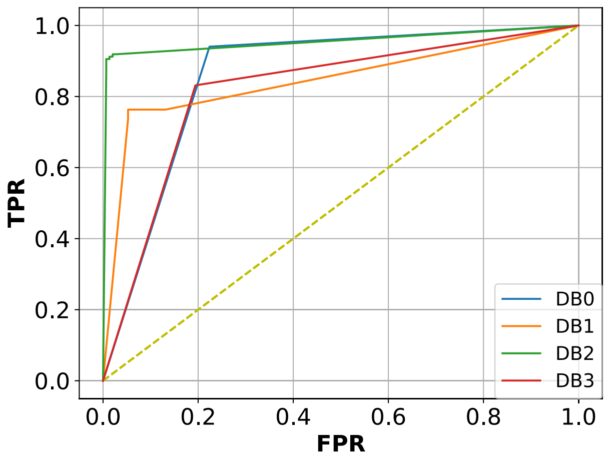
On the other hand, there in not a preferred case of the mutation configuration (TP). The selection of this configuration depends on the percentage distribution that increases the training performance; it means that each agent is free to elect its percentage distribution. An advantage of this behaviour is that each agent specialises in the training process, depending on the input data.

Finally, the other intelligent decision-making method that has a training stage is FDSS model. This method has a genetic algorithm with the configuration parameters shown in Table 6. In contrast to NEAT, FDSS is only used by the collaborative decision-making system per agent. Each FDSS has the same parameters of configuration except the number of inputs of the system. This number depends on the number of agents in the multi-agent system (see column N_I).

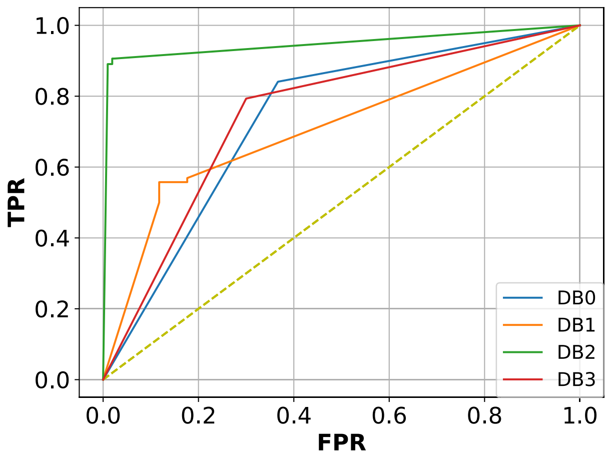
TABLE 6. Summary of the FDSS training configurations for all data sets. Where: E number of epochs per training, N_P size of the population, N_I inputs of the fuzzy system, P_{FDC} crossover probability, P_{FDM} mutation probability, P_{Fm} bit mutation probability, and P_{FC_n} crossover probability of the n crossover operator.

	E	N_P	N_I	P_{FDC}	P_{FDM}
$Value$	80	50	2…6	0.7	0.2

	P_{Fm}	P_{FC_1}	P_{FC_2}	P_{FC_3}	P_{FC_4}
$Value$	0.1	0.2	0.2	0.3	0.3



(a) Training results



(b) Validation results

FIGURE 5. ROC graphs for the performance of a single agent system per database.

V. RESULTS AND DISCUSSION

In this section we present the results obtained over the four data sets described in Section IV-A, namely DB0, DB1, DB2 and DB3.

Numerical results are displayed in comprehensive tables but also accompanied by graphs depicting ROC curves as explained in Section III. However, given that most ROC curves displayed similar trends across the four data sets (which confirm that the proposed framework is robust in handling different scenarios regardless of their nature), in this manuscript we included only those related to DB1 to avoid an unnecessary long gallery of similar images. For a full gallery, please visit our online repository at [51].

A first analysis is performed by considering the LDM system when the system has a single agent. This is reported in Figure 5 where the ROC curves of each data set are plotted with a different colour.

We perform six tests associated with the number of agents in the system. The first test with a single agent as described above; it means that the system uses all the features, and there is a single output given by the local decision. The other five tests are for decision-making systems with a multi-agent system from two to six agents; these tests require a collaborative decision-making process per agent and the sharing of the features vector among the number of agents.

A. RESULTS DESCRIPTION

The results are presented in Tables 7, 8, 9 and 10 that show the metrics results (ACC, RMSE and AUC) of the training and validation processes. These tables present the best two cases per the number of agents (n) vs. the single agent test ($n = 1$ and L). The selected cases are the best related to the accuracy metric at a fixed threshold value of 0.5 in the outcome range of [0, 1]; it means that a positive outcome of the decision-making occurs when β or $\Omega \geq 0.5$.

To refer a specific result inside these tables, we use this notation: $DM_{n\&ID}$ relating the columns of the table, e.g. the best AUC value of all data sets can be found in Table 9 for

TABLE 7. Metrics of the best training and validation results for the DB0 data set. This table contents the best two cooperative decision-making methods accuracy performance per number of agents (*n*) vs. the performance of the single agent case. *A* indicates that all the agents in the *RDM_{CA}* system have the same collaborative decision-making method.

	Training					Validation				
n	DM	ID	ACC	RMSE	AUC	DM	ID	ACC	RMSE	AUC
1	L	-	0.8582	0.3766	0.8582	L	-	**0.7162**	0.5327	0.7189
2	N	1	**0.8881**	0.3024	0.9358	N	2	**0.7568**	0.4932	**0.7371**
2	N	2	0.8657	0.3665	0.8646	N	1	0.6892	**0.4660**	0.6894
3	N	1	0.8284	0.3740	0.8672	Ω_M	A	0.6892	0.4752	0.6500
3	N	2	0.8209	0.3730	0.8797	F	A	0.6892	0.4893	0.7258
4	N	2	**0.8881**	**0.2984**	**0.9550**	N	3	0.6757	0.5695	0.6902
4	N	4	**0.8881**	**0.2987**	**0.9552**	Ω_D	A	0.6622	0.4795	0.6652
5	N	2	**0.8731**	0.3063	0.9345	Ω_D	A	0.7027	**0.4686**	0.7318
5	N	3	**0.8731**	0.3181	0.9240	Ω_P	A	0.7027	0.4751	0.7318
6	N	4	0.8433	0.3314	0.9263	N	1	0.7027	0.4769	**0.7424**
6	N	5	0.8433	0.3447	0.9138	F	A	0.7027	0.4987	0.7144

TABLE 8. Metrics of the best training and validation results for the DB1 data set. This table contents the best two cooperative decision-making methods accuracy performance per number of agents (*n*) vs. the performance of the single agent case. *A* indicates that all the agents in the *RDM_{CA}* system have the same collaborative decision-making method.

	Training					Validation				
n	DM	ID	ACC	RMSE	AUC	DM	ID	ACC	RMSE	AUC
1	L	-	0.8421	0.3974	0.8452	L	-	0.5550	0.6671	0.7082
2	N	1	**0.9079**	0.2925	0.9138	N	1	0.6545	0.5076	0.6858
2	N	2	0.8684	0.3352	0.8805	Ω_D	A	0.6126	0.4147	0.6739
3	N	2	**0.8947**	**0.2901**	0.9186	Ω_D	A	0.7225	0.3722	0.7050
3	N	1	0.8684	**0.2900**	0.9127	N	1	0.7068	0.5566	0.7537
4	N	4	0.8816	0.3236	**0.9474**	F	A	**0.9110**	0.4575	0.7684
4	N	1	0.8684	0.3076	**0.9443**	Ω_D	A	**0.8482**	**0.3131**	0.7441
5	N	5	**0.9079**	0.2989	0.9307	Ω_D	A	0.7435	0.3210	**0.7880**
5	N	1	0.8553	0.3381	0.9193	F	A	0.7382	0.4972	0.7693
6	N	4	**0.8947**	0.3195	0.9086	Ω_D	A	0.8220	**0.3107**	0.7333
6	N	3	**0.8947**	0.3199	0.9100	F	A	0.7225	0.4988	**0.7831**

TABLE 9. Metrics of the best training and validation results for the DB2 data set. This table contents the best two cooperative decision-making methods accuracy performance per number of agents (*n*) vs. the performance of the single agent case. *A* indicates that all the agents in the *RDM_{CA}* system have the same collaborative decision-making method.

	Training					Validation				
n	DM	ID	ACC	RMSE	AUC	DM	ID	ACC	RMSE	AUC
1	L	-	0.9426	0.2396	0.9550	L	-	0.9414	0.2442	0.9699
2	N	2	0.9628	0.1928	0.9624	Ω_V	A	0.9377	0.2495	0.9268
2	N	1	0.9561	0.2096	0.9555	F	A	0.9377	0.2630	0.9447
3	N	3	0.9696	0.1772	0.9886	N	2	0.9414	0.2421	0.9449
3	N	1	0.9527	0.1948	0.9839	N	3	0.9451	0.2240	0.9806
4	N	3	**0.9797**	**0.1298**	**0.9974**	Ω_D	A	0.9414	0.2152	0.9802
4	N	2	0.9662	0.1699	0.9924	Ω_P	A	0.9414	0.2273	0.9768
5	N	4	0.9730	0.1552	0.9938	N	1	**0.9560**	0.1988	**0.9841**
5	N	5	0.9730	0.1719	0.9933	N	4	0.9451	0.2007	0.9807
6	N	6	**0.9764**	0.1571	0.9922	N	5	**0.9670**	**0.1816**	0.9474
6	N	1	0.9730	**0.1438**	**0.9954**	N	1	**0.9560**	**0.1977**	**0.9862**

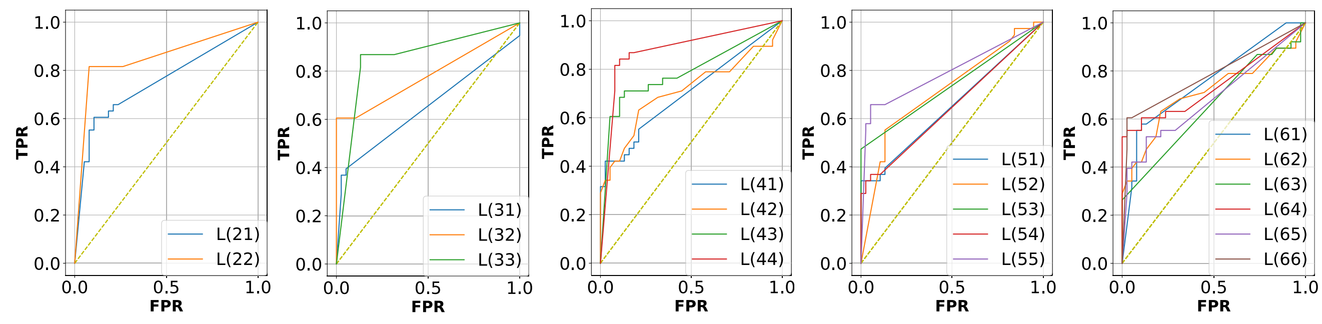
N_{43} in the training section. It means that the best AUC measure belongs to a system with four agents with collaborative decision-making using the NEAT method and from an agent identified as number three.

We also highlight the most relevant results per evaluation metric using bold black text for the best result and bold blue text for the second one. Note that the column *ID* only applies to the models with ffANN since it belongs to the second architecture type of the DMS. A general observation from the information shown in these tables is the increment on the performance of the outcome decision from the collaborative decision-making methods as compared to the single agent. The single agent has the worst accuracy performance for most of the cases.

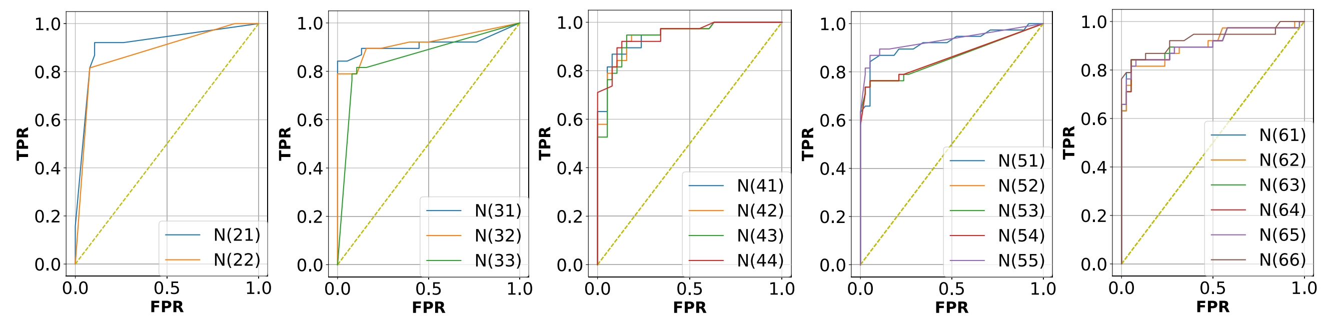
On the other hand, Figures 6 and 7 show the performance of the decision-making methods with multiple agents for data set DB1. They have the result of each of the processes, training, and validation, respectively. Each figure has three rows and five columns, every column corresponds to the number of agents in the *RDM_{CA}* test system (from two to six),

TABLE 10. Metrics of the best training and validation results for the DB3 data set. This table contents the best two cooperative decision-making methods accuracy performance per number of agents (*n*) vs. the performance of the single agent case. *A* indicates that all the agents in the RDM_{CA} system have the same collaborative decision-making method.

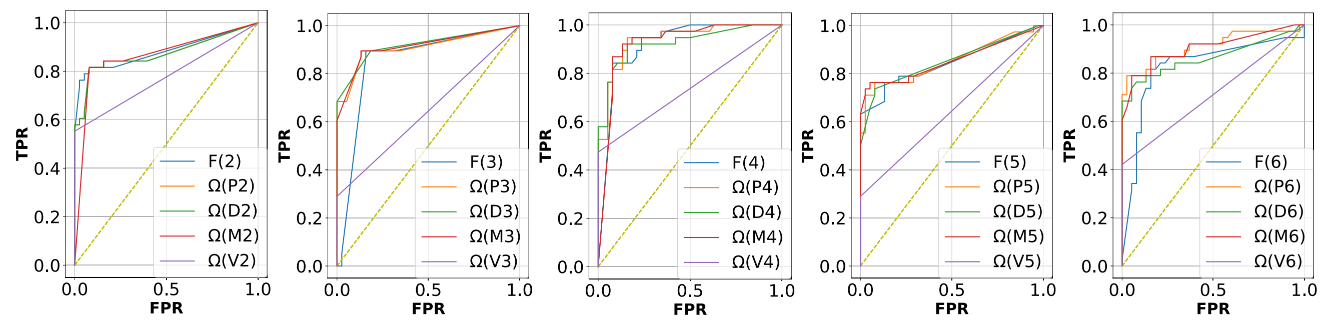
	Training					Validation				
n	DM	ID	ACC	RMSE	AUC	DM	ID	ACC	RMSE	AUC
1	L	-	**0.8186**	0.4259	0.8186	L	-	0.7467	0.5033	0.7467
2	N	1	0.7971	0.3887	0.8390	N	2	0.7300	0.4380	0.7706
2	N	2	0.7971	0.3887	0.8390	N	1	0.7300	0.4391	0.7706
3	N	3	0.7829	0.3820	0.8593	N	3	**0.7700**	**0.3927**	**0.8431**
3	N	2	0.7829	0.3822	0.8576	N	2	**0.7700**	**0.3930**	**0.8398**
4	N	4	0.7757	0.3910	0.8559	N	4	0.7400	0.4144	0.8245
4	Ω_P	A	0.7757	0.4455	0.8559	N	2	0.7400	0.4204	0.8071
5	N	4	**0.8271**	**0.3580**	**0.8992**	N	5	0.7600	0.4147	0.8323
5	N	3	**0.8271**	**0.3584**	**0.9007**	N	4	0.7567	0.4150	0.8306
6	N	6	0.8029	0.3794	0.8792	Ω_P	A	**0.7633**	0.4617	0.8260
6	N	2	0.8014	0.3794	0.8824	N	6	0.7533	0.4323	0.8162



(a) N_{LDM}



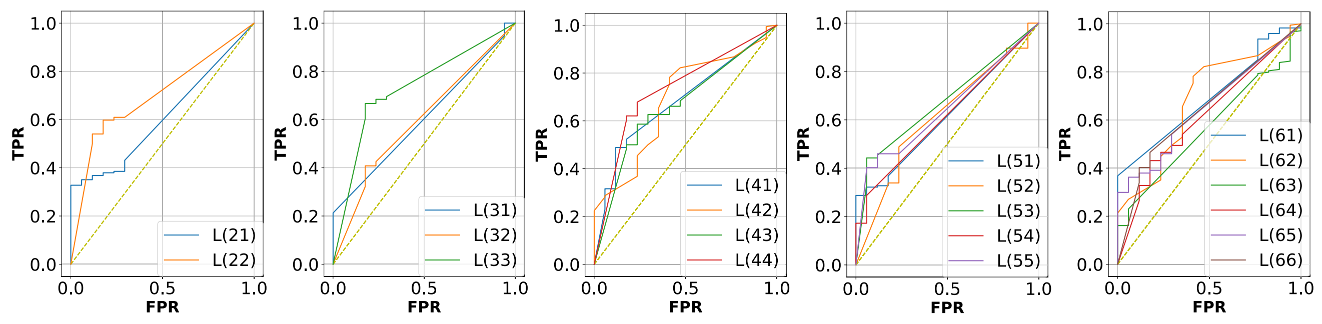
(b) N_{CDM}



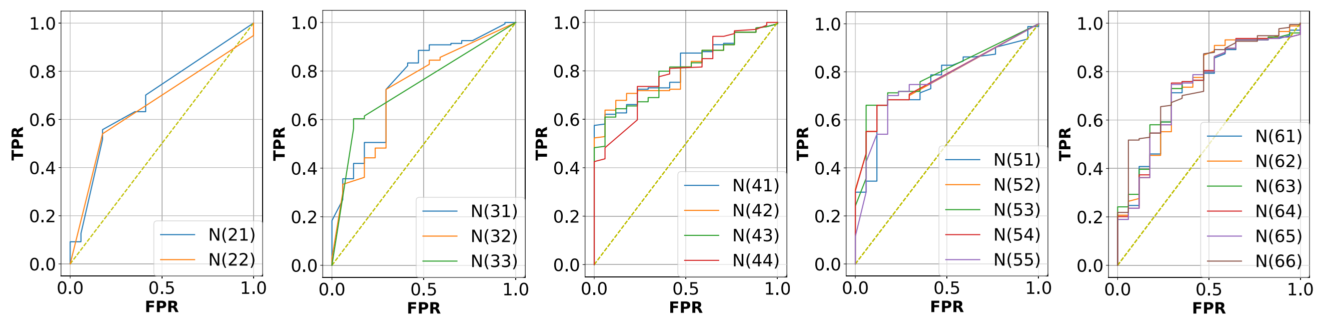
(c) F and $\Omega_{P,D,M,V}$

FIGURE 6. ROC graphs of DB1 data set from systems with multiple agents on the training results. Each column corresponds to the number of agents in the multi-agent system (2, 3, 4, 5, and 6 respectively), and every row presents different decision-making systems (Top row: LDM only per agent, Middle row: NEAT per agent, and Bottom row: global FDSS and statistical methods.)

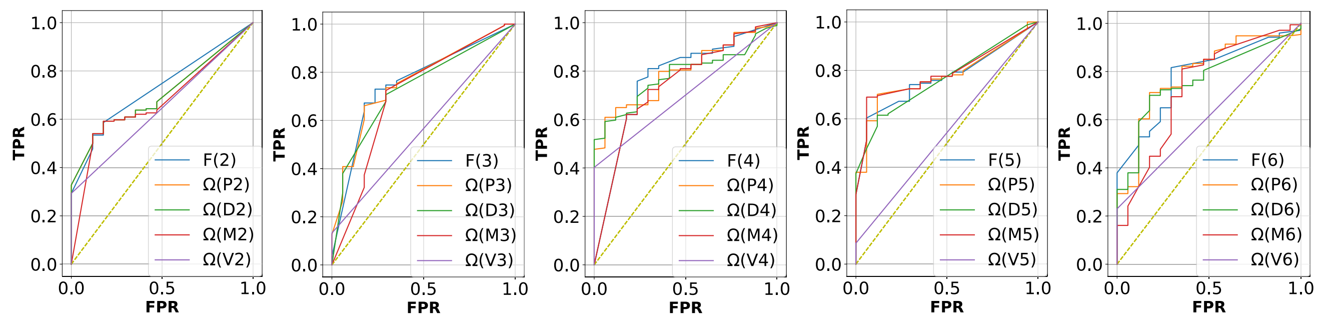
and the rows are the decision-making methods. The top row has the results of the local decision-making methods (L), the middle row has the results of the collaborative ffANN (N), and the bottom row has the results of the FDSS, the statistical methods (i.e. mean, median and maximum value), and the voting method.



(a) N_{LDM}



(b) N_{CDM}



(c) F and $\Omega_{P,D,M,V}$

FIGURE 7. ROC graphs of DB1 data set from systems with multiple agents on the validation results. Each column corresponds to the number of agents in the multi-agent system (2, 3, 4, 5, and 6 respectively), and each row presents different decision-making systems (Top row: LDM only per agent, Middle row: NEAT per agent, and Bottom row: global FDSS and statistical methods.)

The ROC curves of the collaborative decision-making methods in the figures (middle and bottom rows) have an increment of the AUC value, in contrast to the local ones. This can be seen in the graphs which show an increment in the shape concave down to store additional area. This implies that the outcome from the collaborative methods contributes to developing a higher level of confidence. Moreover, collaborative ROC curves can be compare with the single agent case presented in Figure 5, showing an increment in the outcome decision performance.

On the other hand, the ROC curves of the local decision-making methods (first row in Figures 6 and 7) show a reduction of the AUC as the number of agents increases. This reduction occurs because each agent will have fewer features or less information about the observation when there are more agents in the system (see Table 11). Despite this, the curves of the collaborative decision-making methods maintained or increase the AUC as the number of agents increase, and can outperform the best LDM of only 2 agents.

TABLE 11. Summary of the features available for each agent over each data set evaluated. Where *n* is the number of agents in the system.

	n					
	1	2	3	4	5	6
DB0	60	30	20	15	12	10
DB1	44	22	14	11	8	7
DB2	30	15	10	7	6	5
DB3	20	10	6	5	4	3

B. TRAINING STAGE

The best training results per number of agents presented on the left side of Tables 7, 8, 9 and 10 for each of the data sets. Although in the training results we have that the best performance is always with the ffANN, the main outcome worth noting is that CDM aids performance. This can be seen for example in Table 7 where the ACC for agents N_{21}, N_{42} and N_{44} is similar; this despite agents N_{42} and N_{44} having only fifteen features each, while there are thirty features available for agent N_{21} (refer to Table 11 for the number of features per agent). Moreover, the CDM results in the RMSE and AUC being better for the larger number of agents.

C. VALIDATION STAGE

For the validation shown on the right side of Tables 7, 8, 9 and 10 we can see that the best collaborative decision-making method performance could vary among data sets. For example for DB1 is the FDSS rather than ffANN, as is for the others.

D. GENERAL DISCUSSION

The results show that overall the collaborative DM in our RDM_{CA} model results in an increase of the performance in all the metrics (ACC, RMSE and AUC); as seen consistently in all data sets except DB0 (probably because discerning between rock and metal is a relatively simple task). Looking both at the Tables (8, 9 and 10) and Figures 6 and 7 we can see that an increase in the number of agents in the system results in an increase in the performance of CDM.

Relating the CDM methods, the results show as well that using an intelligent method provides a higher performance than any of aggregation methods. This is clear from both the training and validation results where either ffANN or FDSS are the ones providing the best performance rather than any of the other alternatives.

Relating the Global decision making (GDM) as discussed in Section II-E we can see the two different options in our results. Looking at Table 8 we can see that for the case of 4 agents, the best performance was achieved using the FDSS method, therefore we automatically have a global decision from the CDM output. On the other hand, looking at Table 9 for the case of 6 agents, we have that agent N_{65} has the best performance using NEAT, followed by agent N_{61} using NEAT as well; therefore in this case we need to use an explicit GDM step, where the first option would be to choose the CDM from agent N_{65} as the GDM, or if this agent has a fault, then the CDM of agent N_{61}. With these 2 examples we illustrate both possible cases to arrive to a GDM.

VI. CONCLUSION

This paper presents our novel contribution for a multi-agent framework that enhances the decision making capabilities of an agent based classifier. Our results validate the RDM_{CA} framework and shows its performances over four different data sets. Obtained results confirms our assumptions that collaborative decision making systems achieve better accuracy over real-worlds scenarios subject to uncertainty. Indeed, the local decision making process performed by each agent individually never outperforms the final decision making process in which individual contributions, i.e. measures of support, are aggregated and considered simultaneously. In this light, the presented multi-agent decision-making framework has proven to be suitable for dealing with several real-world applications as can "filter out" uncertainty after every decision-making step.

Furthermore, we presented multiple implementations of our novel decision-making framework and performed a thorough comparative analysis among the produced variants. Each variant is based on a different combination of computational intelligence methods for each decision-making step (i.e. LDM and CDM) thus resulting into a hybrid and highly interconnected system. In support of our previous observation, regardless of the implementation, all variants displayed the same behaviour in which the accuracy obtained with LDM gets significantly improved via CDM. Furthermore, we noticed that the number of employed agent can make the difference in achieving good results. Generally speaking, we do not recommend using our proposed RDM_{CA} with only one agent as results suggest that better performances are obtained with at least two agents. Even though systems with redundant agents should be avoided too, multi-agent systems with less than two/three agents lose the benefits of the collaborative action. Obviously, the optimal number of agents can not be indicate "a priori", but should be found empirically in a preliminary phase.

Finally, among the methods used for implementing the CDM, we noticed that "voting" system was the one leading to the lowest performance, while the best results are always obtained through an intelligent decision-making method (i.e. FDSS or ffANN). Further inquiring into this, we can conclude that as the voting method considers each agent's decision as having the same "confidence", this is not always a good choice. The result then show that an intelligent decision-making technique or an asymmetric distribution, such as the work of Yu [52], of the decision weight in the final decision can help to decrease the uncertainty. In our case the FDSS can enhance the global decision by collating intelligently the LDMs while the NEAT enhances each CDM and we can choose the agent which has the highest performance to chose its CDM as the GDM.

Further work will look at generalising this RDM_{CA} architecture to work as well when the agents execute asynchronously which can enable a larger variety of sensors and LDM processes to be used, even if their corresponding sampling and executing times are very different and not suitable to be synchronised or for all agents to wait till all have completed an execution cycle.

REFERENCES

[1] F. Caraffini and M. Gongora, "Collaborative methodology for enhancing sustainability in rural communities and the use of land," De Montfort Univ., Leicester, U.K., Tech. Rep., 2019, doi: 10.21253/DMU.8483960.

[2] J. Colorado, C. Devia, M. Perez, I. Mondragon, D. Mendez, and C. Parra, "Low-altitude autonomous drone navigation for landmine detection purposes," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, Jun. 2017, pp. 540–546.

[3] J. Colorado, M. Perez, I. Mondragon, D. Mendez, C. Parra, C. Devia, J. Martinez-Moritz, and L. Neira, "An integrated aerial system for landmine detection: SDR-based ground penetrating radar onboard an autonomous drone," *Adv. Robot.*, vol. 31, no. 15, pp. 791–808, Aug. 2017.

[4] B. Neupane, T. Horanont, and N. D. Hung, "Deep learning based banana plant detection and counting using high-resolution red-green-blue (RGB) images collected from unmanned aerial vehicle (UAV)," *PLoS ONE*, vol. 14, no. 10, Oct. 2019, Art. no. e0223906.

[5] A. Pena, I. Bonet, D. Manzur, M. Gongora, and F. Caraffini, "Validation of convolutional layers in deep learning models to identify patterns in multispectral images," in *Proc. 14th Iberian Conf. Inf. Syst. Technol. (CISTI)*, Jun. 2019, pp. 1–6.

[6] J. Florez-Lozano, F. Caraffini, C. Parra, and M. Gongora, "Training data set assessment for decision-making in a multiagent landmine detection platform," in *Proc. IEEE World Congr. Comput. Intell.*, Jul. 2020, pp. 1–8.

[7] F. Castanedo, "A review of data fusion techniques," *Sci. World J.*, vol. 2013, pp. 1–19, Oct. 2013. [Online]. Available: http://www.hindawi.com/journals/tswj/2013/704504/

[8] L. Nuzzo, G. Alli, R. Guidi, N. Cortesi, A. Sarri, and G. Manacorda, "A new densely-sampled ground penetrating radar array for landmine detection," in *Proc. 15th Int. Conf. Ground Penetrating Radar*, Jun. 2014, pp. 969–974.

[9] C. M. de Farias, L. Pirmez, G. Fortino, and A. Guerrieri, "A multi-sensor data fusion technique using data correlations among multiple applications," *Future Gener. Comput. Syst.*, vol. 92, pp. 109–118, Mar. 2019.

[10] N. Semeykin, V. Pomozov, and V. Monahov, "Integrated multi-channel unit for humanitarian mine-cleaning operations," in *Proc. 15th Int. Conf. Ground Penetrating Radar*, Jun. 2014, pp. 993–996.

[11] M. Muzammal, R. Talat, A. H. Sodhro, and S. Pirbhulal, "A multi-sensor data fusion enabled ensemble approach for medical data from body sensor networks," *Inf. Fusion*, vol. 53, pp. 155–164, Jan. 2020.

[12] A. Krtalić and M. Bajić, "Development of the TIRAMISU advanced intelligence decision support system," *Eur. J. Remote Sens.*, vol. 52, no. 1, pp. 40–55, Jan. 2019.

[13] W. Wang, "Human detection based on radar sensor network in natural disaster," in *Geological Disaster Monitoring Based on Sensor Networks*. Singapore: Springer, 2019, pp. 109–134.

[14] M. Reza Badello, B. Moshiri, B. N. Araabi, and H. Tebianian, "A novel detection and navigation approach based on OWA fusion method," *Sensor Rev.*, vol. 31, no. 4, pp. 328–340, Sep. 2011.

[15] O. Missaoui, H. Frigui, and P. Gader, "Multi-stream continuous hidden Markov models with application to landmine detection," *EURASIP J. Adv. Signal Process.*, vol. 2013, no. 1, pp. 1–23, Dec. 2013.

[16] J. Prado and L. Marques, "Reducing false-positives in multi-sensor dataset of landmines via sensor fusion regularization," in *Proc. IEEE Int. Conf. Auto. Robot Syst. Competitions (ICARSC)*, Apr. 2017, pp. 204–209.

[17] D. K. C. Ho and P. Plodpradista, "On the use of multiresolution analysis for subsurface object detection using deep ground penetrating radar," in *Detection Sens. Mines, Explosive Objects, Obscured Targets*, J. C. Isaacs and S. S. Bishop, Eds. Bellingham, WA, USA: SPIE, May 2019, p. 9.

[18] P. A. Torrione, K. D. Morton, R. Sakaguchi, and L. M. Collins, "Histograms of oriented gradients for Landmine detection in ground-penetrating radar data," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 3, pp. 1539–1550, Mar. 2014.

[19] X. Dai, L. Jiang, and Y. Zhao, "Cooperative exploration based on supervisory control of multi-robot systems," *Appl. Intell.*, vol. 45, pp. 18–19, Jan. 2016, doi: 10.1007/s10489-015-0741-3.

[20] J. Gomes, P. Mariano, and A. L. Christensen, "Challenges in cooperative coevolution of physically heterogeneous robot teams," *Natural Comput.*, vol. 18, no. 1, pp. 29–46, Sep. 2016.

[21] A. Ben Mekki, J. Tounsi, and L. Ben Said, "Fuzzy BDI agents for supply chain monitoring in an uncertain environment," *Supply Chain Forum, Int. J.*, vol. 17, no. 2, pp. 109–123, Apr. 2016.

[22] S. E. Yuksel, J. Bolton, and P. Gader, "Multiple-instance hidden Markov models with applications to landmine detection," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 12, pp. 6766–6775, Jul. 2015.

[23] J. C. Barca and Y. A. Sekercioglu, "Swarm robotics reviewed," *Robotica*, vol. 31, no. 3, pp. 345–359, Jul. 2012.

[24] P. G. Balaji and D. Srinivasan, "An introduction to multi-agent systems," in *Innovations in Multi-Agent Systems and Applications* (Studies in Computational Intelligence), vol. 310. Berlin, Germany: Springer-Verlag, 2010, ch. 1, pp. 1–27.

[25] A. Dorri, S. S. Kanhere, and R. Jurdak, "Multi-agent systems: A survey," *IEEE Access*, vol. 6, pp. 28573–28593, 2018.

[26] Y. Rizk, M. Awad, and E. W. Tunstel, "Decision making in multiagent systems: A survey," *IEEE Trans. Cognit. Develop. Syst.*, vol. 10, no. 3, pp. 514–529, Sep. 2018.

[27] M. Indiramma and K. R. Anandakumar, "Collaborative decision making framework for multi-agent system," in *Proc. Int. Conf. Comput. Commun. Eng.*, May 2008, pp. 1140–1146.

[28] M. Belaoued, A. Derhab, S. Mazouzi, and F. A. Khan, "MACoMal: A multi-agent based collaborative mechanism for anti-malware assistance," *IEEE Access*, vol. 8, pp. 14329–14343, 2020.

[29] C.-H. Yu, J. Werfel, and R. Nagpal, "Collective decision-making in multi-agent systems by implicit leadership," in *Proc. 9th Int. Conf. Auto. Agents Multiagent Syst.*, vol. 3. Richland, SC, USA: International Foundation for Autonomous Agents and Multiagent Systems, 2010, pp. 1189–1196.

[30] J. Florez-Lozano, F. Caraffini, M. Parra, and C. Gongora, "Source code—A robust decision-making framework based on collaborative agents," De Montfort Univ., Leicester, U.K., Tech. Rep., 2020, doi: 10.21253/DMU.12678905.

[31] Y. Liu, X. Wang, L. Li, S. Cheng, and Z. Chen, "A novel lane change decision-making model of autonomous vehicle based on support vector machine," *IEEE Access*, vol. 7, pp. 26543–26550, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8648365/

[32] S. El-Sappagh, F. Ali, A. Ali, A. Hendawi, F. A. Badria, and D. Y. Suh, "Clinical decision support system for liver fibrosis prediction in hepatitis patients: A case comparison of two soft computing techniques," *IEEE Access*, vol. 6, pp. 52911–52929, 2018.

[33] H. A. Mengash, "Using data mining techniques to predict student performance to support decision making in university admission systems," *IEEE Access*, vol. 8, pp. 55462–55470, 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9042216/

[34] L. Deng, Y. Hu, J. P. Y. Cheung, and K. D. K. Luk, "A data-driven decision support system for scoliosis prognosis," *IEEE Access*, vol. 5, pp. 7874–7884, 2017. [Online]. Available: http://ieeexplore.ieee.org/document/7907219/

[35] F. Miguel, M. Frutos, F. Tohme, and M. M. Babey, "A decision support tool for urban freight transport planning based on a multi-objective evolutionary algorithm," *IEEE Access*, vol. 7, pp. 156707–156721, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8884163/

[36] K. O. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *Evol. Comput.*, vol. 10, no. 2, pp. 99–127, Jun. 2002.

[37] K. O. Stanley, "Efficient evolution of neural networks through complexification," Ph.D. dissertation, Dept. Comput. Sci., Univ. Texas Austin, Austin, TX, USA, 2004.

[38] K. O. Stanley, D. B. D'Ambrosio, and J. Gauci, "A hypercube-based encoding for evolving large-scale neural networks," *Artif. Life*, vol. 15, no. 2, pp. 185–212, Apr. 2009.

[39] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing*, vol. 53. Berlin, Germany: Springer, 2003.

[40] D. M. W. Powers, "The problem of area under the curve," in *Proc. IEEE Int. Conf. Inf. Sci. Technol.*, Mar. 2012, pp. 567–573.

[41] P. Sun, D. Wang, V. C. Mok, and L. Shi, "Comparison of feature selection methods and machine learning classifiers for radiomics analysis in glioma grading," *IEEE Access*, vol. 7, pp. 102010–102020, 2019.

[42] J. Xie and Q. Wang, "Benchmarking machine learning algorithms on blood glucose prediction for type 1 diabetes in comparison with classical time-series models," *IEEE Trans. Biomed. Eng.*, early access, May 24, 2020, doi: 10.1109/TBME.2020.2975959.

[43] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, and S. García, "KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework," *J. Multiple-Valued Logic Soft Comput.*, vol. 17, nos. 2–3, pp. 255–287, 2011.

[44] R. P. Gorman and T. J. Sejnowski, "Analysis of hidden units in a layered network trained to classify sonar targets," *Neural Netw.*, vol. 1, no. 1, pp. 75–89, Jan. 1988.

[45] L. A. Kurgan, K. J. Cios, R. Tadeusiewicz, M. Ogiela, and L. S. Goodenday, "Knowledge discovery approach to automated cardiac SPECT diagnosis," *Artif. Intell. Med.*, vol. 23, no. 2, pp. 149–169, Oct. 2001. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0933365701000823

[46] *UCI Machine learning Repository*. Accessed: Jun. 17, 2020. [Online]. Available: http://archive.ics.uci.edu/ml/index.php

[47] O. L. Mangasarian, W. N. Street, and W. H. Wolberg, "Breast cancer diagnosis and prognosis via linear programming," *Oper. Res.*, vol. 43, no. 4, pp. 570–577, Aug. 1995. [Online]. Available: http://pubsonline.informs.org/doi/abs/10.1287/opre.43.4.570

[48] B. L. Bias, "Variance and arcing classifiers," Dept. Statist., Univ. California, Berkeley, CA, USA, Tech. Rep. 460, Apr. 1996.

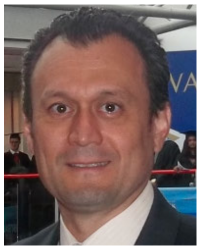
[49] K. O. Stanley. (2017). *Welcome to Neat-Python's Documentation*. [Online]. Available: https://neat-society.readthedocs.io/en/latest/#

[50] L. Baron, S. Achiche, and M. Balazinski, "Fuzzy decision support system knowledge base generation using a genetic algorithm," *Int. J. Approx. Reasoning*, vol. 28, nos. 2–3, pp. 125–148, Nov. 2001.
[51] J. Florez-Lozano, F. Caraffini, M. Parra, and C. Gongora, "Extended results—A robust decision-making framework based on collaborative agents," De Montfort Univ., Leicester, U.K., Tech. Rep., 2020, doi: 10.21253/DMU.12698771.
[52] C.-H. Yu, "Biologically-inspired control for self-adaptive multiagent systems," Ph.D. dissertation, School Eng. Appl. Sci., Inst. Biologically-Inspired Eng., Harvard Univ., Cambridge, MA, USA, 2010.

JOHANA M. FLOREZ-LOZANO (Student Member, IEEE) received the B.S. and M.S. degrees in electronic engineering and the Ph.D. degree in engineering from Pontificia Universidad Javeriana, Bogotá, Colombia, in 2009, 2014, and 2020, respectively. From 2008 to 2012, she has participated with the maintenance and development of digital control system for Oil products transportation. She has been a member with the Intelligent Systems, Perception and Robotics Research Group, Pontificia Universidad Javeriana, since 2009. From 2012 to 2014, she was a Research Assistant with the Internally Funded Project of the Research Group to detect electric transmission lines from images acquired by an UAV. She is currently a Lecturer with the Electronics Department, Pontificia Universidad Javeriana. Her research interests include machine learning techniques, robotics, information fusion, and signal and image processing.

FABIO CARAFFINI (Member, IEEE) received the B.Sc. degree in electronics engineering and the M.Sc. degree in telecommunications engineering from the University of Perugia, Italy, in 2008 and 2011, respectively, the Ph.D. degree in computer science from De Montfort University, U.K., in 2014, and the Ph.D. degree in computing and mathematical sciences from the University of Jyväskylä, Finland, in 2016. He is currently a Senior Lecturer in computer science and mathematics with De Montfort University. His research interests include theoretical and applied computational intelligence with a strong emphasis on metaheuristics for optimisation. He is a Fellow of the Higher Education Academy, U.K.

CARLOS PARRA (Member, IEEE) received the B.Sc. degree in electronics engineering from Pontificia Universidad Javeriana, Bogotá, Colombia, in 1992, the M.Sc. degree in electrical engineering from the Universidad de los Andes, Bogotá, in 1994, and the D.E.A. degree in control science and industrial computer science and the Ph.D. degree from Paul Sabatier University, Toulouse, France, in March 1999. He is currently a Professor with the School of Engineering, Pontificia Universidad Javeriana. His research interests include perception, mobile robots for natural environments, and artificial intelligence.



MARIO GONGORA received the degree in electronic engineering and the M.Sc. and Ph.D. degrees from The University of Warwick, U.K. He is currently an Associate Professor with the School of Computer Science and Informatics, De Montfort University (DMU). He is a Faculty Enterprise Lead and a Senior Member with the Institute of Artificial Intelligence, Faculty of Computing, Engineering and Media. He is also a Leading Researcher in applied computational intelligence. His research interests include robotics and machine learning, application of artificial intelligence techniques in the fields of data mining, large and complex data sets, and modeling natural phenomena, including identification, simulation, and optimisation.