Deep Neuroevolution: Training Deep Neural Networks for False Alarm Detection in Intensive Care Units

Hooman Oroojeni M. J.*, Mohammad Majid al-Rifaie*†, Mihalis A. Nicolaou*

*Department of Computing, Goldsmiths, University of London, SE14 6NW, UK
†Department of Computing and Information Systems, Old Royal Naval College
University of Greenwich, London, SE10 9LS, UK

***Abstract*—We present a neuroevolution based-approach for training neural networks based on genetic algorithms, as applied to the problem of detecting false alarms in Intensive Care Units (ICU) based on physiological data. Typically, optimisation in neural networks is performed via backpropagation (BP) with stochastic gradient-based learning. Nevertheless, recent works (c.f., [1]) have shown promising results in terms of utilising gradient-free, population-based genetic algorithms, suggesting that in certain cases gradient-based optimisation is not the best approach to follow. In this paper, we empirically show that utilising evolutionary and swarm intelligence algorithms can improve the performance of deep neural networks in problems such as the detection of false alarms in ICU. In more detail, we present results that improve the state-of-the-art accuracy on the corresponding Physionet challenge, while reducing the number of suppressed true alarms by deploying and adapting Dispersive Flies Optimisation (DFO).**

I. Introduction

The accurate detection of false alarms in the Intensive Care Unit (ICU) can be significantly beneficial for both patients and the healthcare system. False alarms in the ICU are likely to have a negative impact on the quality of care required for improving patients' health, due to noise disturbance, disruption of care, and lack of sleep impacting patient stress levels.

The focus of this paper lies with detecting abnormalities in the heart function, called arrhythmias, that can be encountered in both healthy and unhealthy subjects. The ICU is equipped with monitoring devices that are capable of detecting dangerous arrhythmias, namely asystole, extreme bradycardia, extreme tachycardia, ventricular tachycardia and ventricular flutter/fibrillation.

Depending on the AAMI guidelines, appropriate measures should be taken within 10 seconds of the commencement of the event as these arrhythmias might cause death [2]. Triggering the alarm when an arrhythmia occurs may improve the chance of saving lives. Mis-configuring, defective wiring, staff manipulation, and patient manipulation or movement may increase the false alarm ratio to 86%. Clinically, 6% to 40% of the ICU alarms proved to have lower priority and do not require immediate measures [3]. False alarms stimulate the patient's mental discomfort [4] and the clinical staff's desensitisation thereby causing a slower response to the triggered alarms [5]. True alarms that have high priority and need an urgent response are only 2 - 9% of all ICU alarms [6]. Therefore, false alarm detection and elimination is an essential area of research.

Deep Neural Networks (DNNs) trained via gradient-based algorithms such as backpropagation are the most common approach used in literature, where NN trained via BP have recently found great success in several applications related to areas such as computer vision and signal processing [7] - despite guarantees of achieving only a local optima when applied to non-convex loss functions. Furthermore, several recent works suggest that first-order methods such as gradient descent are often not the best way to go [1], and that population-based genetic algorithms can provide competitive results. In this paper, we evaluate the application of Dispersive Flies Optimisation (DFO) in finding the optimal weights of a given neural network (NN), instead of using BP. We evaluated the proposed gradient-free method on a subset of the Physionet Challenge 2015 dataset [8]. The goal of this challenge is to reduce the occurrences of false alarms with accurate detection of the above-mentioned life-threatening arrhythmias. This goal is achieved by using multi-modal data such as respiration (RESP), arterial blood pressure (ABP) or Photoplethysmograph (PPG).

II. Related Work

The Physionet Challenge 2015 [8] raises an opportunity for participants to offer different approaches towards improving the classification accuracy of true/false alarms. In this challenge, scoring was based on maximizing True Positives (TP) and True Negatives (TN), while minimising False Negatives (FN) and False Positives (FP). The scoring approach utilises an "err-on-the-safe side" approach, where the suppression of a true alarm (false negative) is penalized much higher than events such as raising a false alarm. This is described in the following equation,

$$Score = \frac{TP + TN}{TP + TN + FP + 5FN}. \quad (1)$$

In this challenge, 38 teams submitted their solutions. The participants used elementary algebra, descriptive statistics,

TABLE I
SUBSET OF PHYSIONET DATASET (572 OUT OF 750 RECORDINGS) THAT CONTAIN THE ECG LEADS II AND V AND PLETH SIGNAL. WE USED THIS SUBSET TO ENSURE THAT WE TRAIN THE NN MODELS ON IDENTICAL LEADS AND PULSATILE WAVEFORM.

Disease Name	True Alarm	False Alarm
Asystole	17	77
Bradycardia	35	37
Tachycardia	90	4
Ventricular Flutter/Fibrillation	6	40
Ventricular Tachycardia	54	212

signal processing, peak detection algorithms, QRS complexes localisation, and the annotation of heartbeats using all the available signals [9]–[18]. Binary Classification Trees (BCTs) [19], [20], Discriminant Analysis Classifiers (DACs) [19], Random Forest [21], Support Vector Machine (SVM) [19], [22], and Fuzzy Logic [10] are used in different submissions as well. The highest score (81.39) in the event one is achieved [10] by applying Fuzzy Logic along with Elementary Algebra and Descriptive Statistics. [19] applied SVM, DACs, and BCTs and ranked fifth in this challenge with a score of 75.55.

III. DATASET DESCRIPTION

The Physionet 2015 challenge [8] provides a training data set containing 750 recordings that are available to the public, as well as 500 hidden records for scoring. This dataset consists of life-threatening arrhythmia alarm records that have been collected from four hospitals in the United States and Europe. The recordings are sourced from the devices made by three major manufacturing companies of intensive care monitor devices. Each record is five minutes or five minutes and 30 seconds long at 250Hz and contains only one alarm. A team of expert annotators labelled them 'true' or 'false'. The commencement of the event is within the last ten seconds of the recordings.

In this challenge, participants can submit their code which would be evaluated according to two type of events: event 1 (real time) and event 2 (retrospective). All recordings have a sample rate of 250Hz and contain two ECG leads and one or more pulsative waveform (RESP, ABP or PPG). The ECGs may contain noise, and pulsatile channels may contain movement artefacts and sensor disconnections. In this paper, we focus on event 1, trim the data to have a length of five minutes and chose a subset of 572 records. These contain the ECG leads II and V and PLETH signal. This decision is made to ensure that we train the NN models on identical leads and pulsatile waveform. In this subset, there are 233 True alarms and 339 False alarms. In each n-fold, the dataset is divided into training, testing and validation using 70% (400), 20% (114), and 10% (58) respectively. Table I describes the dataset.

IV. FEATURE SELECTION

Since in this work our focus is finding the optimal weights of a given model rather than performing feature selection, we utilise the same feature selection method as in [19]. Having performed feature extraction, the authors utilise SVM, DACs and BCTs for predictive analysis. Achieved results indicate a score of 75.55 in the Physionet 2015 event 1. The 33 features which are listed in table II.

TABLE II
THE FEATURES USED TO TRAIN NN MODELS

Minimum optimal interval (1 - 3)
Maximum optimal interval (4 - 6)
Mean optimal interval (7 - 9)
Sum optimal interval (10 - 12)
Median absolute deviation of optimal interval (13-15)
STD optimal interval (16 - 18)
STD / mean optimal interval (19 - 21)
Mean peak height to area under curve ratio (22 - 24)
Median peak height to area under curve ratio (25 - 27)
High-frequency ECG (28)
High-frequency BP (29)
Low-frequency ECG (30)
Low-frequency BP (31)
Average rhythmicity (32)
Peak rhythmicity (33)

V. DISPERSIVE FLIES OPTIMISATION

DFO [23] is an algorithm inspired by the swarming behaviour of flies hovering over food sources. The swarming behaviour of the individuals in DFO consists of two tightly connected mechanisms: formation of the swarms, and their breaking or weakening[1]. In our study, we call the position as NN weights. Algorithm 1 describes the adapted DFO for NN. The NN weights' vector is defined as follows:

$$\vec{w}_i^t = \left[w_{i1}^t, w_{i2}^t, ..., w_{iD}^t\right], \qquad i = 1, 2, ..., N \tag{2}$$

where i represents the i^{th} individual, t is the current time step, and D is the dimensionality of the problem space. In our study D is the number of weights in a given NN model. N is the population size. For the continuous problems, $w_{id} \in \mathbb{R}$, and in discrete cases, $w_{id} \in \mathbb{Z}$ (or a subset of $\mathbb{Z}$). In our study w_{id} is a subset of $\mathbb{R}$. In the first iteration, $t = 0$, the i^{th} vector's d^{th} component is initialised as $w_{id}^0 = \mathcal{N}(0,\ 1)$,

where $\mathcal{N}$ denotes the Gaussian distribution. Therefore, the population is randomly initialised with a set of weights for each in the search space.

In each iteration of the original DFO equation, the components of the NN weights vectors are independently updated, taking into account the component's value, the corresponding value of the best neighbouring individual with the best Physionet score (consider ring topology), and the value of the best individual in the whole swarm. Therefore the updated equation is:

$$w_{id}^{t+1} \quad = \quad w_{i_{nd}}^t + u(w_{sd}^t - w_{id}^t) \tag{3}$$

where $w_{i_{nd}}^t$ is the weight (position) value of $\vec{w}_i^t$'s best *neighbouring* individual in the d^{th} dimension at time step t, w_{sd}^t is the value of the *swarm*'s best individual in the d^{th} dimension at time step t, and $u = U(0, 1)$ is a random

[1]Several elements play a role in *disturbing* the swarms; e.g., the presence of a threat causes the swarms to disperse, leaving their current position; they may return to the position immediately after the danger is over. However, during this period, if another position is discovered which better matches their criteria, the new position is adopted [24].

number generated from the uniform distribution between 0 and 1. In our study, we investigated several extensions to improve the performance of the DFO for deep network optimisation. We update equation 3 by taking into account the corresponding value of the best neighbouring individual and the value of the best in the whole swarm. Therefore, the adapted updated equation is as follows:

$$w_{id}^{t+1} = w_{ind}^{t} + u(w_{sd}^{t} - w_{ind}^{t}) \quad (4)$$

Three main components characterise the algorithm: A dynamic rule for updating the population's position (assisted by a neighbouring social network that informs this update) and communication of the results of the best-found individual to others. A dynamic mechanism to regulate *the disturbance threshold*, Δ, to control the behaviour of the population (exploration or exploitation) in the search space.

As stated earlier, the swarms are disturbed for various reasons; one of the impacts of these disturbances is the displacement of the individuals, which may lead to discovering a better Physionet score through finding better weights for the NN. To consider this eventuality, an element of stochasticity is introduced to the update process. Based on this, the individual components of the population's weights' vectors are reset if a random number, u is less than Δ. This approach guarantees a disturbance to the otherwise permanent stagnation over a possible local maximum. In the original DFO equation, the disturbance is done by updating the parameter with a random number in the acceptable range of minimum and maximum value. In our study, we changed this parameter's update to correlate with the current Δ and the best neighbour, that is w_{id}^{t+1} which is sampled from a Gaussian with the mean set to $w_{i_{nd}}$ and variance to Δ^2. Figure 3 demonstrates Δ's behaviour in 1000 iterations.

Algorithm 1 summarises the adapted DFO algorithm.

In this algorithm, each member of the population is assumed to have two neighbours (i.e. ring topology).

Algorithm 1 Adapted DFO for Training

Input: population size N, model structure L, network weights $\vec{w_i}$, length of weights vector D, loss function $f(.)$.

Δ = 1
while not converged **do**
  $\vec{w}_s$ = arg max $[f(L(\vec{w}_i))]$, $i \in \{1, \ldots, N\}$
  for $i = 1 \rightarrow N$ and $i \neq s$ **do**
    $\vec{w}_{i_n}$ = arg max $[f(L(\vec{w}_{i-1})), f(L(\vec{w}_{i+1}))]$
    for $d = 1 \rightarrow$ D **do**
      if $(U(0,1) < \Delta)$ **then**
        $w_{id}^{t+1} \leftarrow \mathcal{N}(w_{ind}^{t}, \Delta^2))$
      else
        $w_{id}^{t+1} \leftarrow w_{ind}^{t} + u(w_{sd}^{t} - w_{ind}^{t})$
      end if
    end for
    Dynamically update Δ (see Section VI-B)
  end for
end while

Output: Best fly's weight vector, $\vec{w_s}$.

TABLE III
DENSE MODEL STRUCTURE

Layer Name	No of Neurons	Weights Shape	Total Weights	Bias
Dense 1 (Input)	64	(1, 64)	64	64
Dense 2	64	(64, 64)	4096	64
Dense 3	32	(2112, 32)	67584	32
Dense 4 (Output)	2	(32, 2)	64	2

TABLE IV
CONVOLUTION-DENSE MODEL STRUCTURE

Layer Name	No of Neurons	Weights Shape	Total Weights	Bias
Convolution 1 (Input)	32	(2, 1, 32)	64	32
Dense 2 (Input)	64	(32, 64)	2048	64
Dense 3	64	(64, 64)	4096	64
Dense 4	32	(1024, 32)	32768	32
Dense 5 (Output)	2	(32, 2)	64	2

In summary, the DFO is a simple numerical optimiser. Despite the algorithm's simplicity, it is shown that DFO outperforms the standard versions of the well-known Particle Swarm Optimisation (PSO), Genetic Algorithm (GA) as well as Differential Evolution (DE) algorithms on an extended set of benchmarks over three performance measures of error, efficiency and reliability [23]. It is shown that DFO is more efficient in 84.62% and more reliable in 90% of the 28 standard optimisation benchmarks used; furthermore, when there exists a statistically significant difference, DFO converges to better solutions in 71.05% of the problem set. DFO has been applied to various domains including medical imaging [25], quantifying complexity in patterns [26] as well as parameter optimisation [27] and the philosophical autopoiesis argument in computational creativity [28].

VI. EXPERIMENTS

A. Model Configuration

In this study, we conducted an experiment utilising two neural network architectures: a stack of dense layers, and a stack of convolutional and dense layers. The model structures are described in Tables III & IV in detail. In NN, forward propagation includes a set of matrix operations where parameters include weights that connect NN layers to each other, and bias. We focus on finding optimal weights of the network that provide the highest classification accuracy for this task. Depending on the type of

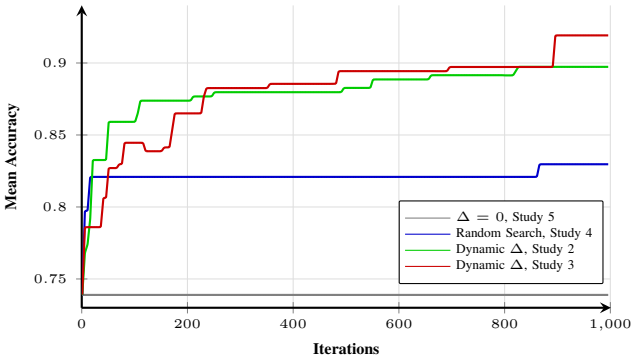

Fig. 1. Model 1: 5-fold cross validation mean accuracy over 1000 iterations. Mean accuracy trend for random search, standard DFO, updated DFO, with dynamic and constant disturbance threshold (Δ).

TABLE V
ACCURACY AND PHYSIONET SCORE OVER 5-FOLD CROSS VALIDATION FOR FIRST [10] AND FIFTH RANK [19] IN PHYSIONET CHALLENGE 2015, NN OPTIMISED WITH BP AND ADAPTED DFO ALGORITHM WITH CONSTANT AND DYNAMIC DISTURBANCE THRESHOLD (Δ)

Author	Method	Mean Accuracy	Physionet 2015 Score
By [19]	SVM, BCTs, DACs	87.24% (+/- 2)	85.50% (+/- 3)
By [10]	Fuzzy Logic	87.78% (+/- 4)	80.09% (+/- 8)
Our Study 1	Dense Network, BP	87.70% (+/- 3)	75.35% (+/- 7)
Our Study 2	Dense NN, Standard DFO, Dynamic Δ, using Eq. 3	90.02% (+/- 3)	79.23% (+/- 5)
Our Study 3	**Dense NN, Adapted DFO, Dynamic Δ, using Eq. 4**	**91.91% (+/- 4)**	**86.77% (+/- 4)**
Our Study 4	Dense NN, Random Search	84.16% (+/- 3)	68.03% (+/- 5)
Our Study 5	Dense NN, Standard DFO, $\Delta = 0$ (i.e. no disturbance)	73.89% (+/- 2)	51.53% (+/- 4)
Our Study 6	Convolution-Dense NN, BP	88.21% (+/- 3)	76.34% (+/- 6)
Our Study 7	Convolution-Dense NN, Standard DFO, Dynamic Δ, using Eq. 3	89.15% (+/- 4)	77.21% (+/- 5)
Our Study 8	**Convolution-Dense NN, Adapted DFO, Dynamic Δ using Eq. 4**	**91.88% (+/- 2)**	**86.81% (+/- 4)**
Our Study 9	Convolution-Dense NN, Random Search	82.39% (+/- 5)	66.40% (+/- 7)
Our Study 10	Convolution-Dense NN, Standard DFO, $\Delta = 0$ (i.e. no disturbance)	73.90% (+/- 3)	52.53% (+/- 2)

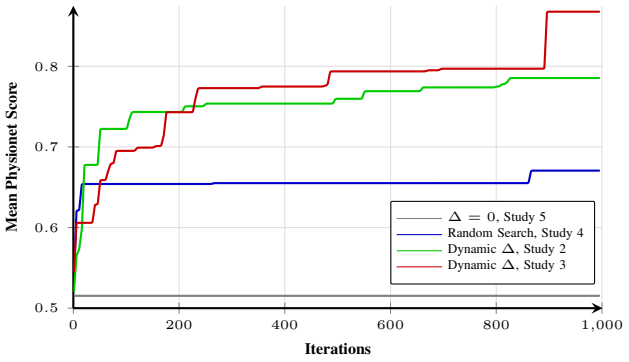

Fig. 2. Model 1: 5-fold cross validation mean Physionet score over 1000 iterations. Mean Physionet score trend for random search, standard DFO, updated DFO, with dynamic and constant disturbance threshold (Δ).

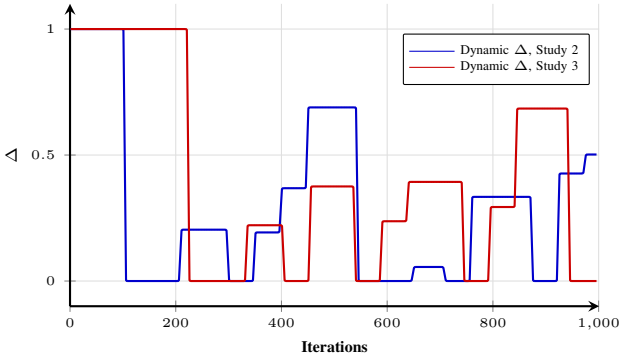

Fig. 3. Model 1: 5-fold cross validation disturbance threshold (Δ) trend. Visualising Δ trend with considering dynamic and constant disturbance threshold (Δ) over 1000 iterations.

neurons and input shape in a NN, the shape of the output weights of that neuron varies. This study uses two models: model 1 consists of four dense layers, and model 2 one convolution and four dense layers. For instance, in model 1 of our study, the first layer is dense with 64 neurons. In the first layer, input shape is (-1, 33, 1) and the output weight is (1, 64) where 1 is the third axis of the layer's input shape, and 64 is the number of neurons in this layer: $[(-1, 33, 1) \times (1, 64)] + (64) = (-1, 33, 64)$

Model 2 includes a convolution as the first layer. This model has a similar input shape (-1, 33,1). The first layer has 32 neurons, and the shape of connecting weights to the next layer is (-1, 2, 1, 32) where 2 is the convolution window size, 1 is the third axis of the input shape, and 32 is the number of neurons in the convolution layer.

B. DFO Configuration

In our study, DFO is used to find the optimal weights in both NN models. The number of parameters (i.e. weights and bias) in models 1 & 2 are 71970 and 39234 respectively (see Tables III and IV). Each member of the population (fly) has a set of parameters representing the weights (including biases) of the NN model. Once all the parameters of each fly are initialised and loaded onto the NN model, the fitness (score) of each fly is calculated using 5-fold cross-validation, by using the mean Physionet score. After each iteration, each fly's best neighbour, and the best fly in the swarm are identified. The best fly holds the highest Physionet score amongst the population.

Before updating each weight, a value u is sampled from a uniform distribution $U(0, 1)$. If u is less than Δ, the weight is updated with the fly's best neighbour as focus (μ), therefore, $w_{id}^{t+1} \leftarrow \mathcal{N}(w_{i_{nd}}^{t}, \Delta^2)$, otherwise, the fly's weight is updated with the focus on the best fly in the swarm $w_{id}^{t+1} \leftarrow w_{i_{nd}}^{t} + u(w_{sd}^{t} - w_{i_{nd}}^{t})$. We also implemented a mechanism to control the value of Δ. In the first phase or the parameter optimisation, Δ is set to 1 to bias the algorithm towards exploration; this process continues until there is no improvement in k iterations. Following this, Δ is set to zero, allowing the flies to converge to the best location. Once again, if no improvement is noticed in the Physionet score, Δ is increased by a random number between 0 and δ[2] ($\Delta \leftarrow \Delta + U(0, \delta)$). The algorithm is then set to run, if there is an improvement followed by a k iteration state of idleness, Δ is set to 0 again to exploit the recent finding. Alternatively if there is no improvement after the allowed idle time-frame, Δ is incremented further. In a situation where $\Delta > 1$, Δ is set to $U(0, 1)$, and the process continues as explained above until the termination points, which is $3,000$ iterations. Figure 1, 2 and 3 demonstrate trend of improvement in accuracy, Physionet score and variations of Δ over the first 1000 iterations.

[2]Note that throughout this in this paper, we use $k = 50$ and $\delta = 0.5$.

VII. Results

We compare the performance of the proposed DFO-training scheme for NN with the winning and 5th-ranked entries in the Physionet challenge 2015 (Table V), as well as the same NN architectures optimized with BP instead of the proposed gradient-free DFO training scheme. Model 1 consists of four dense layers and Model 2 consists of a convolution layer and four dense layers (see Table III & IV). As a benchmark, we use the accuracy and Physionet scores of [10] and [19], that achieve the accuracy and the Physionet scores of (87.24%, 85.50%) and (87.78%, 80.09%) respectively. We further investigate various implementations of DFO, as well as random search. For all experiments, we use 5-fold cross-validation. The DFO modified version for Model 1 & 2 achieved the highest score among the other results (see Table V). Resulting accuracy and Physionet scores are (91.91%, 86.77%) and (91.88%, 86.81%) respectively, achieving better results than both back propagation-trained NN, as well as challenge entries. The behaviour of DFO, while having the constant Δ value of 0 and random search, is also investigated. Their accuracy and Physionet scores are (73.89%, 51.53%) and (84.16, 68.03) respectively. The models optimised via neuroevolution with DFO outperforming both (i) the networks trained by BP, as well as (ii) the winning entry of the Physionet challenge.

VIII. Conclusion and Future Work

We presented a method for training neural networks based on neuroevolution by utilising the DFO algorithm in a gradient-free population-based scheme. We evaluate our method on the problem of detecting false alarms in ICUs. Results show that the proposed method outperforms (i) backpropagation-trained networks with the same or similar architecture, as well as (ii) the winning entries of the corresponding Physionet challenge.

References

[1] F. P. Such, V. Madhavan, E. Conti, J. Lehman, K. O. Stanley, and J. Clune, "Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning," *arXiv preprint arXiv:1712.06567*, 2017.

[2] A. for the Advancement of Medical Instrumentation *et al.*, "Cardiac monitors, heart rate meters, and alarms," *American National Standard (ANSI/AAMI EC13: 2002) Arlington, VA*, pp. 1–87, 2002.

[3] S. T. Lawless, "Crying wolf: false alarms in a pediatric intensive care unit." *Critical care medicine*, vol. 22, no. 6, pp. 981–985, 1994.

[4] S. Parthasarathy and M. J. Tobin, "Sleep in the intensive care unit," *Intensive care medicine*, vol. 30, no. 2, pp. 197–206, 2004.

[5] M.-C. Chambrin, "Alarms in the intensive care unit: how can the number of false alarms be reduced?" *Critical Care*, vol. 5, no. 4, p. 184, 2001.

[6] C. L. Tsien and J. C. Fackler, "Poor prognosis for existing monitors in the intensive care unit," *Critical care medicine*, vol. 25, no. 4, pp. 614–619, 1997.

[7] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[8] G. D. Clifford, I. Silva, B. Moody, Q. Li, D. Kella, A. Shahin, T. Kooistra, D. Perry, and R. G. Mark, "The physionet/computing in cardiology challenge 2015: reducing false arrhythmia alarms in the icu," in *Computing in Cardiology Conference (CinC), 2015*. IEEE, 2015, pp. 273–276.

[9] S. Fallet, S. Yazdani, and J.-M. Vesin, "A multimodal approach to reduce false arrhythmia alarms in the intensive care unit," in *Computing in Cardiology Conference (CinC), 2015*. IEEE, 2015, pp. 277–280.

[10] F. Plesinger, P. Klimes, J. Halamek, and P. Jurak, "False alarms in intensive care unit monitors: detection of life-threatening arrhythmias using elementary algebra, descriptive statistics and fuzzy logic," in *Computing in Cardiology Conference (CinC), 2015*. IEEE, 2015, pp. 281–284.

[11] P. Couto, R. Ramalho, and R. Rodrigues, "Suppression of false arrhythmia alarms using ecg and pulsatile waveforms," in *Computing in Cardiology Conference (CinC), 2015*. IEEE, 2015, pp. 749–752.

[12] S. Ansari, A. Belle, and K. Najarian, "Multi-modal integrated approach towards reducing false arrhythmia alarms during continuous patient monitoring: the physionet challenge 2015," in *Computing in Cardiology Conference (CinC), 2015*. IEEE, 2015, pp. 1181–1184.

[13] C. Liu, L. Zhao, and H. Tang, "Reduction of false alarms in intensive care unit using multi-feature fusion method," in *Computing in Cardiology Conference (CinC), 2015*. IEEE, 2015, pp. 741–744.

[14] N. Sadr, J. Huvanandana, D. T. Nguyen, C. Kalra, A. McEwan, and P. de Chazal, "Reducing false arrhythmia alarms in the icu by hilbert qrs detection," in *Computing in Cardiology Conference (CinC), 2015*. IEEE, 2015, pp. 1173–1176.

[15] C. Tsimenidis and A. Murray, "Reliability of clinical alarm detection in intensive care units," in *Computing in Cardiology Conference (CinC), 2015*. IEEE, 2015, pp. 1185–1188.

[16] S.-K. Teo, J. C. Wong, B. Yang, F. Yang, L. Feng, T. W. Lim, and Y. Su, "Reducing false arrhythmia alarms in the icu," in *Computing in Cardiology Conference (CinC), 2015*. IEEE, 2015, pp. 1177–1180.

[17] J. J. Gierałtowski, I. Grzegorczyk, K. Ciuchciński, K. Kośna, M. Soliński, and P. Podziemski, "Algorithm for life-threatening arrhythmias detection with reduced false alarms," in *Computing in Cardiology Conference (CinC), 2015*. IEEE, 2015, pp. 1201–1204.

[18] R. He, H. Zhang, K. Wang, Y. Yuan, Q. Li, J. Pan, Z. Sheng, and N. Zhao, "Reducing false arrhythmia alarms in the icu using novel signal quality indices assessment method," in *Computing in Cardiology Conference (CinC), 2015*. IEEE, 2015, pp. 1189–1192.

[19] C. H. Antink and S. Leonhardt, "Reducing false arrhythmia alarms using robust interval estimation and machine learning," in *Computing in Cardiology Conference (CinC), 2015*. IEEE, 2015, pp. 285–288.

[20] M. Caballero and G. M. Mirsky, "Reduction of false cardiac arrhythmia alarms through the use of machine learning techniques," in *Computing in Cardiology Conference (CinC), 2015*. IEEE, 2015, pp. 1169–1172.

[21] L. M. Eerikäinen, J. Vanschoren, M. J. Rooijakkers, R. Vullings, and R. M. Aarts, "Decreasing the false alarm rate of arrhythmias in intensive care using a machine learning approach," in *Computing in Cardiology Conference (CinC), 2015*. IEEE, 2015, pp. 293–296.

[22] V. Kalidas and L. S. Tamil, "Enhancing accuracy of arrhythmia classification by combining logical and machine learning techniques," in *Computing in Cardiology Conference (CinC), 2015*. IEEE, 2015, pp. 733–736.

[23] M. M. al-Rifaie, "Dispersive flies optimisation," in *Proceedings of the 2014 Federated Conference on Computer Science and Information Systems*, 2014, pp. 529–538.

[24] J. Downes, "The swarming and mating flight of diptera," *Annual review of entomology*, vol. 14, no. 1, pp. 271–298, 1969.

[25] M. M. al-Rifaie and A. Aber, "Dispersive flies optimisation and medical imaging," in *Recent Advances in Computational Optimization*. Springer, 2016, pp. 183–203.

[26] M. M. al-Rifaie, "On symmetry, aesthetics and quantifying symmetrical complexity," in *Evolutionary and Biologically Inspired Music, Sound, Art and Design*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2017, accepted and in press.

[27] H. A. Alhakbani and M. M. al-Rifaie, "Optimising SVM to classify imbalanced data using dispersive flies optimisation," in *Proceedings of the 2017 Federated Conference on Computer Science and Information Systems*. IEEE, 2017, pp. 399–402.

[28] M. M. al Rifaie, F. F. Leymarie, W. Latham, and M. Bishop, "Swarmic autopoiesis and computational creativity," *Connection Science*, pp. 1–19, 2017. [Online]. Available: http://dx.doi.org/10.1080/09540091.2016.1274960