IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 8 May 2022, accepted 30 May 2022, date of publication 4 July 2022, date of current version 11 July 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3187961

The Deep Radial Basis Function Data Descriptor (D-RBFDD) Network: A One-Class Neural Network for Anomaly Detection

MEHRAN HOSSEIN ZADEH BAZARGANI, ARJUN PAKRASHI, AND BRIAN MAC NAMEE
School of Computer Science, University College Dublin (UCD), Belfield, Dublin 4, D04 V1W8 Ireland

Corresponding author: Mehran Hossein Zadeh Bazargani (mehran.hosseinzadehbazargani@ucd.ie)

This work was supported by the Science Foundation Ireland under Grant 15/CDA/3520 and Grant 12/RC/2289_P2.

ABSTRACT Anomaly detection is a challenging problem in machine learning, and is made even more so when dealing with instances that are captured in low-level, raw data representations without a well-known and well-behaved set of engineered features. Images or data streams from sensors are good examples of such low-level, raw data representations. The Radial Basis Function Data Descriptor (RBFDD) network is an effective solution for anomaly detection, however, it is a shallow model that does not deal well with low-level, raw data representations. This article investigates approaches to transform an RBFDD network into a deep one-class classifier that works well for anomaly detection problems with low-level, raw data representations. We show that approaches based on simple transfer learning are not effective and our results suggest that this is because the latent representations learned by generic classification models are not suitable for anomaly detection. We show that an alternative approach that adds multiple convolutional layers before the Radial Basis Function (RBF) layer of an RBFDD network——to form a Deep Radial Basis Function Data Descriptor (D-RBFDD) network—is very effective. This is demonstrated in a set of evaluation experiments based on multiple anomaly detection scenarios created from publicly available image classification datasets, and a real-world anomaly detection dataset in which different types of arrhythmia are detected in electrocardiogram (ECG) data. Our experiments show that the D-RBFDD network outperforms state-of-the-art anomaly detection methods including the Deep Support Vector Data Descriptor (Deep SVDD), One-Class Support Vector Machine (OCSVM), and Isolation Forest on the image datasets, and produces competitive results on the ECG dataset.

INDEX TERMS Anomaly detection, one-class classification, artificial neural networks, deep learning.

I. INTRODUCTION

Chandola & Kumar [1] define *anomaly detection* as “*the problem of finding patterns in data that do not conform to expected behavior*”. Building machine learning models for anomaly detection is made especially challenging by limited, or no, access to anomalous patterns during training. *One-class classification* [2], in which a machine learning model is trained to recognize normal data and flag an anomaly when something fails to be recognized as normal, is a common approach to anomaly detection. Like any other machine learning task, however, training a model can be challenging when dealing with raw data as opposed to data based on a set of well-behaved engineered features—for example, when working with image data [3], audio data [4], or streaming data from sensors [5]. Shallow models particularly suffer from this issue, and it is common practice to employ deep learning [6] in these scenarios.

In this article we propose a deep anomaly detection model that can be trained in a fully end-to-end fashion, and is suitable for use with raw, high-dimensional data sources such as images and timeseries data from sensors. Our proposed model, the *Deep Radial Basis Function Data Descriptor* (D-RBFDD) network, is based on our previous

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

work on the Radial Basis Function Data Descriptor (RBFDD) network [7]. RBFDD networks (which in turn are based on Radial Basis Function (RBF) networks [8], [9]) are effective and efficient anomaly detectors that learn a compact set of Gaussian kernels to cover the normal region of input space, and recognize anomalies as instances that sit outside this region. The positions of these learned kernels, and the magnitude of the weights connecting each of them to the output layer, also facilitate straight-forward post-hoc explanation of outputs [10]–[12]. Typical RBF networks (including RBFDD networks), however, are shallow neural networks with a single hidden layer and do not perform well when trained on low-level, raw data [7]. This motivates deepening these kind of networks to make them more effective in these scenarios.

We identify three ways to deepen RBFDD networks to work with raw input data:

1) Based on simple transfer learning using the latent representation from a generic pre-trained network as input to the RBFDD network.
2) By extending the first approach to include fine-tuning the pre-trained network as part of training the RBFDD network.
3) By adding multiple computational layers (or hidden layers) before the RBFDD network that are fully trained with the rest of the network in an end-to-end fashion.

Our exploration of the effectiveness of these three approaches, shows that the final approach that trains the entire network from random initialization—referred to as *Deep RBFDD* (D-RBFDD)—out-performs the other two deepening approaches. This addresses a fundamental question of whether the latent representations learned by large models trained for multi-class classification (the type of model most commonly used for simple transfer learning) are suitable as input for anomaly detection models, or whether they are too entangled with the multi-class classification problem. The result of our evaluation experiments—using anomaly detection scenarios constructed from well-known image classification datasets and a dataset from a real-world electrocardiogram (ECG) anomaly detection task—suggest that they are not suitable. Our results also show that the D-RBFDD approach out-performs existing state-of-the-art anomaly detection approaches (including RBFDD) on the image datasets, while producing competitive results on the ECG dataset. D-RBFDD is therefore an effective anomaly detector trained in an end-to-end fashion, that also has the advantages that come with an approach based on RBF networks: it is efficient, it lends itself to easy interpretation, and the local learning approach can adapt to dynamic definitions of normality and accommodate concept drift [13].

The main contributions of this article are:

- We propose an effective anomaly detection approach, the D-RBFDD network, for deepening RBFDD networks for the task of anomaly detection. This enables RBFDD networks to learn from low-level, raw data representations.
- It is shown that the latent representations learned by large models trained for multi-class classification are not suitable as input for anomaly detection models.

The remainder of this article is structured as follows. In Section II we describe common approaches to anomaly detection including deep learning methods. In Section III we describe the RBFDD network approach and illustrate different methods for deepening it. The setup of our experiments is described in Section IV. The results of these experiments are presented in Section V and discussed in Section VI. Section VII summarises the conclusions from our work and Section VIII discusses directions for future work.

II. RELATED WORK

Machine learning approaches to anomaly detection are dominated by a family of algorithms that adapt the Support Vector Machine (SVM) [8] algorithm to work with only examples of a single class: One-Class SVM (OCSVM) [14]. In fact Khan & Madden [2] go so far as to say that one-class classification approaches should be divided into two categories: *OCSVM-based and everything else*. In this section we briefly describe both the OCSVM-based approaches and *everything else*. Within the latter category deep learning approaches have emerged in recent years and can be can be categorized as either *mixed* or *end-to-end* approaches. We review the most important methods in each category.

A. OCSVM-BASED ANOMALY DETECTION

Much like the standard SVM approach, OCSVM models separate normal data points from the origin in the feature space using a hyper-plane found by maximizing the distance between the origin and this hyper-plane. At test time, normal instances will be found beyond this hyperplane while anomalies will be found between the hyper-plane and the origin. Although any kernel function can be used with OCSVM models, Gaussian kernels work particularly well [2]. The main issue with OCSVM models is that they do not scale well. For large datasets the computational and storage requirements of OCSVMs grow polynomially with dataset size [15]. The Support Vector Machine Data Description (SVDD) model [16] is a well known variation of OCSVM that uses hyper-spheres rather than hyper-planes to achieve separation. The goal when training an SVDD model is to find a tight spherically shaped boundary that encompasses the normal data.

B. NON-OCSVM-BASED ANOMALY DETECTION

On the non-OCSVM side, Isolation Forests (iForests) [17] and Auto-Encoder neural networks (AENs) [6] (and their many variations such as Variational Auto-Encoders (VAEs) [18] and De-noising Auto-Encoders (DAEs) [19]) are effective anomaly detectors. An iForest model isolates individual data points in a training set by constructing a decision tree that splits the input space randomly and repeatedly. The intuition behind this approach is that fewer splits should be required to isolate anomalous instances than normal ones. An auto-encoder network (AEN) is an artificial neural network that learns to compress and encode input data into a lower dimension and then reconstruct the input data back from this compressed representation. If an AEN is trained to reconstruct only *normal* data instances, it can detect anomalies by flagging test instances for which the reconstruction error is very high. Deep auto-encoders have been shown to be effective on problems with raw data inputs [18], [20].

In the remainder of this section we review the most important deep learning approaches for anomaly detection categorising them as either *mixed* or *end-to-end* approaches.

1) MIXED APPROACHES

In a *mixed approach* a deep model is trained in an unsupervised way to work as a feature extractor that produces the data for a, typically shallow, anomaly detector—for example an OCSVM. The deep models used for learning features tend to be reconstruction-based models such as Deep Belief Networks (DBNs) or deep AENs [21]–[23]. For example, in [24] an unsupervised DBN is trained to extract generic underlying features, and an OCSVM anomaly detector is trained using these features. The mixture of a DBN and a OCSVM is shown to out-perform a standalone OCSVM.

In [25], in order to detect anomalous behavior in large-scale network traffic data, a DBN model is trained as an unsupervised dimensionality reduction step, whose output features are then fed into a multi-layered ensemble SVM. In [26], a fully unsupervised model is proposed for detecting anomalous frames in video. For every frame of the video appearance and motion features are extracted and fed into two separate Stacked Denoising Auto-Encoders (SDAEs). A fusion of these two types of features are fed into a third SDAE. The features obtained in the bottle-neck layers of the three SDAEs are then fed into three OCSVMs, each of which produces an anomaly score. The three anomaly scores are combined to make the final decision for an input video frame. Similarly, in [27] a hybrid of DBN and Long Short Term Memory (LSTM) networks is used to detect behavioral anomalies (e.g., DOS attacks, web attack, and infiltration) in computer networks. This approach, however, relies on some access to labeled anomalies during training, essentially treating it as a multi-class classification problem. First, using a random sub-sample of the data, the DBN network is pre-trained in an unsupervised fashion, after which, the training proceeds in a supervised fashion to make the outputs of the pre-trained DBN network capture the important features within the computer traffic data. In a final step, the temporal patterns within the data are modeled by adding LSTM layers on top of the pre-trained DBN network.

In [28], a two-stage semi-supervised approach to anomaly detection for image data is proposed, where deep representations from the normal data are first learned by solving a proxy classification task. Next, these learned representations are used to train a one-class classifier. The proxy classification task is to distinguish between normal data instances and synthetic anomalies created using the Cut-Paste [29] data augmentation technique (Cut-Paste cuts an image patch from one image and pastes it at a random location of another image to create an anomaly). The use of synthetic anomalies is designed to build models that will generalise to unseen real anomalies at test time. After the proxy classifier has been trained, for a given input data, the classifier would produce an output. Finally, a Gaussian estimator one-class classifier is used to compute the anomaly score for that input data, using the output. The *log* density of the Gaussian estimator is used as the anomaly score and both its mean and covariance are computed solely using the normal data samples.

In [30], a ResNet model [31], pre-trained on the ImageNet [31] multi-class classification dataset, is used to transform image data to a learned feature space. In addition, the ResNet model is further fine-tuned by minimizing a compactness loss similar to SVDD [16], which makes its learned features more concentrated and suitable for the task of anomaly detection. An anomaly score for a query data instance can then be computed by measuring the average distance from its k-nearest neighbor normal data points in this feature space.

The main issue with mixed approaches is that the deep feature extractor is not trained for an anomaly detection objective, but on a different objective such as minimizing reconstruction error. As a result, the features learned may not be useful inputs for the anomaly detection model.

2) END-TO-END APPROACHES

End-to-end approaches address the issue with the mixed approaches, and aim to make the latent representations learned more appropriate for anomaly detection by defining a one-class cost function. The cost function is then used to train an entire network in an end-to-end fashion, guiding the network to produce representations that are appropriate for anomaly detection. For example, the Deep Auto-encoding Gaussian Mixture Model (DAGMM) [32] uses an AEN to reduce the dimensionality of the input data, and the reconstruction error and low-dimensional representation from the bottle-neck of the AEN are fed into a Gaussian Mixture Model (GMM) [8]. Similarly, the One-Class Neural Network (OCNN) [33] combines the rich feature extraction property of deep neural networks with a novel OCSVM-like cost function. First, a deep AEN is trained to produce good representative features of the input data. Next, the encoder portion of this pre-trained AEN is fed into a simple one-layer neural network, the final output of which is used to compute the cost. The weights of both the encoder and the one-layer neural network are trained simultaneously, by minimizing the cost function. By combining the capability of deep neural networks to extract rich features from the data with the proposed cost function, the aim is to obtain the hyper-plane that separates the normal data from the origin.

AnoGAN [34] is a deep model for anomaly detection based on Generative Adversarial Networks (GANs) [35]. The generator network is trained to learn the distribution of the training data. Given a test instance it searches for a point in the latent space of the generator that would generate a sample that is closest to the test point. If an accurate distribution of the normal data has been learned, for a normal query, x, there should be a representation, z, in the latent space that the generator could use to generate a new data point, $G(z)$, that is similar to the normal query x. For an anomalous query a good representation, z, should not be found and, as a result, the generated data, $G(z)$, will not be similar to the query.

Finally, inspired by the Support Vector Data Descriptor (SVDD) model [16], Deep-SVDD [36] is another deep one-class neural network designed for anomaly detection. While the neural network is trained, the volume of a hyper-sphere that envelopes the data in the latent space is minimized. Thus, the neural network is trained to map the input data into a hyper-sphere with minimum volume. There are two versions of Deep SVDD: (1) Soft-boundary Deep SVDD which makes a compromise between the volume of the hyper-sphere and violations of the boundary; and (2) One-Class Deep SVDD which is a simpler version that assumes that most of the training data is normal.

The effectiveness of end-to-end approaches, and in particular end-to-end approaches optimized using a cost function with a direct anomaly detection objective, motivates our proposed D-RBFDD network. Moreover, it is desirable for the anomaly detectors to be both interpretable and adaptable to new data and changing concepts of normality. These characteristics are not easily associated with SVMs [37]–[39], or approaches built upon an SVM foundation [40], unfortunately. In contrast, because of their local learning approach, RBF networks easily lend themselves to interpretation [10], [41], and are adaptable to changing concepts [42], [43]. This makes a deep end-to-end anomaly detector based on RBF networks an attractive approach. D-RBFDD—a fast, effective anomaly detector, trained in an end-to-end fashion and capable of learning latent representations directly aligned with an anomaly detection objective, that lends itself to easy interpretation and adaptation to changing concepts of normality—is explained in detail in the next section.

III. DEEP RADIAL BASIS FUNCTION DATA DESCRIPTOR (D-RBFDD) NETWORKS

This section describes the RBFDD network and three alternatives for adding depth to these networks, the last of which we refer to as the Deep RBFDD (D-RBFDD) network.

A. RADIAL BASIS FUNCTION DATA DESCRIPTOR (RBFDD) NETWORKS

In our previous work [7] we proposed the RBFDD network, which adapts the Radial Basis Function (RBF) network for anomaly detection. An RBF network is a local-representation learning technique used for classification that divides the input space among a set of local kernels. In an RBF network, for every input data point (depending on where in the input space it appears) a fraction of these locally-tuned kernel units gets activated. Activation is measured using a function of the distance between an input instance, X, and the center, μ_h, of every kernel unit h. When training an RBF network, it is common practice to use a statistical clustering method (e.g., K-means clustering [8]) as a pre-training stage to initialize kernel centers. This makes sure that the centers of the Gaussians are placed in the regions of the input space where training data resides, which facilitates the training process. Finally, the distance metric typically used in RBF networks is Euclidean distance, $||X - \mu_h||$, and the activation function for the local kernels (i.e., Gaussians) is usually implemented using a Gaussian function:

$$P_h(X) = \exp\left(-\frac{||X - \mu_h||^2}{2s_h^2}\right) \tag{1}$$

where μ_h and s_h denote the mean and standard deviation of the kernel unit h. Activation is maximum when $X = \mu_h$, and decreases as X and μ_h diverge.

RBFDD networks adapt the RBF network approach to learn a set of Gaussian kernels that compactly represent normal instances in a training set, thus transforming them into anomaly detectors. A trained RBFDD network can be used as an anomaly detector by recognizing instances that are not covered by this compact representation. Figure 1 shows the architecture of an RBFDD network. Here x_d denotes the d^{th} feature in the input data X, which is a D-dimensional vector. In the output node of the network the *tanh* non-linear activation function proposed in [44] (i.e., $1.7159 \times tanh\left(\frac{2}{3}z\right)$) is used, as it avoids saturation. More specifically, as shown later in Eq.(4), during training an RBFDD network is pushed to output values as close as possible to the target value of $+1$. Unlike the traditional *tanh* activation function, with the *tanh* activation function proposed in [44], the desired $+1$ target value happens in the non-saturated area of the function, where the input value to the function is exactly equal to $+1$—this is where the non-linearity of the function is maximum.

For a given D-dimensional input data instance, X_i, the output of an RBFDD model is computed as:

$$y_i = 1.7159 \times tanh\left(\frac{2 \times z(X_i)}{3}\right) \tag{2}$$

$$z(X_i) = \sum_{h=1}^{H} w_h \times P_h(X_i) \tag{3}$$

where w_h is a weight connecting the Gaussian kernel h to the output unit.

After training, the output of the RBFDD network, y_i, for a given input, X_i, should be high if X_i belongs to the normal region of the input space and low otherwise. In the RBFDD network, the unsupervised pre-training phase used to train RBF networks [9] (i.e., k-means clustering [8]) remains

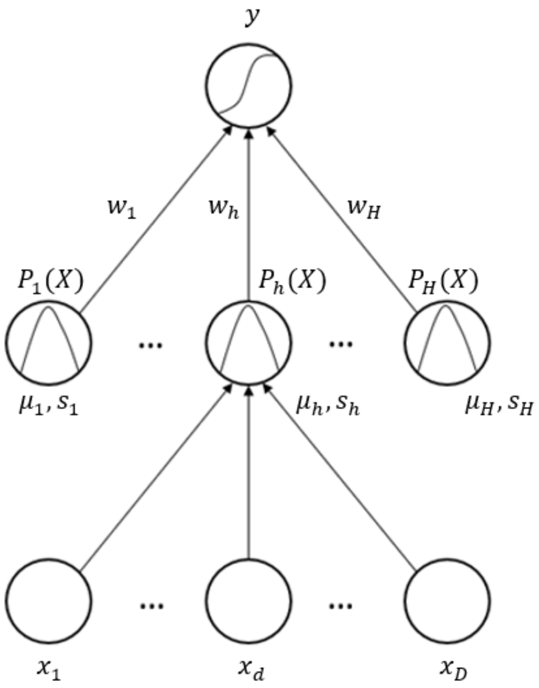


FIGURE 1. The RBFDD network.

in place. Following this step, the backpropagation of error algorithm is used with gradient descent to find the optimal values for the network parameters. In this process the cost function minimized for mini-batches of size N is:

$$E = \sum_{i=1}^{N}\left(\frac{1}{2}\left[(1-y_i)^2 + \beta\sum_{h=1}^{H}s_h^2 + \lambda\sum_{h=1}^{H}w_h^2\right]\right) \tag{4}$$

This cost function is a weighted summation of three terms. In the first term, $(1-y_i)^2$, y_i is the output of the RBFDD network for input data instance, X_i. This term encourages the network to learn a model that outputs a value as close as possible to $+1$ for instances belonging to the normal class. The second term, regularizes the spreads of the Gaussian kernels in the hidden layer of the network, and is the squared L-2 norm [6] of the spreads for the H Gaussians in the network. This encourages the most compact set of Gaussians capable of representing the normal data to be found. The third term, regularizes the weights connecting the RBFDD hidden layer units to the output unit. This stops the weights from becoming so large that they would actually ignore the outputs from the hidden units, and makes the RBFDD network robust to outliers in the training set [6]. Minimizing Eq.(4), using gradient descent, finds the most compact set of Gaussian kernels whose collective output is still high for the normal region of the input space and low everywhere else (i.e., where the anomalies are expected to appear). RBFDD networks use radial kernels, and thus, they might lack the necessary flexibility to learn certain distributions. To overcome this limitation of RBFDD, we previously proposed the Elliptical Basis Function Data Descriptor (EBFDD) [45], where we make the anomaly detector more flexible by replacing the radial kernels with elliptical kernels. EBFDD was shown to perform better than RBFDD, however, it achieved this at significantly increased computational cost (EBFDD requires a covariance matrix inversion which is a very computationally expensive operation). We believe, however, that the same flexibility can be achieved by adding computational layers before the RBFDD layer to transform the input data into a space where RBFDD can be applied effectively. Also, this can be done efficiently due to the lower computation time of RBFDD compared to EBFDD. Thus, we avoid the computational complexity of the EBFDD networks by adding more layers and retaining the capacity of the deep model to learn complex distributions in the normal data.

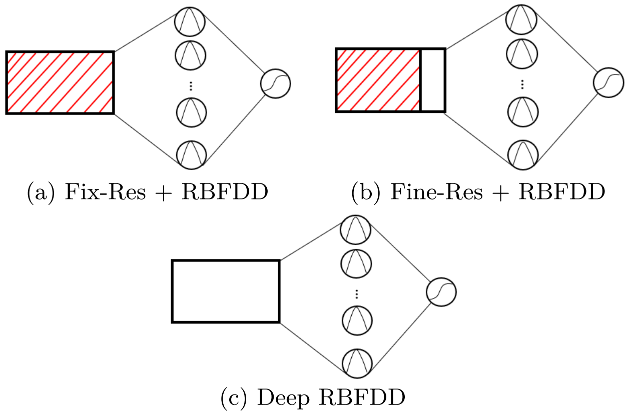


FIGURE 2. Three approaches to deepening RBFDD networks. The red hatching highlights the fixed portion of each model, while the white portions are trainable.

Although the RBFDD network is an effective anomaly detector when used with well-behaved sets of input features, it does not perform well on high-dimensional raw data representations (e.g., pixel values in images or raw sensor data). This is the main motivation for deepening the structure of the RBFDD network so that we can apply it to anomaly detection problems with raw high-dimensional input data. The next section describes different alternatives for placing extra computational layers before the RBFDD network.

B. DEEPENING RBFDD NETWORKS

We explore three ways to add depth to RBFDD networks (illustrated in Figure 2):

- *Transfer learning* [46] is exploited and the latent representation produced at the final layer of a network that is already trained on a large generic dataset is used as input to the RBFDD network.
- The latent representation from a pre-trained network, such as that described above, can be *fine-tuned* as part of training an RBFDD network using the cost function in Eq.(4).
- Computational layers placed before the RBFDD network can be trained from random initialization as part of an *end-to-end* deep RBFDD training process.

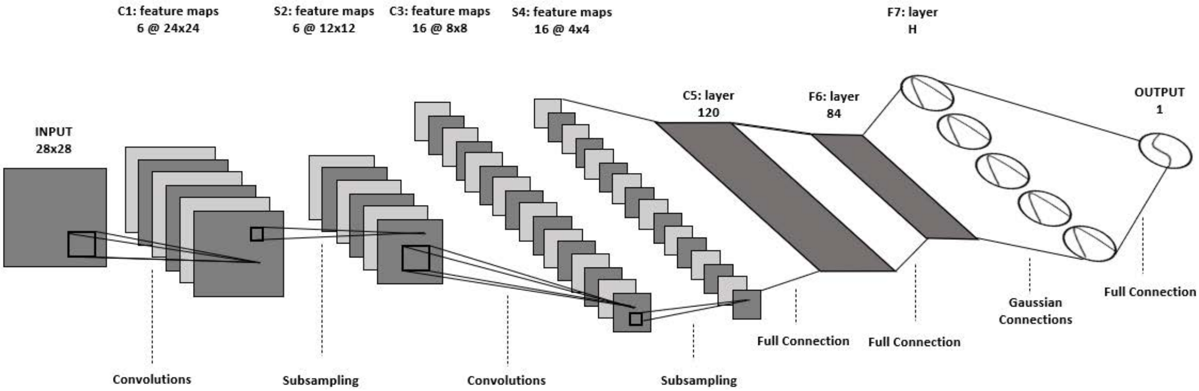


FIGURE 3. The deep RBFDD anomaly detector.

For the approach using transfer learning, referred to as *Fix-Res + RBFDD*, we use a fixed, pre-trained ResNet-18 model trained on the ImageNet [31] dataset, and extract the latent representation after its last convolutional layer for each data instance. This representation is then passed to a standard RBFDD model. In the second approach, referred to as *Fine-Res + RBFDD*, again we connect a pre-trained ResNet-18 model to an RBFDD model. In this case, however, we fine-tune the last 4 convolutional layers, and the last 4 batch-norm layers of the pre-trained ResNet-18 model as part of training the RBFDD model using the cost function in Eq.(4). This fine-tuning step is expected to make the latent representation passed to the RBFDD model more appropriate for anomaly detection, and improve the overall performance.

The final approach, that we refer to as the Deep RBFDD (D-RBFDD) network, attaches randomly initialized computational layers before the RBFDD layer and trains the entire network in an end-to-end fashion based on minimizing the cost function in Eq.(4). Provided that the cost function can generate sufficient signal to train the entire deep model, the advantage of this method is that by using end-to-end training the latent representation that this network passes to the RBFDD layer will be more suited to anomaly detection than the representation generated by the pre-trained classification network, even when it is fine-tuned. In D-RBFDD we add layers following the simple LeNet-5 network architecture [47] preceding the RBFDD layer. The overall D-RBFDD network architecture is illustrated in Figure 3.

To facilitate the application of k-means clustering in the RBFDD pre-training phase we apply a *tanh* non-linear activation [44] to the latent representations coming into the RBFDD layer. This ensures that the k-means algorithm is provided with a bounded latent representation, and leads to better model initialization. The pseudo-code for training the D-RBFDD network is described in Algorithm 1. First, the entire training set X is fed into the $LeNet-5$ portion of the D-RBFDD network, in order to produce the pre-RBFDD layer hidden representations, $\hat{X}$. Next, $k-means$ is used to initialize the centers of the Gaussians, μ, and then a heuristic is used to initialize the spreads, s, for the Gaussians. Specifically, for a given Gaussian, the distance of the farthest data point (among all of the hidden representations $\hat{X}$) from the center of that Gaussian is computed and then half of that distance is assigned as the spread, s, for that Gaussian. Euclidean distance is used as the distance metric. Finally, the value of k used in the k-means clustering step is equal to the number of Gaussians H, and needs to be tuned—hyper-parameter tuning is explained in detail in Section IV-B.

In the next step, the entire D-RBFDD network is trained using mini-batch gradient descent for max_epoch epochs. For each mini-batch in X, the final outputs of the D-RBFDD model, $\hat{y}$, are generated; the cost is computed based on Eq.(4); and, using the back-propagation of error algorithm, the gradients are computed and the learnable parameters of the D-RBFDD network (i.e., the centers and spreads of the RBFDD layer and all weights) are updated.

The next section describes the setup of an experiment designed to evaluate the effectiveness of these approaches to deepening the RBFDD network, and to compare this to the effectiveness of current state-of-the-art anomaly detection approaches.

IV. EXPERIMENTAL SETUP

We have designed an experiment to evaluate the performance of different approaches to deepening RBFDD networks, and to compare these to state-of-the-art anomaly detection approaches.[1] We include the following state-of-the-art anomaly detection approaches in this experiment: One-class Deep Support Vector Data Descriptor (DeepSVDD-OC) networks [48], Soft-boundary Deep Support Vector Data Descriptor (DeepSVDD-SB) networks [48], One-Class Support Vector Machines (OCSVMs) [14], Isolation Forest (iForest) [17], RBFDD networks [7], and deep Convolutional

[1] The code for the D-RBFDD network is available on GitHub repository: https://github.com/MLDawn/DRBFDD

Algorithm 1 The Training Algorithm for the D-RBFDD Network

```
Input: $X \leftarrow \{X_1 \ldots X_N\}$ (Training set)
Output: $Trained\ D - RBFDD\ network$
Model = D-RBFDD()
$\hat{X}$ = LeNet-5(X)
$\mu$, s = k-means($\hat{X}$)
for epoch = 1 to max_epoch do
    for mini_batch in $X$ do
        $\hat{y}$ =Model(mini_batch)
        Compute cost using Eq. (4)
        Back-propagate the gradients for Model
        Update the parameters of Model
    end for
end for
return Model
```

Auto Encoders (CAEs). In all cases only *normal* data is used during model training.

To further investigate how effectively representations can be transferred from pre-trained classification networks to anomaly detection tasks, in the case of the OCSVM and iForest models (as well as RBFDD networks), we have also considered the scenario where the latent representation learned by a pre-trained classification network is used as input (i.e., *Fix-Res* + *OCSVM*, *Fix-Res* + *iForest*, and *Fix-Res* + *RBFDD*). We also use the latent representation learned by the version of RBFDD that fine-tunes the pre-trained classification model representation as input to these shallow models to better understand the impact of fine-tuning (i.e., *Fine-Res* + *OCSVM*, *Fine-Res* + *iForest*, and *Fine-Res* + *RBFDD*).

A. DATASETS & ANOMALY DETECTION SCENARIOS

We use two well-known labelled image classification datasets—MNIST [49] and Fashion MNIST [50]—as well as a real-world highly-imbalanced anomaly detection timeseries dataset—the MIT-BIH Arrhythmia Database[2] [51], [52]—in order to explore the effectiveness of the three approaches to deepening RBFDD networks:

- MNIST[3] [49] contains a training set of 60,000 gray-scale handwritten digit images of size 28 × 28 pixels and a test set of 10,000 similar images. The task is to classify the digit present in each image.
- Fashion-MNIST[4] [50] contains a training set of 60,000 gray-scale 28 × 28 pixel images of different clothing items (e.g. t-shirt/top, trouser, pullover, dress) and a test set of 10,000 similar images. The task is to classify the clothing item present in each image.
- MIT-BIH is an arrhythmia detection dataset[5] [51], [52] containing 48 half-hour excerpts of two-channel ambulatory ECG recordings, obtained from 48 subjects. Out of the 19 anomalous classes in the dataset, the 4 most common classes are used to make four binary anomaly detection scenarios. The reason for this is that a lot of the anomalous classes are very infrequent (e.g. less than 100 instances) and so it would not be possible to make a binary classification problem with them that is comparable to the other problems studied in this paper in terms of the frequency with which anomalies appear in the test data. We have also added a *One vs. All* scenario, where examples of the 4 most common anomalous classes are combined into one class. This generates a total of five anomaly detection scenarios for this dataset.

For the MNIST and Fashion-MNIST datasets we generate multiple anomaly detection scenarios using these datasets. In each scenario we consider one class as normal and another class as anomalous. These pairs of classes (shown in Table 1) were selected to cover simple and challenging scenarios. For example, for MNIST we have a simple scenario where digit 0 is considered normal and digit 1 is anomalous, and similarly for Fashion MNIST we have a scenario where *T-shirts/tops* are normal and *boots* are anomalous. Images from these pairs of classes are easily discernible, and we expect to see high performance across most of the algorithms. We also have more challenging scenarios. For instance, from MNIST we have a scenario where digit 4 is normal and digit 9 is anomalous, and for Fashion MNIST we have a scenario where *T-shirts/tops* are normal and *shirts* are anomalous. These pairs of images are not easy to separate as they are so similar.

For the MIT-BIH Arrhythmia dataset, we pre-processed the data to reduce the dimensionality by down-sampling from 360Hz to 187Hz and extracted individual heart-beats,[6] each of which has an associated ground-truth label in the dataset.

In all experiments, models are trained using *only* instances of the normal class. During testing we provide unseen samples from both normal and anomalous classes to measure the performance of the different models. For all datasets, feature values have been normalized to [0, 1]. In particular, the normalization for both the MNIST and Fashion MNIST datasets is done by dividing individual pixel values by 255, as this is the maximum pixel value for these grey-scale images. For the MIT-BIH Arrhythmia dataset, the sample values range from 0 to 2047, both values inclusive, with 1024 as the mid-point. This is due to the fact that, at the digitization step, a resolution of 11-bits has been used, resulting in 2^{11} levels, which are the actual resultant values of the signal in this dataset. Thus, normalization is

[2] https://physionet.org/content/mitdb/1.0.0/

[3] http://yann.lecun.com/exdb/mnist/

[4] https://www.kaggle.com/zalando-research/fashionmnist

[5] https://physionet.org/content/mitdb/1.0.0/

[6] The peak of each heartbeat is labelled in this dataset. Following the approach in [53], we considered the mid-point between every two consecutive peak values to be the border between two consecutive heart-beats. All extracted heart-beats are zero-padded or truncated to a length that is higher than 95% of the extracted heart-beat lengths (417).

TABLE 1. Results for experiments using the MNIST and Fashion MNIST datasets. Each column is labelled N-A, where N = normal class and A = anomalous class. The values in each cell are AUC scores followed by relative rank in parentheses. The average rank per algorithm is given in the last column. The labels for Fashion MNIST are: T: T-shirts/tops, B: Ankle boots, S:Shirts, Sn:Sneakers and Sa:Sandals.

	MNIST						Fashion MNIST				
	0 - 1	7 - 1	4 - 9	7 - 9	9 - 4	9 - 7	T- B	T - S	Sa - Sn	B - Sa	Avg. Rank
iForest	0.9648 (7)	0.7725 (12)	0.6296 (7)	0.7948 (5)	0.7484 (6)	0.7355 (6)	0.9963 (3)	**0.8182** (1)	0.5394 (10)	0.9536 (7)	6.4 (6)
Fix-Res + iForest	0.4795 (14)	0.4608 (14)	0.5904 (12)	0.5939 (14)	0.5595 (14)	0.5941 (14)	0.6212 (14)	0.5614 (13)	0.5304 (11)	0.4782 (14)	13.4 (14)
Fine-Res + iForest	0.7609 (11)	0.9698 (5)	0.5807 (14)	0.6910 (9)	0.6898 (10)	0.6566 (11)	0.9761 (10)	0.4585 (14)	0.4957 (14)	0.7753 (10)	10.8 (12)
OCSVM	0.9962 (3)	0.9623 (6)	0.8320 (2)	**0.9209** (1)	0.9245 (2)	0.9125 (2)	0.9967 (2)	0.7872 (6)	0.5708 (9)	0.9708 (5)	3.8 (3)
Fix-Res + OCSVM	0.5506 (13)	0.6924 (13)	0.5808 (13)	0.6098 (13)	0.6035 (11)	0.6268 (13)	0.8981 (12)	0.5992 (12)	0.5100 (12)	0.6011 (13)	12.5 (13)
Fine-Res + OCSVM	0.8542 (9)	0.9746 (3)	0.5975 (11)	0.7181 (7)	0.7119 (8)	0.6850 (9)	0.9780 (8)	0.6429 (11)	0.5036 (13)	0.7232 (12)	9.1 (9)
RBFDD	**0.9988** (1)	0.9722 (4)	0.7585 (3)	0.8187 (4)	0.8069 (5)	0.8583 (5)	0.9954 (4)	0.7898 (5)	0.6562 (6)	0.9830 (4)	4.1 (4)
Fix-Res + RBFDD	0.7923 (10)	0.8332 (11)	0.6273 (8)	0.6157 (12)	0.6941 (9)	0.6714 (10)	0.9740 (11)	0.6881 (9)	0.6197 (8)	0.7299 (11)	9.9 (11)
Fine-Res + RBFDD	0.9422 (8)	0.9119 (9)	0.6217 (9)	0.6612 (11)	0.7236 (7)	0.7152 (7)	0.9901 (7)	0.8055 (2)	**0.7310** (1)	0.8431 (9)	7.0 (7)
CAE-1	0.7000 (12)	0.8964 (10)	0.7137 (6)	0.8934 (3)	0.5688 (13)	0.6419 (12)	0.8426 (13)	0.6823 (10)	0.6379 (7)	0.8875 (8)	9.4 (10)
CAE-2	0.9914 (5)	0.9514 (7)	0.6110 (10)	0.6614 (10)	0.5843 (12)	0.7139 (8)	0.9764 (9)	0.7378 (8)	0.7013 (4)	0.9647 (6)	7.9 (8)
DeepSVDD-OC	0.9906 (6)	**0.9943** (1)	0.7455 (4)	0.7071 (8)	0.9140 (3)	0.8950 (4)	0.9950 (6)	0.8001 (4)	0.6567 (5)	0.9871 (3)	4.4 (5)
DeepSVDD-SB	0.9957 (4)	0.9916 (2)	0.7365 (5)	0.7417 (6)	0.9132 (4)	0.8969 (3)	0.9951 (5)	0.8020 (3)	0.7023 (3)	**0.9896** (1)	3.6 (2)
D-RBFDD	0.9981 (2)	0.9512 (8)	**0.8450** (1)	0.8971 (2)	**0.9480** (1)	**0.9137** (1)	**0.9987** (1)	0.7459 (7)	0.7161 (2)	0.9887 (2)	**2.7** (1)

TABLE 2. Results for experiments using the MIT-BIH Arrhythmia dataset. The label of each anomalous class is given at the top of the columns (for the label descriptions see the Appendix). The values of each cell are AUC scores followed by the relative rank in parentheses. The average rank per algorithm is given in the last column.

	L	R	V	/	One vs. All	Avg. Rank
iForest	0.5743 (8)	0.7118 (8)	0.6819 (8)	0.7713 (8)	0.6808 (8)	8.0 (8)
OCSVM	0.6684 (5)	0.7582 (6)	0.8647 (5)	0.8591 (6)	0.7830 (5)	5.4 (5)
RBFDD	0.7002 (4)	0.8182 (4)	0.8722 (4)	0.8947 (5)	0.8043 (4)	4.2 (4)
CAE-1	0.6331 (6)	0.7525 (7)	0.7416 (6)	0.8139 (7)	0.7174 (6)	6.4 (7)
CAE-2	0.5994 (7)	0.8139 (5)	0.7407 (7)	0.9395 (1)	0.7023 (7)	5.4 (5)
DeepSVDD-OC	0.7700 (3)	0.8352 (3)	0.9241 (3)	0.9187 (4)	0.8324 (3)	3.2 (3)
DeepSVDD-SB	0.7835 (1)	0.8596 (1)	0.9363 (1)	0.9316 (2)	0.8346 (2)	1.4 (1)
D-RBFDD	0.7723 (2)	0.8458 (2)	0.9361 (2)	0.9261 (3)	0.8507 (1)	2.0 (2)

achieved by dividing individual values in the ECG signals[7] by 2047.

The generated anomaly detection scenarios for all three datasets used are summarized in Table 4 in the Appendix.

B. EXPERIMENTAL DESIGN

To evaluate models we use an approach based on bootstrapping that makes maximum use of the anomalous samples available. For each iteration we randomly select 80% of all normal instances in the dataset (with no replacement) to train the model. The remaining 20% of normal instances is then mixed with all of the anomalous instances to form the test set. We perform each experiment 10 times, and measure the area under the ROC curve (AUC) on the test set, then average the AUC scores over the 10 runs.

We perform hyper-parameter tuning using a grid search that repeats the above process for each hyper-parameter combination. The range of hyper-parameters searched are listed in Table 5 in the Appendix.

We report the best averaged AUC from the grid search for the corresponding experiment. We are aware that reporting the performance of the models with the best set of hyper-parameters over-estimates the generalization performance of the models (known as the problem of *many comparisons in induction algorithms* [54]). However, as our goal is a relative comparison of algorithms, rather than an absolute estimate of generalization error, and all algorithms are evaluated in the same way this is an appropriate evaluation approach that makes better use of limited anomalous samples than measuring performance on a separate hold-out test set.

C. STATE-OF-THE-ART APPROACHES

Each state-of-the-art approach compared in this experiment is tuned to achieve its best possible performance (full details are provided in the Appendix). For all OCSVM models we use Gaussian kernels, as recommended in [2]. The hyper-parameters tuned for OCSVM models are ν, and γ, where ν is the upper bound on the fraction of training errors and a lower bound for the fraction of support vectors, and γ is the kernel coefficient. For iForest, the only hyper-parameter to be tuned is the number of estimators.

To explore their performance we use different CAE architectures, each with similar capacity to the D-RBFDD model. For the image classification datasets CAE-1 has two 2D convolutional layers in the encoder and two transposed 2D convolutional layers in the decoder, while CAE-2 has three convolutional layers in the encoder and three transposed convolutional layers in the decoder. For the ECG

[7] Since, the original mid-point value of 1024 is translated to 0.50 in the normalized space, then 0.50 is the value by which we pad at the end of our signals after segmenting the heart-beats. In terms of truncating, we simply truncate at the end of the signals where the heart-beat has a length of more than 417 (heart-beats that have a length in the top 5% of the heart-beat lengths).

dataset, CAE-1 has two 1D convolutional layers in the encoder and two transposed 1D convolutional layers in the decoder. CAE-2 has the same structure but the second 1D convolutional layer has twice the number of convolutional filters compared to CAE-1. For all CAEs rectified linear activation functions and max-pooling are used at each layer.

For the RBFDD network and the D-RBFDD network, the hyper-parameters that are tuned are the number of Gaussians in the hidden layer, and the coefficients of the cost function (Eq. (4)): β and λ whose values fall in the range of (0, 1]. The D-RBFDD network, is based on the LeNet-5 network architecture [47]. For the ECG dataset we replace the 2D convolutions with 1D convolutions.

For DeepSVDD, a LeNet-type network architecture is used [36] for the image datasets. For the ECG dataset we replace this with the 1D LeNet-5 architecture used in the D-RBFDD network. In both versions of DeepSVDD the weight decay coefficient λ is a tuned hyper-parameter. For DeepSVDD-SB, ν, is also a tunable hyper-parameter, whose role is to control the trade-off between violations of the boundary and the volume of the hyper-sphere. Following the training method in [36], the training of DeepSVDD models also includes a learning rate scheduler that reduces the learning rate by a factor of 10 after a 75% of the specified training epochs have been completed.

In the experiments using the image classification datasets we use a pre-trained ResNet-18 model [31] trained on the ImageNet [55] dataset[8] for transfer learning. No transfer learning is used for the ECG dataset, as reliable large-scale pre-trained models for ECG sensor data are not available.

V. RESULTS

The results of the experiments based on the image classification datasets are detailed in Table 1. These results were achieved using the best hyper-parameter combinations found during the grid search described in Section IV-B (these are listed in Tables 6 and 7 in the Appendix). For each anomaly detection scenario the different approaches have been ranked and the average ranks for each approach are summarized in the last column of each table. These results show the effectiveness of deepening RBFDD for raw datasets, and allow us to compare the three different strategies for deepening described in Section III. The fact that the D-RBFDD model has out-performed the RBFDD model, in the majority of cases, demonstrates the value of using the deep model to generate a latent representation suitable for use by RBFDD. Moreover, it is interesting to note that none of the models that use the latent representation output by the fixed, pre-trained ResNet-18 model out-perform their equivalent models trained on the raw, high-dimensional representation. This is the case for the RBFDD models as well as for the OCSVM and iForest models. This is a reminder of the issue with *mixed approaches* for deep anomaly detection mentioned in Section II. The fixed pre-trained ResNet-18 model has been trained for a multi-class classification objective and the latent representations generated by this network seem to be too entangled with that task to be very useful for anomaly detection.

This is further underlined by the fact that, in almost all cases, the performance of the models (RBFDD, OCSVM, and iForest) using latent representations produced by the fine-tuned ResNet-18 (i.e., *Fine-Res* + *OCSVM*, *Fine-Res* + *iForest*, and *Fine-Res* + *RBFDD*) improves over the versions of the models trained using the representations from the fixed ResNet-18 model (i.e., *Fix-Res* + *OCSVM*, *Fix-Res* + *iForest*, and *Fix-Res* + *RBFDD*). This fine-tuning is done using the RBFDD network. However, it is important to note that in most cases this performance was still not better than the models using raw data (i.e., OCSVM, iForest and RBFDD).

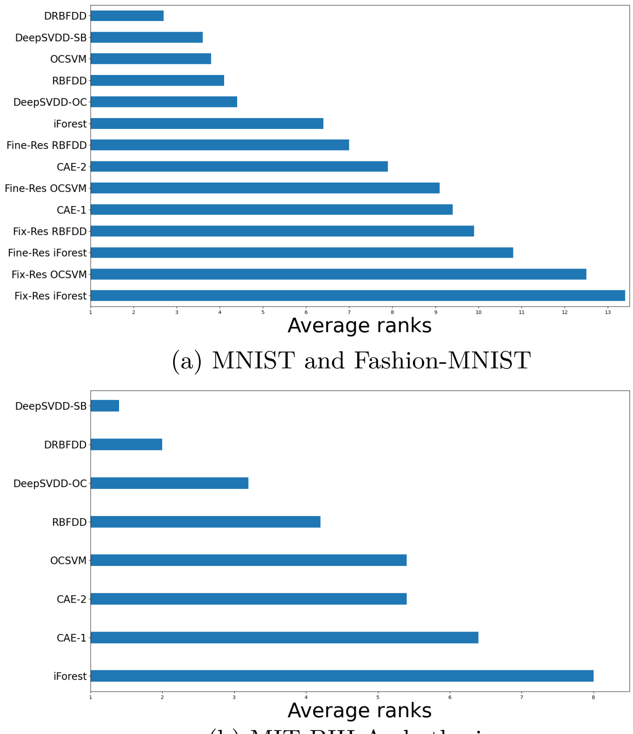
Together these results show that deepening RBFDD networks allows them to work effectively with raw inputs, and that the D-RBFDD approach is the best way to do this out of those compared. This conclusion is reinforced by the results based on the ECG dataset shown in Table 2. In experiments using this dataset D-RBFDD outperforms RBFDD in all cases.

By examining the results for the image classification datasets in Table 1 and those based on the ECG dataset in Table 2 together we can evaluate how D-RBFDD compares to other state-of-the-art anomaly detection algorithms. In the image classification dataset cases, the results show that, overall, the D-RBFDD network out-performs the other algorithms as it has the lowest average rank (lower ranks are better). On the ECG dataset DeepSVDD-SB has a slightly better average rank than D-RBFDD, although D-RBFDD performs better in the *One vs. All* scenario which is particularly important for anomaly detection as it is likely that anomalies will arise from very different data distributions. In order to facilitate the comparison between the benchmark algorithms, the average ranks reported in Table 1 and Table 2, are also sorted and visualised using bar charts in Figure 4.

To further investigate and understand the overall differences between the performances of the different algorithms and the effectiveness of D-RBFDD, we perform non-parametric statistical significance tests for multiple classifier comparison. Following [56] a Friedman test followed by a Finner p-value correction was performed on the results in Tables 1 and 2. This test analyzes the difference in performance between each pair of algorithms with respect to the different anomaly detection scenarios.

The statistical test results for the image classification and ECG datasets are summarized in the critical difference plots (with significance level of $\alpha = 0.05$) in Figure 5a and Figure 5b respectively. Two algorithms not connected with bold horizontal lines are significantly different. The p-values of the statistical tests and the pairwise win/lose/tie results are provided in in Tables 9 and 10 in the Appendix.

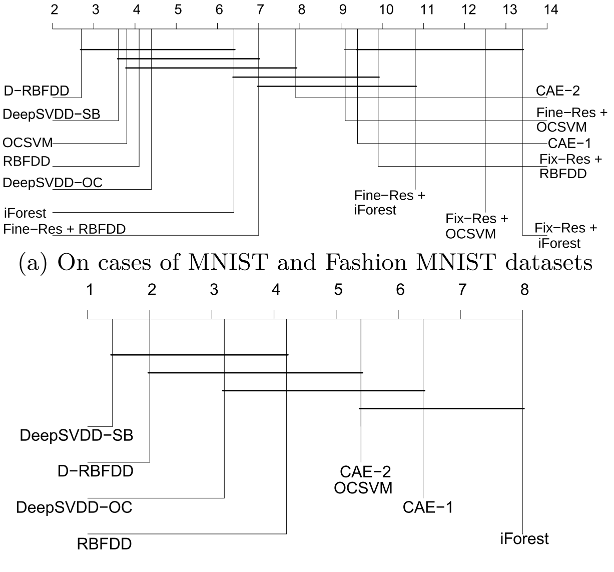
[8] The ResNet-18 implementation used is available at: https://github.com/pytorch/vision/tree/master/torchvision/\\models/resnet.py



(a) MNIST and Fashion-MNIST

(b) MIT-BIH Arrhythmia

FIGURE 4. **Average ranks (lower is better) of the benchmark algorithms across the MNIST, Fashion-MNIST and MIT-BIH Arrhythmia datasets.**



(a) On cases of MNIST and Fashion MNIST datasets

(b) On cases of MIT-BIH Arrhythmia dataset

FIGURE 5. **Critical difference plots from a Friedman test using a significance level of 0.05 on the different anomaly detection scenarios. Algorithms *not* connected with horizontal bars are significantly different.**

Figure 5a shows that D-RBFDD performs significantly and consistently better than the following algorithms: Fine-Res + RBFDD, CAE-2, Fine-Res + OCSVM, CAE-1, Fix-Res + RBFDD, Fix-Res + OCSVM, Fix-Res + iForest, Fine-Res + iForest. In the case of DeepSVDD-SB, OCSVM, RBFDD, DeepSVDD-OC and iForest, the null-hypothesis of the test could not be rejected with a significance level of $\alpha = 0.05$, but D-RBFDD performed better in average rank. On the other hand, in a simple and direct pairwise win/lose/tie based comparison, D-RBFDD won in at least 70% and up to 100% of the anomaly detection cases when compared to the other algorithms (see Appendix). This indicates that D-RBFDD typically performs as good as or better than the benchmark algorithms it is compared to.

From Figure 5b we can see that DeepSVDD-SB attained the best average rank of 1.4 on the ECG dataset. D-RBFDD achieved similar performance with an average rank of 2.0. In most scenarios DeepSVDD-SB has performed slightly better than D-RBFDD, but, interestingly D-RBFDD achieved the best performance in the *One vs. All* case. Although, from an overall comparison of DeepSVDD-SB and D-RBFDD, the null hypothesis could not be rejected. D-RBFDD performed better than iForest and CAE-1 at the significance level of $\alpha = 0.01$ and $\alpha = 0.05$ respectively, and performed better than OCSVM and CAE-2 with a significance level of $\alpha = 0.1$.

Overall these results indicate that adding extra computational layers to RBFDD makes it a much more effective anomaly detector for problems with raw low-level data representations. Also, selecting D-RBFDD will lead to at least similar or better performance than the other approaches, making it an attractive solution for anomaly detection for these types of datasets. We believe that this strong performance, coupled with the easy interpretability and adaptability of approaches based on RBF networks make D-RBFDD a compelling approach.

In order to gain some insight into the convergence of the D-RBFDD network, the changes in cost as model training progresses through training epochs, for each anomaly detection scenario for all three datasets are plotted in Figure 6 (the best set of hyper-parameters given in Tables 6, 7 and 8 in the Appendix are used for each model). For instance, for all six scenarios in the MNIST dataset after nearly 800 epochs, D-RBFDD models are converging. Whereas, in the case of Fashion MNIST, across all four scenarios, D-RBFDD models converge after around 200 epochs. In the case of MIT-BIH Arrhythmia dataset, the same normal dataset is used for all scenarios so only a single line is shown. These plots illustrate the rate with which training converges and how well it converges in the different cases—in all cases the D-RBFDD network learns quickly during the initial training epochs.

Finally, to gain some insight regarding the comparative computational complexity of the models used we report the training time required for the different approaches.[9] We

[9] To do this an approach that can be applied to all methods in our experiments is required for a fair comparison. One could look at this in terms of number of trainable parameters, however, it is not possible to talk about the number of trainable parameters for the iForest and OCSVM models in a meaningful way. Hence, for a fair comparison training time is used.

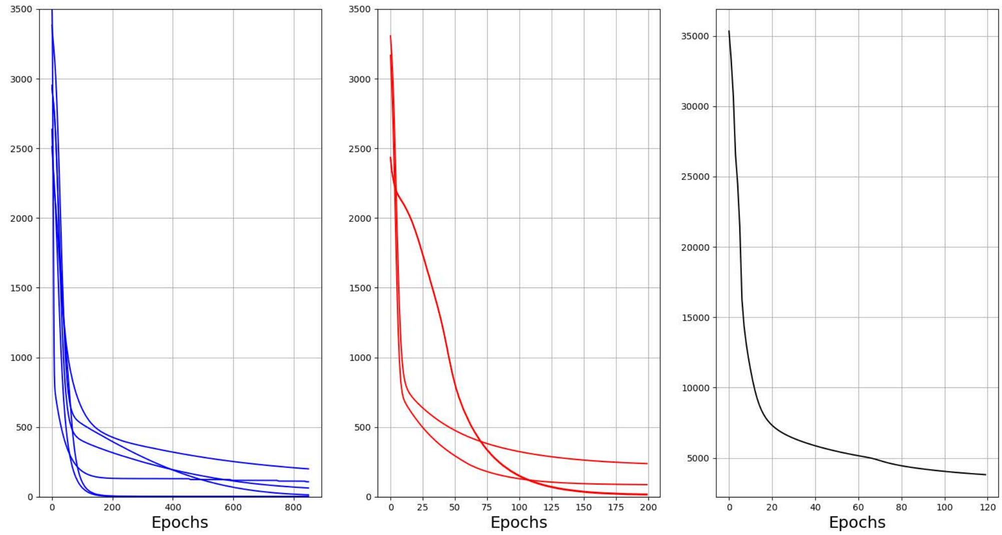


FIGURE 6. **The convergence behaviour of D-RBFDD: six scenarios of MNIST (left), four scenarios of Fashion MNIST (middle) and the MIT-BIH dataset (right).**

TABLE 3. **Total training time, in seconds, of the models with their best performing hyper-parameters.**

	MNIST 7-9	MIT-BIH Arrhythmia One-vs-All
iForest	7.78s	201.37s
OCSVM	6.02s	36.33s
RBFDD	264.00s (K-means 14s + 0.50 s/epoch × 500 epochs)	50.76s (K-means 0.76s + 2.50 s/epoch × 20 epochs)
CAE-1	25.00s (0.25 s/epoch × 100 epochs)	70.60s (3.53 s/epoch × 20 epochs)
CAE-2	26.00s (0.26 s/epoch × 100 epochs)	1825.00s (3.65 s/epoch ×k 500 epochs)
DeepSVDD-OC	29.00s (Centre initialisation 3s + 0.26 s/epoch × 100 epochs)	295.00s (Centre initialisation 10s + 2.85 s/epoch × 100 epochs)
DeepSVDD-SB	30.00s (Centre initialisation 3s + 0.27 s/epoch × 100 epochs)	298.00s (Centre initialisation 10s + 2.88 s/epoch × 100 epochs)
D-RBFDD	31.15s (K-means 5.15s + 0.26 s/epoch × 100 epochs)	69.52s (K-means 12.12s + 2.87 s/epoch × 20 epochs)

report training times[10] for one scenario using the MNIST dataset and one scenario using the MIT-BIH Arrhythmia dataset in Table 3. For each approach, models are trained using the best performing hyper-parameters found using grid search. Since the pre-trained ResNet-18 is not applicable in the MIT-BIH Arrhythmia dataset, we will not include the models based on ResNet-18 here. For all artificial neural networks, the total training time, as well as, the training time per epoch are reported. The time for computing the centres of the Gaussians in the case of D-RBFDD, and the time for computing the centres in the case of DeepSVDD-OC and DeepSVDD-SB are also reported. Finally, The total number of epochs are set based on the best performing value per model.

As shown in Table 3, in particular, in the case of D-RBFDD, DeepSVDD-OC and DeepSVDD-SB, it can be seen that the per epoch training times are very close. In the MIT-BIH Arrhythmia dataset, both DeepSVDD-OC and DeepSVDD-SB achieve their best performance after 100 epochs, whereas DRBFDD does so after 20 epochs, and hence the big gap in their total training time (i.e., 295.0s, 298.0s, and 69.52s respectively).

VI. DISCUSSION

The experimental results in the previous section show that transfer learning using a pre-trained ResNet-18 with fixed weights, does not work well for anomaly detection. We believe that the reason for this is that the ResNet-18 model has been trained for a multi-class classification task and that the latent representations that it generates are too entangled with that task to be useful for anomaly detection. Interestingly, we see that if the final layers of the ResNet-18 model are fine-tuned using the RBFDD network cost function, the performance improves in most cases. Overall, it can be concluded that, selecting D-RBFDD for an anomaly detection task on raw data would lead to performance that is at least as good as or significantly better than current state-of-the-art algorithms.

In addition, on the image datasets, the D-RBFDD network has shown superior performance to state-of-the-art one-class classifiers—DeepSVDD, OCSVM, iForest, and CAEs. For the experiments using the ECG dataset, the D-RBFDD network has produced comparable results to those of the DeepSVDD-SB algorithm, and out-performed it in the *One*

[10] The reported training times in Table 3 are generated using an identical machine (spec: Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz 16GB) in order to make sure they are comparable with one another.

vs. All scenario, which is particularly important for anomaly detection as it is likely that anomalies will arise from very different data distributions. We have also observed that the proposed D-RBFDD model, has indeed out-performed its shallower version, the RBFDD network, in almost all of our benchmark experiments. This suggests that, when dealing with raw data we have benefited from the introduction of depth in the D-RBFDD network. It should also be noted that, unlike some of the state-of-the-art algorithms, in particular OCSVMs, D-RBFDD networks are scalable and can work with large datasets and high-dimensional data. This indicates that the D-RBFDD network is an attractive option for the task of anomaly detection.

It is also worth reflecting on some of the limitations of the D-RBFDD network. The first limitation is the sensitivity of D-RBFDD networks to the number of Gaussian kernels—the addition or removal of kernels can dramatically affect model performance. Thus, the number of Gaussians is a hyper-parameter that needs to be carefully fine-tuned.

Finally, the current version of D-RBFDD networks requires pre-training (i.e., K-means clustering), which initialises the Gaussian kernels in the hidden space. It would be desirable to eliminate this step and allow for a seamless end-to-end training without any need for pre-training.

VII. CONCLUSION

In this article, we have proposed a deep one-class neural network, the D-RBFDD network, that adds convolutional layers before an RBFDD network. The D-RBFDD network is trained in an end-to-end fashion on an objective that is designed specifically for anomaly detection. We have shown that this network has successfully turned the shallow RBFDD network into a deep one-class classifier, suitable for anomaly detection on high-dimensional raw data such as images and sensor data. Our experiments also show that transfer learning using a pre-trained ResNet-18 with fixed weights, does not work well for anomaly detection.

Hence, the main contributions of our work are two-fold: (1) we propose an effective anomaly detection approach, Deep RBFDD, for deepening RBFDD networks for the task of anomaly detection, suitable for learning from low-level, raw data representations; and (2) we show that the latent representations learned by large models trained for multi-class classification are not suitable as input for anomaly detection models.

VIII. FUTURE WORK

D-RBFDD shows significant improvement in performance over its predecessor RBFDD, and competitive performance with state-of-the-art anomaly detection algorithms. This is a pre-requisite for broadening the application of such networks to more challenging scenarios such as learning from streams of incoming data, where the main challenge is the dynamic nature of what constitutes normal and anomalous. Applying D-RBFDD to such evolving scenarios seems reasonable as here we have the ability to control the number of Gaussians, which allows a high degree of adaptability for scenarios where concept drift is a concern and the definition of normal can change over time. Thus, by adding/removing/replacing Gaussians, the D-RBFDD network could learn a variety of new emerging contexts as well as forget expired ones. As a result, in the future, we plan to exploit the flexibility of D-RBFDD to adapt it for an on-line learning scenario where detection and handling concept drift in the incoming stream of data is important. We will explore approaches to allow the D-RBFDD network, to self-expand and prune to adapt to the appearance or disappearance of concepts. We will also explore the interpretability of D-RBFDD networks, since they have the potential to provide explanations as to why an input is flagged as an anomaly. The features learned by the RBFDD component in the D-RBFDD network (i.e., centers and spreads of Gaussian kernels and associated weights) provide us with a level of interpretability that has the potential to be quite informative in terms of understanding the model learned and the reasoning behind flagging anomalies.

APPENDIX

IX. EXPERIMENT SCENARIOS

We have used three datasets: two classification datasets (MNIST and Fashion MNIST), and a highly imbalanced anomaly detection dataset (MIT-BIH Arrhythmia). In Table 4, we have summarized the datasets and the scenarios we have used for our experiments. In the case of MNIST and Fashion MNIST, we use instances of a certain class as normal, and instances of another class as anomalous. The MIT-BIH Arrhythmia dataset has a normal class and 19 anomalous classes, of which we have used the top four common anomalous classes in our experiments.

X. HYPER-PARAMETERS USED

We have performed extensive hyper-parameter tuning for our experiments using grid search. Table 5 summarizes the set of hyper-parameters and the explored values for tuning these hyper-parameters for each algorithm.

XI. BEST HYPER-PARAMETERS FOUND

Tables 6, 7, and 8 report the best hyper-parameters found for each algorithm, for each experiment scenario. Each row shows an algorithm and the a list of values for its relevant tuned hyper-parameters. The order of the hyper-parameter values for each algorithm is as follows:

- For RBFDD, Fix-Res + RBFDD, Fine-Res + RBFDD, and D-RBFDD: The order of the best hyper parameters is: (H, number of Epochs, η, β, λ)
- For OCSVM the order of the best hyper parameters is: (ν, γ)
- For CAE-1/CAE-2: (number of epochs, η)
- for DeepSVDD-OC and DeepSVDD-SB the order of best hyper-parameters is: (η, number of epochs, λ) and (η, number of epochs, λ, ν), respectively.

For example in Table 6, the hyper-parameters for D-RBFDD are H $=$ 21, number of Epochs $=$ 100, $\eta = 10e-5$, $\beta = 0.01$, $\lambda = 0.5$

TABLE 4. The normal and anomalous classes (descriptions in the parentheses) for each dataset. We have six experiment scenarios for MNIST, four for Fashion MNIST, and four for MIT-BIH Arrhythmia.

Dataset	Normal	Anomalous
MNIST	0	1
	7	1
	4	9
	7	9
	9	4
	9	7
Fashion-MNIST	T (T-shirts/tops)	B (Ankle boots)
	T (T-shirts/tops)	S (Shirts)
	Sa (Sandals)	Sn (Sneakers)
	B (Ankle boots)	Sa (Sandals)
MIT-BIH Arrhythmia Database	N (Normal)	L (Left bundle branch block beat)
	N (Normal)	R (Right bundle branch block beat)
	N (Normal)	V (Premature ventricular contraction)
	N (Normal)	/ (Paced beat)

TABLE 5. The hyper-parameter ranges explored for each algorithm, where D denotes the dimensionality of the input data to the RBFDD layer in both the RBFDD network and the D-RBFDD network. η is the learning rate and the other hyper-parameters are as explained in the main body of the article.

Algorithms	Hyper-parameters	Investigated Hyper-Parameters
RBFDD	number of kernels (H)	[0.01D, 0.05D, ..., 0.25D, 0.50D, 0.75D, 2D]
	η	[0.01, 0.001, 0.0001]
	β	[0.9, 0.5, 0.01, 0.0001]
	λ	[0.9, 0.5, 0.01, 0.0001]
	epochs	[20, 100, 500]
D-RBFDD	number of kernels (H)	[0.25D, 0.50D, 0.75D, 2D]
	η	[0.01, 0.001, 0.0001]
	β	[0.9, 0.5, 0.01, 0.0001]
	λ	[0.9, 0.5, 0.01, 0.0001]
	epochs	[20, 100, 500]
DeepSVDD	η	[0.01, 0.0001, 0.00001]
	learning rate milestone for the scheduler	0.75
	λ	[0.9, 0.5, 0.01, 0.0001]
	ν (Only for Soft-Boundary)	[0.9, 0.5, 0.01, 0.0001]
	epochs	[20, 100, 500]
OCSVM	ν	[0.9, 0.5, 0.01, 0.0001]
	γ	[0.9, 0.5, 0.01, 0.0001]
CAE-1/CAE-2	η	[0.01, 0.001, 0.0001]
	error functions	mean squared error, cross-entropy
	epochs	[20, 100, 500]
Isolation Forest	number of estimators	[100, 200, 500, 800, 1000]

XII. MULTIPLE CLASSIFIER TESTING

For multiple classifier comparison, a Friedman test followed by a Finner p-value correction was performed. The p-values of the result is shown in Table 9 for the MNIST and Fashion MNIST datasets, and in Table 10 for results for the MIT-BIH Arrhythmia dataset.

In Tables 9 and 10 the lower diagonal shows the p-values of the post-hoc Friedman test (with the Finner p-value correction) with the corresponding significance level with which the null-hypothesis can be rejected. Critical difference plots with a significance level of $\alpha = 0.05$ from the results in Tables 9 and 10 are shown in Figures 5a and 5b respectively. The scales above each critical difference plot are the average ranks of the corresponding algorithms. The algorithms which are not connected with horizontal bars are significantly different with the significance level of $\alpha = 0.05$. For the algorithms connected with the horizontal lines, the null-hypothesis of the Friedman test could not be rejected with the given significance level.

A simple win/lose/tie count with respect to datasets for each pair of algorithms is shown in the upper diagonal of Table 9 and 10. For example in Table 9, when comparing DeepSVDD-SB and D-RBFDD, we see the value (7/3/0). This means that D-RBFDD had better scores in 7 cases and worse scores in 3 cases out of a total of 10 cases.

TABLE 6. Best hyper-parameter combinations found for MNIST using grid search.

	0 - 1	7 - 1	4 - 9	7 - 9	9 - 4	9 - 7
iForest	800	100	1000	200	200	200
OCSVM	0.0001,0.01	0.0001,0.01	0.0001, 0.5	0.0001, 0.5	0.01, 0.9	0.01, 0.9
CAE-1	20, 1e-05	20, 0.0001	500, 0.01	100, 0.01	500,0.01	20,0.00001
CAE-2	20, 1e-05	20, 1r-05	20, 0.01	100,0.01	20, 0.01	20, 1e-05
RBFDD	7, 500, 0.01, 0.0001, 0.0001	7, 500, 0.01, 0.0001, 0.0001	7, 100, 0.01, 0.0001, 0.01	39, 500, 1e-05, 0.5, 0.0001	39, 100, 0.0001, 0.9, 0.0001	39, 20, 0.01, 0.0001, 0.0001
Fix-Res + RBFDD	51, 20, 0.01, 0.01, 0.9	5, 20, 0.01, 0.01, 0.5	25, 100, 0.0001, 0.01, 0.0001	5, 20, 0.01, 0.0001, 0.5	5, 20, 0.01, 0.5, 0.9	51, 500, 1e-05, 0.9, 0.0001
Fine-Res + RBFDD	25, 100, 0.0001, 0.5, 0.5	25, 500, 1e-05, 0.5, 0.01	25, 100, 1e-05, 0.5, 0.9	25, 20, 0.0001, 0.0001, 0.9	51, 20, 0.0001, 0.01, 0.01	5, 500, 1e-05, 0.5, 0.5
DeepSVDD-OC	0.0001, 500, 0.0001	0.01, 500, 0.9	0.01, 20, 0.5	1e-05, 100, 0.0001	0.01, 100, 0.0001	0.01, 20, 0.0001
DeepSVDD-SB	0.0001, 500, 0.0001, 0.9	0.01, 100, 0.5, 0.9	0.01, 20, 0.9, 0.9	0.01, 100, 0.5, 0.9	0.01, 20, 0.0001, 0.5	0.01, 20, 0.0001, 0.5
D-RBFDD	21, 100, 1e-05, 0.01, 0.5	42, 500, 0.0001, 0.0001, 0.9	21, 500, 1e-05, 0.0001, 0.0001	63, 100, 1e-05, 0.9, 0.0001	63, 500, 1e-05, 0.0001, 0.5	21, 100, 1e-05, 0.01, 0.0001

TABLE 7. Best hyper-parameter combinations found for Fashion MNIST using grid search.

	T(0) - B(9)	T(0) - S(6)	Sa(5) - Sn(7)	B(9) - Sa(5)
iForest	200	200	1000	500
OCSVM	0.9,0.5	0.01,0.9	0.01, 0.9	0.9, 0.0001
CAE-1	20,0.01	500,0.01	500, 0.00001	500, 0.0001
CAE-2	500, 0.0001	500,0.01	500, 1e-05	500, 1e-05
RBFDD	39, 500, 0.0001, 0.5, 0.0001	39, 20, 0.01, 0.9, 0.5	39, 100, 1e-05, 0.01, 0.0001	39, 500, 0.0001, 0.5, 0.0001
Fix-Res + RBFDD	51, 20, 0.01, 0.9, 0.5	76, 20, 1e-05, 0.0001, 0.9	25, 500, 0.0001, 0.5, 0.5	20, 1e-05, 0.0001, 0.9
Fine-Res + RBFDD	51, 500, 0.0001, 0.9, 0.01	76, 20, 0.01, 0.01, 0.9	76, 100, 0.01, 0.01, 0.5	7, 1e-05, 0.0001, 0.9
DeepSVDD-OC	0.0001, 500, 0.0001	0.01, 100, 0.0001	0.01, 100, 0.5	0.0001, 500, 0.0001
DeepSVDD-SB	0.0001, 500, 0.0001, 0.5	0.01, 100, 0.0001, 0.9	0.01, 20, 0.01, 0.0001	0.01, 100, 0.0001, 0.9
D-RBFDD	42, 100, 1e-05, 0.01, 0.01	42, 20, 1e-05, 0.0001, 0.01	63, 100, 0.0001, 0.01, 0.9	21, 500, 0.0001, 0.0001, 0.9

TABLE 8. Best hyper-parameter combinations found for MIT-BIH Arrhythmia dataset using grid search.

	L	R	V	/	One vs. All
iForest	1000	800	500	1000	800
OCSVM	0.01, 0.5	0.01, 0.5	0.01, 0.9	0.01, 0.5	0.01, 0.5
RBFDD	37, 20, 0.0001, 0.01, 0.0001	74, 20, 0.0001, 0.0001, 0.01	74, 20, 0.0001, 0.0001, 0.01	18, 500, 1e-05, 0.0001, 0.01	37, 20, 0.0001, 0.01, 0.01
CAE-1	0.0001, 20, ’MSE’	0.0001, 500, ’MSE’	0.0001, 20, ’MSE’	0.01, 100, ’MSE’	0.0001, 20, ’MSE’
CAE-2	1e-05, 20, ’MSE’	0.01, 500, ’MSE’	1e-05, 20, ’MSE’	0.01, 500, ’MSE’	0.01, 500, ’MSE’
DeepSVDD-OC	1e-05, 20, 0.0001	1e-05, 100, 0.0001	0.01, 20, 0.0001	0.01, 20, 0.0001	1e-05, 100, 0.0001
DeepSVDD-Soft-boundary	1e-05, 20, 0.0001, 0.5	1e-05, 100, 0.0001, 0.5	0.0001, 100, 0.0001, 0.5	0.01, 20, 0.0001, 0.9	1e-05, 100, 0.0001, 0.5
DRBFDD	16, 20, 1e-05, 0.01, 0.9	63, 20, 1e-05, 0.5, 0.0001	63, 20, 1e-05, 0.9, 0.5	63, 20, 1e-05, 0.9, 0.01	12, 20, 1e-05, 0.01, 0.5

TABLE 9. Result of Friedman test with Finner p-value adjustment and pairwise win/lose/tie counts over the different cases for the MNIST and Fashion MNIST datasets. Upper diagonal: win/lose/tie. Lower diagonal: Friedman test with p-values after the Finner adjustment. * $\alpha = 0.1$, ** $\alpha = 0.05$ and *** : $\alpha = 0.01$.

	D-RBFDD	DeepSVDD-SB	OCSVM	RBFDD	DeepSVDD-OC	iForest	Fine-Res + RBFDD	CAE-2	Fine-Res + OCSVM	CAE-1	Fix-Res + RBFDD	Fine-Res + iForest	Fix-Res + OCSVM	Fix-Res + iForest
D-RBFDD		7/3/0	7/3/0	7/3/0	8/2/0	9/1/0	8/2/0	9/1/0	9/1/0	10/0/0	10/0/0	9/1/0	10/0/0	10/0/0
DeepSVDD-SB	0.6870		4/6/0	6/4/0	7/3/0	7/3/0	8/2/0	10/0/0	10/0/0	9/1/0	10/0/0	10/0/0	10/0/0	10/0/0
OCSVM	0.6175	0.9149		5/5/0	6/4/0	9/1/0	8/2/0	9/1/0	9/1/0	9/1/0	9/1/0	9/1/0	10/0/0	10/0/0
RBFDD	0.5251	0.8075	0.8812		4/6/0	8/2/0	8/2/0	9/1/0	9/1/0	9/1/0	10/0/0	10/0/0	10/0/0	10/0/0
DeepSVDD-OC	0.4489	0.7068	0.7758	0.8812		7/3/0	8/2/0	8/2/0	9/1/0	9/1/0	10/0/0	10/0/0	10/0/0	10/0/0
iForest	0.0926 *	0.2028	0.2387	0.2924	0.3660		8/2/0	6/4/0	9/1/0	6/4/0	8/2/0	9/1/0	10/0/0	10/0/0
Fine-Res + RBFDD	0.0495 **	0.1223	0.1475	0.1923	0.2387	0.7758		6/4/0	8/2/0	7/3/0	9/1/0	8/2/0	10/0/0	10/0/0
CAE-2	0.0149 **	0.0495 **	0.0605 *	0.0854 *	0.1154	0.5006	0.6870		6/4/0	8/2/0	8/2/0	7/3/0	9/1/0	10/0/0
Fine-Res + OCSVM	0.0032 ***	0.0110 **	0.0135 **	0.0200 **	0.0300 **	0.2202	0.3418	0.5860		5/5/0	6/4/0	9/1/0	9/1/0	9/1/0
CAE-1	0.0019 ***	0.0073 ***	0.0096 ***	0.0135 **	0.0200 **	0.1764	0.2713	0.5006	0.8812		5/5/0	5/5/0	8/2/0	10/0/0
Fix-Res + RBFDD	0.0009 ***	0.0033 ***	0.0046 ***	0.0073 ***	0.0110 **	0.1154	0.1923	0.3660	0.7068	0.8075		6/4/0	10/0/0	10/0/0
Fine-Res + iForest	0.0001 ***	0.0009 ***	0.0012 ***	0.0019 ***	0.0032 ***	0.0441 **	0.0854 *	0.1923	0.4489	0.5251	0.6870		7/3/0	7/3/0
Fix-Res + OCSVM	0.0000 ***	0.0000 ***	0.0000 ***	0.0001 ***	0.0001 ***	0.0046 ***	0.0110 **	0.0339 **	0.1223	0.1615	0.2387	0.4489		8/2/0
Fix-Res + iForest	0.0000 ***	0.0000 ***	0.0000 ***	0.0000 ***	0.0000 ***	0.0012 ***	0.0032 ***	0.0110 **	0.0495 **	0.0676*	0.1154	0.2387	0.6870	

TABLE 10. Result of Friedman test with Finner p-value adjustment and pairwise win/lose/tie counts over the different cases for the MIT-BIH Arrhythmia dataset. Upper diagonal: win/lose/tie. Lower diagonal: Friedman test with p-values after the Finner adjustment. * $\alpha = 0.1$, ** $\alpha = 0.05$ and *** $\alpha = 0.01$.

	DeepSVDD-SB	D-RBFDD	DeepSVDD-OC	RBFDD	OCSVM	CAE-2	CAE-1	iForest
DeepSVDD-SB		4/1/0	5/0/0	5/0/0	5/0/0	4/1/0	5/0/0	5/0/0
D-RBFDD	0.7116		5/0/0	5/0/0	5/0/0	4/1/0	5/0/0	5/0/0
DeepSVDD-OC	0.3395	0.5369		5/0/0	5/0/0	4/1/0	5/0/0	5/0/0
RBFDD	0.1573	0.2707	0.5738		5/0/0	4/1/0	5/0/0	5/0/0
OCSVM	0.0450 **	0.0851 *	0.2707	0.5369		3/2/0	5/0/0	5/0/0
CAE-2	0.0450 **	0.0851 *	0.2707	0.5369	1.0000		2/3/0	5/0/0
CAE-1	0.0116 **	0.0250 **	0.0960 *	0.2707	0.5738	0.5738		5/0/0
iForest	0.0006 ***	0.0015 ***	0.0135 **	0.0487 **	0.1902	0.1902	0.3951	

REFERENCES

[1] V. Chandola, A. Banerjee, and V. Kumar, “Anomaly detection: A survey,” *ACM Comput. Surv.*, vol. 41, no. 3, pp. 1–58, 2009.

[2] S. S. Khan and M. G. Madden, “One-class classification: Taxonomy of study and review of techniques,” *CoRR*, vol. abs/1312.0049, pp. 1–35, Nov. 2013.

[3] C. Zhou and R. C. Paffenroth, “Anomaly detection with robust deep autoencoders,” in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2017, pp. 665–674.

[4] E. Rushe and B. M. Namee, “Anomaly detection in raw audio using deep autoregressive networks,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 3597–3601.

[5] S. Ahmad, A. Lavin, S. Purdy, and Z. Agha, “Unsupervised real-time anomaly detection for streaming data,” *Neurocomputing*, vol. 262, pp. 134–147, Nov. 2017.

[6] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[7] M. H. Bazargani and B. M. Namee, “Radial basis function data descriptor (RBFDD) network: An anomaly detection approach,” in *Proc. ODD V5.0 Workshop, Outlier Detection De-Constructed*, London, U.K., 2018.

[8] E. Alpaydin, *Introduction to Machine Learning*. Cambridge, MA, USA: MIT Press, 2004.

[9] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2007.

[10] Y. Jin and B. Sendhoff, “Extracting interpretable fuzzy rules from RBF networks,” *Neural Process. Lett.*, vol. 17, no. 2, pp. 149–164, 2003.

[11] Z. Xi and G. Panoutsos, “Interpretable machine learning: Convolutional neural networks with RBF fuzzy logic classification rules,” in *Proc. Int. Conf. Intell. Syst. (IS)*, Sep. 2018, pp. 448–454.

[12] M. F. Augusteijn and K. Shaw, “Constructing a query facility for RBF networks,” in *Proc. Mex. Int. Conf. Artif. Intell.* Acapulco, Mexico: Springer, Apr. 2000, pp. 376–388.

[13] P. Lindstrom, S. J. Delany, and B. M. Namee, “Handling concept drift in a text data stream constrained by high labelling cost,” in *Proc. 23rd Int. FLAIRS Conf.*, 2010, pp. 1–6.

[14] B. Schölkopf, J. C. Platt, J. C. Shawe-Taylor, A. J. Smola, and R. C. Williamson, “Estimating the support of a high-dimensional distribution,” *Neural Comput.*, vol. 13, no. 7, pp. 1443–1471, 2001.

[15] M. Awad and R. Khanna, *Support Vector Machines for Classification*. Berkeley, CA, USA: Apress, 2015, pp. 39–66, doi: 10.1007/978-1-4302-5990-9_3.

[16] D. M. J. Tax and R. P. W. Duin, “Support vector data description,” *Mach. Learn.*, vol. 54, no. 1, pp. 45–66, Jan. 2004, doi: 10.1023/B:MACH.0000008084.60811.49.

[17] F. T. Liu, K. M. Ting, and Z.-H. Zhou, “Isolation forest,” in *Proc. 8th IEEE Int. Conf. Data Mining*, Dec. 2008, pp. 413–422.

[18] J. An and S. Cho, “Variational autoencoder based anomaly detection using reconstruction probability,” *Special Lecture IE*, vol. 2, no. 1, pp. 1–18, Dec. 2015.

[19] E. Marchi, F. Vesperini, F. Eyben, S. Squartini, and B. Schuller, “A novel approach for automatic acoustic novelty detection using a denoising autoencoder with bidirectional LSTM neural networks,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, South Brisbane, QLD, Australia, Apr. 2015, pp. 1996–2000, doi: 10.1109/ICASSP.2015.7178320.

[20] Y. Zhao, B. Deng, C. Shen, Y. Liu, H. Lu, and X.-S. Hua, “Spatio-temporal autoencoder for video anomaly detection,” in *Proc. 25th ACM Int. Conf. Multimedia*, Oct. 2017, pp. 1933–1941.

[21] Y. Xiong and R. Zuo, “Recognition of geochemical anomalies using a deep autoencoder network,” *Comput. Geosci.*, vol. 86, pp. 75–82, Jan. 2016.

[22] W. Yan and L. Yu, “On accurate and reliable anomaly detection for gas turbine combustors: A deep learning approach,” in *Proc. Annu. Conf. Prognostics Health Manage. Soc.*, 2015.

[23] Y. Wang, W.-D. Cai, and P.-C. Wei, “A deep learning approach for detecting malicious Javascript code,” *Secur. Commun. Netw.*, vol. 9, no. 11, pp. 1520–1534, Jul. 2016.

[24] S. M. Erfani, S. Rajasegarar, S. Karunasekera, and C. Leckie, “High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning,” *Pattern Recognit.*, vol. 58, pp. 121–134, Oct. 2016.

[25] N. Marir, H. Wang, G. Feng, B. Li, and M. Jia, “Distributed abnormal behavior detection approach based on deep belief network and ensemble SVM using spark,” *IEEE Access*, vol. 6, pp. 59657–59671, 2018.

[26] D. Xu, Y. Yan, E. Ricci, and N. Sebe, “Detecting anomalous events in videos by learning deep representations of appearance and motion,” *Comput. Vis. Image Understand.*, vol. 156, pp. 117–127, Mar. 2017.

[27] A. Chen, Y. Fu, X. Zheng, and G. Lu, “An efficient network behavior anomaly detection using a hybrid DBN-LSTM network,” *Comput. Secur.*, vol. 114, Mar. 2022, Art. no. 102600, doi: 10.1016/j.cose.2021.102600.

[28] C. Li, K. Sohn, J. Yoon, and T. Pfister, “Cutpaste: Self-supervised learning for anomaly detection and localization,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*. New York, NY, USA: Computer Vision Foundation, Jun. 2021, pp. 9664–9674.

[29] D. Dwibedi, I. Misra, and M. Hebert, “Cut, paste and learn: Surprisingly easy synthesis for instance detection,” in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 1301–1310.

[30] T. Reiss, N. Cohen, L. Bergman, and Y. Hoshen, “PANDA: Adapting pretrained features for anomaly detection and segmentation,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*. New York, NY, USA: Computer Vision Foundation, Jun. 2021, pp. 2806–2814.

[31] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[32] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen, “Deep autoencoding Gaussian mixture model for unsupervised anomaly detection,” in *Proc. 6th Int. Conf. Learn. Represent. (ICLR)*, Vancouver, BC, Canada, Apr./May 2018, pp. 1–19. [Online]. Available: https://openreview.net/forum?id=BJJLHbb0

[33] R. Chalapathy, A. K. Menon, and S. Chawla, “Anomaly detection using one-class neural networks,” 2018, *arXiv:1802.06360*.

[34] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, “Unsupervised anomaly detection with generative adversarial networks to guide marker discovery,” in *Proc. Int. Conf. Inf. Process. Med. Imag.* Boone, NC, USA: Springer, Jun. 2017, pp. 146–157.

[35] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, “Generative adversarial networks: An overview,” *IEEE Signal Process.*, vol. 35, no. 1, pp. 53–65, Jan. 2018, doi: 10.1109/MSP.2017.2765202.

[36] L. Ruff, R. Vandermeulen, N. Goernitz, L. Deecke, S. A. Siddiqui, A. Binder, E. Müller, and M. Kloft, “Deep one-class classification,” in *Proc. 35th Int. Conf. Mach. Learn.*, vol. 80, Jul. 2018, pp. 4393–4402.

[37] G. Rätsch, S. Sonnenburg, and C. Schäfer, “Learning interpretable SVMs for biological sequence classification,” *BMC Bioinf.*, vol. 7, Mar. 2006, Art. no. S9.

[38] D. M. J. Tax and P. Laskov, “Online SVM learning: From classification to data description and back,” in *Proc. IEEE XIII Workshop Neural Netw. Signal Process.*, Sep. 2003, pp. 499–508.

[39] H. Tian, N. L. D. Khoa, A. Anaissi, Y. Wang, and F. Chen, “Concept drift adaption for online anomaly detection in structural health monitoring,” in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage. (CIKM)*. New York, NY, USA: Association for Computing Machinery, Nov. 2019, pp. 2813–2821, doi: 10.1145/3357384.3357816.

[40] M.-N. Nguyen and N. A. Vien, “Scalable and interpretable one-class SVMs with deep learning and random Fourier features,” in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*. Dublin, Ireland: Springer, Sep. 2018, pp. 157–172.

[41] Y. Jin and B. Sendhoff, “Extracting interpretable fuzzy rules from RBF neural networks,” *Neural Process. Lett.*, vol. 17, no. 2, pp. 149–164, Apr. 2003, doi: 10.1023/A:1023642126478.

[42] T. Liu, S. Chen, S. Liang, S. Gan, and C. J. Harris, “Fast adaptive gradient RBF networks for online learning of nonstationary time series,” *IEEE Trans. Signal Process.*, vol. 68, pp. 2015–2030, 2020.

[43] H. G. Han, Q. L. Chen, and J. F. Qiao, “An efficient self-organizing RBF neural network for water quality prediction,” *Neural Netw.*, vol. 24, no. 7, pp. 717–725, 2011.

[44] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, “Efficient BackProp,” in *Neural Networks: Tricks of the Trade*, G. Montavon, G. B. Orr, and K.-R. Müller, Eds., 2nd ed. Berlin, Germany: Springer, 2012, pp. 9–48, doi: 10.1007/978-3-642-35289-8_3.

[45] M. H. Bazargani and B. M. Namee, “The elliptical basis function data descriptor (EBFDD) network: A one-class classification approach to anomaly detection,” in *Proc. Eur. Conf. Mach. Learn. (ECML)*, 2019, pp. 107–123.

[46] K. Weiss, T. M. Khoshgoftaar, and D. Wang, “A survey of transfer learning,” *J. Big Data*, vol. 3, no. 1, p. 9, 2016.

[47] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[48] L. Ruff, N. Görnitz, L. Deecke, S. A. Siddiqui, R. A. Vandermeulen, A. Binder, E. Müller, and M. Kloft, "Deep one-class classification," in *Proc. 35th Int. Conf. Mach. Learn. (ICML)*, vol. 80, J. G. Dy and A. Krause, Eds., Stockholm, Sweden, Jul. 2018, pp. 4390–4399. [Online]. Available: http://proceedings.mlr.press/v80/ruff18a.html

[49] Y. LeCun and C. Cortes. (2010). *MNIST Handwritten Digit Database*. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[50] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," 2017, *arXiv:1708.07747*.

[51] G. B. Moody and R. G. Mark, "The impact of the MIT-BIH arrhythmia database," *IEEE Eng. Med. Biol. Mag.*, vol. 20, no. 3, pp. 45–50, May/Jun. 2001.

[52] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, "PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals," *Circulation*, vol. 101, no. 23, pp. e215–e220, Jun. 2000.

[53] S. S. Xu, M.-W. Mak, and C.-C. Cheung, "Towards end-to-end ECG classification with raw signal extraction and deep neural networks," *IEEE J. Biomed. Health Inform.*, vol. 23, no. 4, pp. 1574–1584, Jul. 2019.

[54] I. Tsamardinos, A. Rakhshani, and V. Lagani, "Performance-estimation properties of cross-validation-based protocols with simultaneous hyper-parameter optimization," in *Artificial Intelligence: Methods and Applications*, A. Likas, K. Blekas, and D. Kalles, Eds. Cham, Switzerland: Springer, 2014, pp. 1–14.

[55] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[56] S. García, A. Fernández, J. Luengo, and F. Herrera, "Advanced non-parametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power," *Inf. Sci.*, vol. 180, no. 10, pp. 2044–2064, 2010.

MEHRAN HOSSEIN ZADEH BAZARGANI received the B.Sc. degree in information technology from the University College of Nabi Akram (UCNA), Iran, in 2010, the M.Sc. degree in computer engineering from Eastern Mediterranean University (EMU), Cyprus, in 2013, and the Ph.D. degree in computer science from University College Dublin (UCD), in 2021. He is currently a Postdoctoral Researcher with University College Dublin (UCD), Ireland. He is the Founder of the machine learning educational platform MLDawn (www.mldawn.com). His research interests include machine learning, deep learning, anomaly detection, active learning, and timeseries analysis.

ARJUN PAKRASHI received the B.Sc. degree (Hons.) in computer science from Calcutta University (CU), India, in 2011, the master's degree in computer science from Banaras Hindu University (BHU), India, in 2013, and the Ph.D. degree in computer science from University College Dublin (UCD), in 2020. He worked in industry in 2015. He is currently a Postdoctoral Researcher with the Insight Centre for Data Analytics and the VistaMilk Science Foundation Ireland Research Centers, University College Dublin. His research interests include machine learning, multi-label classification, ensemble methods, and anomaly detection.

BRIAN MAC NAMEE received the Ph.D. degree in computer science from Trinity College Dublin, Ireland, in 2004. After a period working in industry and at the Dublin Institute of Technology, he joined University College Dublin, Ireland, in 2015, where he is an Associate Professor, the Director of the Science Foundation Ireland Centre for Research Training in Machine Learning (www.ml-labs.ie), and a Funded Investigator with the Insight and VistaMilk Science Foundation Ireland Research Centers. He coauthored the textbook *Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Worked Examples and Case Studies* (MIT Press, 2015 and 2020). His research interests include machine learning, and in particular novelty detection, human-in-the-loop machine learning, data visualization for machine learning, and machine learning applications in domains, such as medicine, agriculture, and space.