Detection of Anomalies in Multivariate Time Series Using Ensemble Techniques

Anastasios Iliopoulos
Dept. Informatics & Telematics
Harokopio University of Athens
Athens, Greece
itp20152@hua.gr

John Violos
Dept. Informatics & Telematics
Harokopio University of Athens
Athens, Greece
violos@hua.gr

Christos Diou
Dept. Informatics & Telematics
Harokopio University of Athens
Athens, Greece
cdiou@hua.gr

Iraklis Varlamis
Dept. Informatics & Telematics
Harokopio University of Athens
Athens, Greece
varlamis@hua.gr

***Abstract*—Anomaly Detection in multivariate time series is a major problem in many fields. Due to their nature, anomalies sparsely occur in real data, thus making the task of anomaly detection a challenging problem for classification algorithms to solve. Methods that are based on Deep Neural Networks such as LSTM, Autoencoders, Convolutional-Autoencoders, etc., have shown positive results in such imbalanced data. However, the major challenge that algorithms face when applied to multivariate time series is that the anomaly can arise from a small subset of the feature set. To boost the performance of these base models, we propose a feature-bagging technique that considers only a subset of features at a time, and we further apply a transformation that is based on nested rotation computed from Principal Component Analysis (PCA) to improve the effectiveness and generalization of the approach. To further enhance the prediction performance, we propose an ensemble technique that combines multiple base models toward the final decision. In addition, a semi-supervised approach using a Logistic Regressor to combine the base models' outputs is proposed. The proposed methodology is applied to the Skoltech Anomaly Benchmark (SKAB) dataset, which contains time series data related to the flow of water in a closed circuit, and the experimental results show that the proposed ensemble technique outperforms the basic algorithms. More specifically, the performance improvement in terms of anomaly detection accuracy reaches 2% for the unsupervised and at least 10% for the semi-supervised models.**

***Index Terms*—Anomaly Detection, Ensemble Methods, Deep Learning, Time Series, Multivariate**

I. Introduction

A significant portion of actual data from systems, phenomena, or measurements is represented by time series. The ever-increasing volume of data makes it challenging for humans to monitor and analyze data, and for this, they rely on algorithms and machine learning models that automate such tasks and improve the monitoring process. In the case of time series, machine learning models try to learn the patterns behind the evolution of a series and either predict future values or detect abnormal situations when they occur. An anomaly is defined as an observation or sequence of observations that deviate significantly from the distribution of the data, and they usually constitute a very small portion of the total data [1]. The anomalies are encountered whenever something goes wrong during the evolution of an operation, a phenomenon, or a process over time. It is crucial to identify these anomalous points and automatically determine if the corresponding sequence of values is an anomaly or not. Anomaly detection is linked to several real-life applications, such as the identification of fraudulent bank transactions, the early detection of machine malfunctioning, the detection of symptoms that indicate the existence of a disease or virus [2], and the detection of system intrusions [3], just to mention a few.

Anomaly detection can be approached mostly from two perspectives. It can be approached as a binary classification problem [4] that classifies an observation as an anomaly or not. It can also be approached as an outlier detection problem where we seek unusual patterns or values that are far from the majority of observations [5]. This study follows the first approach and examines anomaly detection as a binary classification problem. However, apart from the fact that by definition anomalies are very sparse in a dataset, they can also vary significantly in type, which makes it very difficult for a standard classification algorithm to deal with. The three main anomaly types as described in [6] are: i) point anomalies, which refer to anomalies that have extreme values, ii) collective anomalies, which correspond to individual points that have common values with many other points, but which act very strangely as a group, and iii) context anomalies, which refer to points that in some environments or context are considered normal but in some other context are counted as anomalies.

While most anomaly detection models focus on a specific type of anomaly and neglect all other types, our proposed ensemble approach combines multiple models (weak models) that examine different types of anomaly and different features

Identify applicable funding agency here. If none, delete this.

of the data. The ensemble methods we used combines the outputs of the weak models leveraging the diversity of their abilities to capture anomalies in multiple features concurrently. More specifically, in the context of this work, a multi-step approach has been designed, implemented, and experimentally evaluated both in usupervised and semi-supervised setup. At first, multiple models, also named learners, are trained, each one using a different, randomly chosen, subset of features. Then a transformation is applied on each subset computed from principal component analysis (PCA) to capture the variance of data and inject diversity.

In the unsupervised setup the basic models are combined using the majority voting technique while in the semi-supervised alternative a Logistic Regressor is used to combine the predictions and provide a final answer.

The multi-step approach allows the detection of anomalies even when they are hidden in lower dimensions while at the same time, preserving the diversity of each learner. As a result, the ensembles can perform well in higher dimensions and demonstrates an increased performance compared to the original methods.

The major contributions of this paper are listed as follows:

- A method that applies Feature Bagging to the full set of features is combined with multiple basic anomaly detectors that specialize in different anomaly types and manages to uncover anomalies hidden in subsets of features.
- A combination with a transformation based on Principal Component Analysis to the resulting feature sets further increases the diversity of detectors and better captures the different types of anomaly.
- The two feature selection and transformation techniques, are combined with multiple detectors in an ensemble model, which outperforms the baseline methods in terms of prediction performance.
- the methodological approach proposed in this work is experimentally validated on a popular anomaly detection dataset.

The rest of this document is structured as follows: Section 2 briefly surveys the literature on anomaly detection techniques with an emphasis on ensemble methods and their application on time series. Section 3 briefly formulates the challenge that we address in this study. Section 4 formally defines the proposed approach and provides the details of the implemented model. Section 5 describes the experimental evaluation and provides an interpretation of the results. Finally, Section 6 concludes the paper and discusses the next steps of this work.

II. Related Work

Anomaly detection techniques can be divided into six categories according to Cook [7]: statistical and probabilistic, pattern matching, distance-based, clustering-based, predictive, and finally ensemble. Additionally, Chandola et al [6] divide the anomaly detection techniques into supervised, semi-supervised, and unsupervised based on the training mode employed in each case. In a similar way, the methods of the literature are categorized into three main groups: statistical methods, machine learning methods, and deep learning methods [6]. In these categorizations, statistical methods assume that anomalies are generated from a statistical model. In contrast, in machine learning methods, the anomaly generation mechanism is considered a black box and anomalies are detected based on the data. Finally, the third category comprises techniques that are based exclusively on neural networks. They are often considered part of machine learning so there is often no separation between the second and third categories.

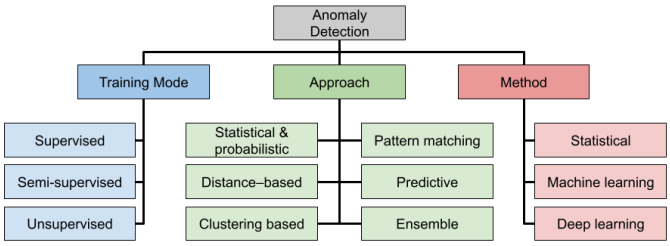

Fig. 1. Alternative groupings of anomaly detection techniques.

Several methods in the literature employ statistical [8] and machine learning techniques [9] to detect anomalies in multivariate time series. The works that build on statistics rely on assumptions about the distribution of data, but their advantage is the cheap computation and the straightforwardness in explaining them. On the other hand, machine learning models can generalize better, but they have more computationally expensive training and retraining in case new datasets arrive. The respective works list several machine learning approaches based on Multi-layer perceptrons, Support vector machines, and Auto-associator approaches such as the PCA.

A grouping of the alternative techniques is depicted in Figure 1. The methodology we propose is based on machine learning techniques and combines elements both from unsupervised and semi-supervised anomaly detection. It can be considered an ensemble deep learning model as it uses a variety of neural networks as a base and builds upon them. The ensemble model makes use of the basic neural networks to discover sequential and spatial relationships in time series. Concerning the nature and distribution of data, we do not make any special assumptions.

In the related literature, we encounter methods that are close to our proposed model, so in the following, we discuss their similarities and differences and use them in the experimental comparison. Chauhan and Vig [10] use the Long Short Term Memory (LSTM) network [11] on an electrocardiography signals dataset that contains 1-dimension time series to detect anomalies. Malhotra et al. [12] use CNNs on a balanced dataset of healthy and broken tooth gears, to detect anomalies. Autoencoders have been first introduced in [13] and have also been extended to detect anomalies in [14]. Malhotra in [14] uses a network similar to the LSTM Autoencoder on three 1-dimension datasets and one proprietary multiple-

dimension dataset. Park in [15] uses an LSTM-Variational Autoencoder (VAE) model for anomaly detection problems on multidimensional time series. In [16] authors have built a collection of multivariate time series, which is used to evaluate anomaly detection algorithms and provide a variety of algorithms as a baseline. Finally, in [17] a method is proposed that results in a hybrid network using LSTM and Capsule networks, which are evaluated on the SKAB dataset.

Although many of the above-mentioned methods are available to detect anomalies in [16] we see that the performance they achieve is not satisfactory and there is significant room for improvement. In this direction, authors in [18] suggest the selection of subsets of the feature set to take advantage of the fact that anomalies in high dimensional data are hardly detected in all the dimensions and can better be detected using only a subset of features. In the same direction, the authors in [19] employ ensembles and boost their performance by increasing the diversity of the learners they employ. For this purpose, they propose a technique called Rotation Forest to increase diversity and accuracy together.

As machine learning methods are evolving, ensemble methods emerge, aiming to combine the advantages of the individual models and methods to capture the different aspects of data. However, ensemble methods are not yet widely used in anomaly detection [20], [21]. Furthermore, the authors in [22] concluded that anomaly detection using an ensemble of methods and an unsupervised approach has limited value, and suggested the use of ensemble methods in semi-supervised or supervised approaches. Therefore, to address the limitations of the above-mentioned approaches and increase the performance of the anomaly detection task, we first recommend the use of both Feature Bagging [18] and a transformation based on Rotations [19] which we call Nested Rotations, simultaneously in an unsupervised setup. To further extend the prediction performance we also evaluate the use of an ensemble technique that follows a semi-supervised approach using the aforementioned techniques. The Feature Bagging technique allows us to shrink the feature space and predict anomalies based only on a subset of features. With the transformation based on Nested Rotations, we divide the problem sub-space into partitions and rotate them to increase the data diversity. For this purpose, we employ the PCA method which gives us orthogonal eigenvectors to use as our new axes.

III. Problem formulation

Let $X = (X_t : t \in T)$ be a multivariate time series, where T is the index set and $X_t \in \mathbb{R}^d, \forall t \in T$. For all $t \in T$ we assign a score which is called anomaly score. Let IQR be the IQR of all anomaly scores the we get during the training phase then, based on a threshold $\delta = 1.5 * \text{IQR}$ it is decided if X_t is an anomaly when the score is greater than δ or not an anomaly otherwise. So we denote the anomaly score of X_t as $\text{ASc}(X_t)$ and transform this problem into a binary problem by defining $\text{ASc}_{\text{binary}}(X_t) = 1$ if $\text{ASc}(X_t) > \delta$ and $\text{ASc}_{\text{binary}}(X_t) = 0$ otherwise. Every detector has a different scale for anomaly score for a given X_t but all detectors have binary outcomes ($\text{ASc}_{\text{binary}}(X_t)$) when we define a threshold. Our approach defines multiple detectors so let $\text{ASc}^i_{\text{binary}}(X_t)$ be the binary outcome for the i-th detector for thr X_t instance. Then we could aggregate the anomaly scores of detectors into one $ASc(X_t) = \text{agg}_{\forall i}(ASc^i(X_t))$ and then transform it into binary. The alternative approach, and the one that we use in this study, is to aggregate the binary outcomes into one binary outcome $ASc_{\text{binary}}(X_t) = \text{agg}_{\forall i}(ASc^i_{\text{binary}}(X_t))$.

IV. Proposed approach

As explained above, we approach the anomaly detection problem in multivariate time series as a binary classification problem and rely on an ensemble model and feature engineering to improve the classification performance. We assume that the distribution of data is unknown and thus we depend on a representation learning approach to capture the patterns hidden in the multivariate time-series data. For this purpose, we apply two feature aggregation and a transformation techniques, namely Feature Bagging and transformation based on Nested Rotation (computed by applying PCA), on five different base architectures (i.e. Autoencoder, Convolutional Autoencoder, LSTM, LSTM Autoencoder and LSTM Variational Autoencoder) in the fully unsupervised approach. In the semi-supervised approach, the predictions of the resulting models are combined with the help of a Logistic Regressor, which is trained to create an efficient ensemble. As depicted in Figures 2 and 3, it is important to decide on the number of models to use in the ensemble both in unsupervised learning and semi-supervised learning, and this is affected by the number of base models, the different feature subsets and the rotations applied to them. In the first step of the semi-supervised approach, the base architectures that will be used to develop the prediction models must be carefully selected to capture the intrinsic characteristics of the various time series. In both setups (unsupervised and semi-supervised) it is important to apply the Feature Bagging technique on the dataset and create several subsets, each one comprising a subset of the original feature, that will be used to train the respective models. The next step is the Nested Rotations of each subspace (defined by the respective subset of features), which is a transformation performed using PCA on partitions of each subspace. The final step in the semi-supervised version is the training of the models with the transformed subsets and their integration in an ensemble predictor, based on a Logistic Regression.

A. Feature Bagging

Feature Bagging has been proposed by Lazarevic and Kumar [18] for multidimensional time series as a solution to anomaly detection problems in high-dimensional data. The authors argued that it is pointless to detect point anomalies based on the similarity (or distance) in high dimensions since the respective metrics lose their meaning when the number of dimensions increases. According to the authors, the larger the number of dimensions, the further the points are separated from each other, which results in multiple isolated points that

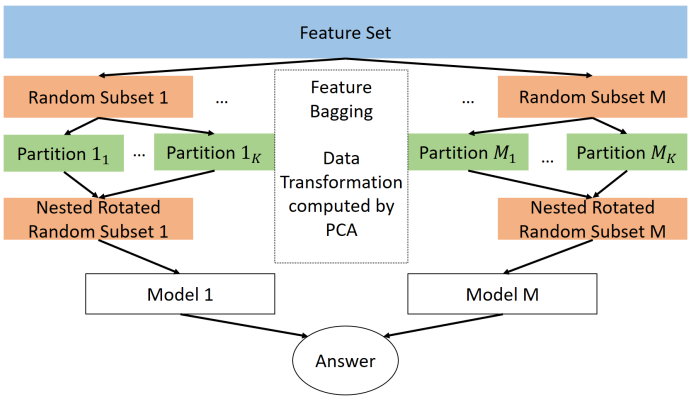

Fig. 2. Feature Bagging & Nested Rotations

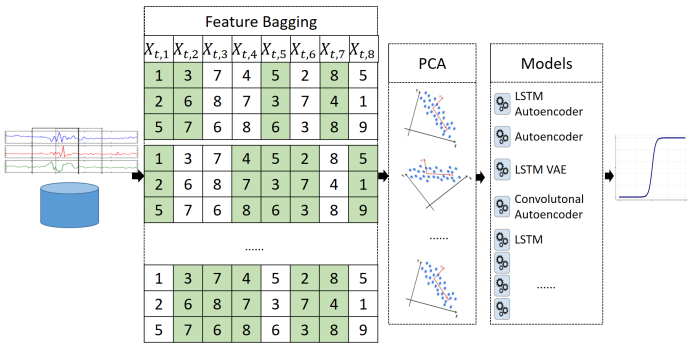

Fig. 3. Stacking/Fusion Feature Bagging with Nested Rotations models with Logistic Regression

are eventually (but falsely) considered anomalies. In addition, if we consider that most of the dimensions introduce noise, then it is more likely that anomalies will be found in a subset of the dimensions. So the Feature Bagging algorithm is proposed for creating subspaces, where it would be easier to identify anomalies.

The process for applying feature bagging and feeding the method ensemble is as follows:

1) Let $X = (X_t : t \in T)$ be a multivariate time series, where T is the index set and $X_t \in \mathbb{R}^d, \forall t \in T$.
2) we select several basic (multivariate time series classification) algorithms (e.g. LSTM, autoencoders, etc.) and train a set M of models. Individual models do not need to use the same architecture, more than one model can result from the same algorithm. For each model $\text{Model}_m \in \{\text{Model}_1, ..., \text{Model}_M\}$ we repeat the following steps:
3) a) We select a random number N_m, where m denotes the $m-$Model, from a uniform distribution between $\lfloor \frac{d}{2} \rfloor$ and $(d-1)$.
   b) Then we randomly select a subset F_m of the original X comprising N_m features without repetition.
   c) We train the anomaly detection model Model_m on the F_m subset.
4) for each instance, using the set of $\text{Model}_1, ..., \text{Model}_M$ models, we get an anomaly score ASc_m from each model for this instance.
5) The final anomaly score of an instance is generated by applying a collection function over the scores derived from each model, $ASc = agg(ASc_1, ASc_2, ..., ASc_M)$. We have selected Majority Voting as the collection function in our experiments.

B. Feature Bagging with Nested Rotations

The technique has emerged from an algorithm that has been designed for classification problems, called Rotation Forest [19]. Rotation Forest extends the popular Random Forest algorithm, which randomly selects subsets of the original feature space to train separate decision tree models, which are then combined in an ensemble. The main novelty of Rotation Forest is that it applies a PCA on the randomly selected subsets of features to get a "rotation matrix" (the principal components), which is then multiplied by the feature subsets to get the rotated features. The algorithm is described below:

1) First the data is normalized. Normalizing the data helps prevent very high values from dominating and influencing the result.
2) For each dimension the average value is calculated.
3) The correlation matrix (covariance matrix) is then calculated for all dimensions. That is $\text{COV}(X,Y) = \frac{1}{m} \cdot \sum_{i=1}^{m}(X - \mu_X)(Y - \mu_Y)$, $\forall X, Y$ where X, Y are two dimensions (two features) and $X \neq Y$.
4) Then the eigenvectors and eigenvalues of the previous table are calculated using SVD. These vectors are also called Principal Components and the respective values represent the "importance" (or informativeness) of each component.
5) We put the vectors in descending order according to the eigenvalues. That is, the vector with the largest eigenvalue is entered first, then the one with the next largest eigenvalue etc. Then we choose the k first vectors.
6) Finally, we take the matrix with the eigenvectors and multiply its inverse with the original data matrix to make the rotation, so:

$$\begin{pmatrix}\text{new data}\end{pmatrix} = \begin{pmatrix} p_1 & ... & p_k \end{pmatrix}^T \begin{pmatrix}\text{data}\end{pmatrix}$$

Instead of keeping only the most "informative" eigenvectors we decide to keep all of them. In doing so we rotate our data according to these axes. To create an ensemble using the rotation method the procedure is as follows:

1) Let $X = (X_t : t \in T)$ be a multivariate time series, where T is the index set and $X_t \in \mathbb{R}^d, \forall t \in T$.
2) We select the M basic models that will compose the ensemble model as before and for each of them we perform the following steps:
   a) We apply the Feature Bagging technique by selecting a random number N_m from a uniform distribution between $\lfloor \frac{d}{2} \rfloor$ and $(d-1)$ for each of the models. This step will give us a subset F_m. The m index denotes the $m-$Model.

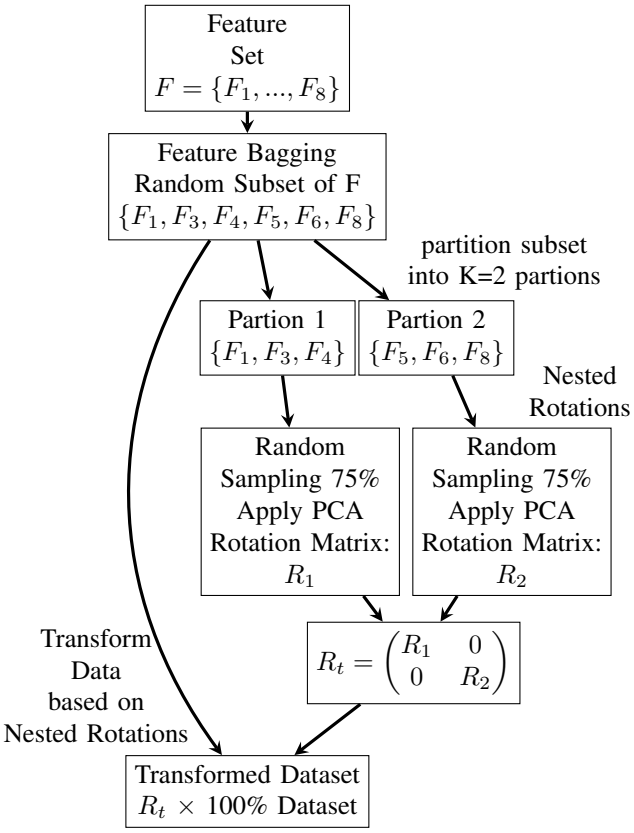

Fig. 4. Nested Rotations

b) for each subset F_m we apply a transformation based on nested rotations by further partitioning it into K subsets such that $K_{m_1} \bigcup K_{m_2} \bigcup ... \bigcup K_{m_K} = F_m$. We then apply PCA on these subsets as shown in Figure 4. So each subset F_m is further partitioned into K partitions:

- We subsampling the subset F_m in order to increase diversity of the partitions. This is done because if two models $m1, m2$ happens to have the same subset $F_{m1} = F_{m2}$ and happens to have the same partitioning $\{f_1, f_2, f_3\}$ and $\{f_4, f_5, f_6\}$ then by subsampling the $F_{m1} = F_{m2}$ resulting into different partitions and thus $K_{m2_1} \neq K_{m1_1}$ and $K_{m2_2} \neq K_{m1_2}$. So by applying PCA (in the next steps) it is not resulting in the same rotation matrices.
- We apply PCA on each partition.
- from each partition we get a rotation matrix R_k^m, where k denotes the k-partition with dimensions (dimension of F_m)$/K \times$(dimension of F_m)$/K$ (for simplicity we suppose that the dimension of F_m is divided with K but it is not necessary).

c) From the previous step (which we called Nested Rotations) we have a transformation for each subset F_m:

$$R_m = \begin{pmatrix} R_1^m & 0 & \cdots & 0 \\ 0 & R_2^m & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & R_K^m \end{pmatrix}$$

where each R_k^m denotes a submatrix corresponding to a partition and 0 denotes the zero-submatrix.

d) Finally, we get the new (nested rotated) data by applying R_m on the subset F_m. So each the models $\text{Model}_1, ..., \text{Model}_M$ is trained on these "nested rotated" data.

3) Finally for each instance, using the set of $\text{Model}_1, ..., \text{Model}_M$ models, we get an anomaly score ASc_m from each model for this instance.
4) The final anomaly score of an instance is generated by applying a collection function over the scores derived from each model, $ASc = agg(ASc_1, ASc_2, ..., ASc_M)$. We have selected Majority Voting as the collection function in our experiments.

The number of individual models M, the number of partitions K that we split each subset F_m the portion of data that we use when we subsampling the each F_m to in order to create the partitions, all these are parameters of this algorithm.

C. Stacking Feature Bagging with Nested Rotations models

In this section, we make use of the previous technique to boost the performance of our prediction in a semi-supervised setup using a Logistic Regressor to combine the individual models as a collection function instead of the Majority Voting we used in the unsupervised setup.

Let $X = (X_t : t \in T)$ be a multivariate time series, where T is the index set and $X_t \in \mathbb{R}^d, \forall t \in T$. We select the M basic models that will compose the ensemble model as before and for each of them we perform the following steps: The procedure for implementing this ensemble method is as follows:

1) First, we divide the training set into two training subsets, one (subset A) for training the individual models and one (subset B) for training the Logistic Regressor. This split guarantees that there will be no leak of information between the individual models and the Logistic Regressor.
2) In the next step we choose the number of individual models M we need to create using the Feature Bagging technique combined with Nested Rotations as in the previous section.
3) With these individual models (trained on data set A) we make predictions on the data set B that the models have not been trained on.
4) As a final step we train a Logistic Regression to improve the prediction performance. More precisely a prediction is made on every element of data set B, and thus every element gets T anomaly scores. So a new data set created $B' = b_1, b_2, b_3$ where $b_i = (ASc_1^i, ASc_2^i, ..., ASc_T^i)$ where ASc_t^i is the anomaly score that model Model_m gave to this element b_i. So at the final semi-supervised

step, we train a Logistic Regression model in data set B' given the labels, to learn how to combine the individual predictions.

The number of individual models M, the number of partitions K that we split each subset F_m the portion of data that we use when we subsampling the each F_m to in order to create the partitions, all these are parameters of this algorithm which derived from the Feature Bagging with Nested Rotations algorithm.

V. Experimental Evaluation

A. Experimental setup

The proposed models of anomaly detection in multivariate time series with ensemble techniques are implemented and evaluated in Python programming language using the frameworks NumPy, pandas, Scikit-learn, TensorFlow 2, and the Keras API. Specifically, we used five basic architectures which are Autoencoder, LSTM, LSTM Autoencoder, LSTM Variational Autoencoder, and Convolutional Autoencoder. The experiments took place in the Google Colab environment except for the model that was proposed in [17] called LSTMcaps.

The datasets we used are taken from Skoltech Anomaly Benchmark (SKAB) [16], which is a collection of time series produced by the operation of a water pump device (motor) monitored by various sensors. The set of sensors generates eight values at every moment thus resulting in multivariate time series with eight features, and each monitoring file contains normal operation and anomaly points. SKAB therefore can be used to evaluate techniques and models in the context of Anomaly Detection research in multivariate time series. SKAB contains 34 multivariate time series of 37401 moments in total or 1100.02 moments per time series on average. Their size is 3.41 MB in total or 102.76 KB per time series on average. They do not contain missing values and they do not contain duplicates. Each point of the time series is a vector of 10 values which are, the timestamp (representing the exact time when the following values were recorded), the anomaly label, and the 8 values that sensors recorded (Accelerometer1RMS, Accelerometer2RMS, Current, Pressure, Temperature, Thermocouple, Voltage, and Volume Flow RateRMS). Each time series is a recorded experiment that starts in a normal state and after a while by switching valves, anomalies are injected.

The evaluation metrics that we report are the F1-Score and Area Under Curve (AUC) of the receiver operating characteristic curve (ROC curve), which is more appropriate for a task that is by definition highly unbalanced between the majority (normal) and the minority (anomaly) class.

The AUC is defined as follows: Let TPR $= \frac{TP}{P}$ and FPR $= \frac{FN}{N}$ be the true positive rate and the false positive rate respectively, where TP stands for the true positive cases, P for all the positive cases, FN for the false negatives and N for all the negative cases. The anomalous points are considered positives and the non-anomalous points are considered negatives. Then the AUC metric is the area under the curve defined by ROC curve as plotted from TPR and FPR pairs for various $\delta \in \mathbb{R}$ where δ is the threshold of our anomaly detection method. F1-Score defined as $\mathrm{F}_1 = \frac{2 \cdot \mathrm{Precision} \cdot \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}}$ where Precision $= \frac{\mathrm{TP}}{\mathrm{TP+FP}}$, Precision $= \frac{\mathrm{TP}}{\mathrm{TP+FN}}$. For both metrics, higher values denote better performance. These two metrics are widely used to evaluate anomaly detection methods (e.g. in [16], [17] and in [23], [24]).

B. Results

In Table I we can see the comparative results of the proposed ensemble methods and other baseline approaches. Every section consists of the architecture the ensemble is based on. We further divide each section and show the F1 Score and AUC regarding 3 types of models of the corresponding architecture, the plain version of the model, the ensemble using Feature Bagging, and the ensemble using Feature Bagging with Nested Rotations. We also included LSTMcaps using the F1 Score that the authors provided in their work. Furthermore, we extend the table and added a section on top of the table for our semi-supervised approach. For each architecture and for each time series we created and train each model as follows:

1) Split/slice the time series into two parts.
2) Train the corresponding model with the first part of the time series. For the "mixed" section we used 3 parts, the first two for training and the third part for testing as described in the subsection "Stacking Feature Bagging with Nested Rotations models".
    a) for the Feature Bagging model we created 17 models of the corresponding architecture.
    b) for the Feature Bagging with Nested Rotations model we created 17 models of the corresponding architecture. For the parameters of all models, we set the proportion of data to be 75%, and the K partitions on which the PCA algorithm is applied was set to 2.
    c) for the architecture "mixed" we used 60 Feature Bagging with Nested Rotations (12 for each architecture) using the first part of the time series and combined them with a Logistic Regressor using the second part of the time series. For the parameters of all models, we set the proportion of data to be 75%, the K subset on which the PCA algorithm is applied was set to 2.
3) Test the corresponding model using the second part of the time series:
    a) for the Feature Bagging model of the corresponding architecture every instance/moment of the time series has 17 labels of 0/1 and we use Majority Voting as an aggregation function to conclude to a single 0/1 label.
    b) for the Feature Bagging with Nested Rotations model of the corresponding architecture using the same logic as above.
    c) for the architecture "mixed" we used the third part of the time series for testing.

At the end of this process, we use the macro-average to conclude the F1 Score and AUC for every architecture and its corresponding models. This evaluation approach has been followed by other researchers in similar tasks (e.g. [16] and [17]) to evaluate their models.

In Table I that follows, we see that Feature Bagging can slightly improve the performance of most base methods. More specifically, the use of Feature Bagging improved the best performance of the plain versions of autoencoder, LSTM autoencoder and LSTM VAE, whereas couldn't beat the best performance of the plain versions of a convolutional autoencoder and the LSTM model without Feature Bagging. This is an indication that Feature Bagging captures anomalies even if they are hidden in a subset of features. To further increase the performance then we combine Feature Bagging and Nested Rotations and again we combine the results with majority voting. We can clearly see that these two techniques in combination provide us with around 2-4% better performance than the plain version of models. The only model that beats the performance of Feature Bagging with Rotation is the LSTM.

As we can see the best model in the unsupervised setup is when we used the Convolutional Autoencoder architecture applying Feature Bagging with Nested Rotations techniques resulting in an ensemble using Majority Voting. This model has a 0.7873 F1 Score and 0.8315 AUC which outperforms the others. Finally as expected in the semi-supervised environment the performance is boosted, and the improvement is at least 10%.

TABLE I
MODEL COMPARISON

Architecture (mode)	Model Title	F1 Score	AUC
Mixed (semi-supervised)	Stacking FBR models with Logistic Regression	**0.85**	**0.88**
Convolutional Autoencoder (unsupervised)	**Feature Bagging with Nested Rotations**	**0.7873**	**0.8315**
	Feature Bagging	0.7451	0.80
	Plain	0.7622	0.8117
LSTM Autoencoder (unsupervised)	**Feature Bagging with Nested Rotations**	**0.7641**	**0.8190**
	Feature Bagging	0.7465	0.8050
	Plain	0.7410	0.8021
LSTMCaps (unsupervised)	LSTMCaps	0.74	-
LSTM (unsupervised)	Feature Bagging with Nested Rotations	0.7074	0.7749
	Feature Bagging	0.6723	0.7500
	Plain	**0.7225**	**0.7867**
LSTM Variational Autoencoder (unsupervised)	**Feature Bagging with Nested Rotations**	**0.7014**	**0.7704**
	Feature Bagging	0.6978	0.7680
	Plain	0.6815	0.7570
Autoencoder (unsupervised)	**Feature Bagging with Nested Rotations**	**0.6259**	**0.7231**
	Feature Bagging	0.5999	0.7089
	Plain	0.5935	0.7050

C. Discussion

From the results of the tables I, we see that Feature Bagging can improve the performance of some models but not all of them. However, when it is combined with nested rotations, the performance is usually better. More specifically, when we combine it with the transformation based on nested rotations computed by PCA into the Feature Bagging with Nested Rotations technique we have an overall improvement of 2%. The fact that every Feature Bagging with Nested Rotations performs better against the majority of the corresponding architectures leads us to the conclusion that this ensemble technique can take anomaly detection a step further. Also, we can conclude that Feature Bagging and Nested Rotations act complementary: Feature Bagging reveals the anomalies and the features that are most affected, whereas Nested Rotations inject diversity and boost the performance of the ensemble. Integrating multiple PCAs with feature bagging leads to increased effectiveness in most cases, as can be seen from the results in Table I. One possible explanation for this is that multiple PCAs after random subspace selection further increases the diversity of the individual classifiers in the ensemble. A deeper investigation of the effect of PCA on the model ensemble is left as future work.

On the other hand, the semi-supervised ensemble model which combines Feature Bagging with Nested Rotations on the individual models, using a Logistic Regressor, gives us the best results on the anomaly detection problem and outperforms all other methods. However, its main limitation is that it needs a few training samples in order to learn how to combine individual models. It also needs much more time for training and inference since it produces multiple models. This is due to the fact that apart from the 60 learners it employs, the PCA algorithm is very time-consuming in its computations. Since PCA is applied many more times than the number of models in the ensemble the computational cost can significantly increase. However, the process can be easily parallelized using a new thread for each model and a multi-core system to do the training or inference.

About the ensemble models (either the unsupervised or the one in semi-supervised) the final number of learners as well as the other parameters of algorithms has been chosen after performing a grid-like search that balances between the available architectures and the number of models to train for each architecture and the other parameters while respecting our limitations in time and computational resources.

VI. Conclusions and Next steps

In this work, we implemented two ensemble techniques applying in unsupervised mode and semi-supervised mode for anomaly detection in multivariate time series. This problem has been approached by various algorithms both from machine techniques and deep learning models. We used 5 basic deep machine learning models and built on them to implement ensemble techniques to achieve even better results. The ensemble models built initially were using two techniques called Feature Bagging and a transformation based on Nested Rotations computed by PCA. The result showed us that these techniques could improve our basic models in combination with unsupervised learning. Feature Bagging alone had almost the same performance as basic models in some cases. When we combined it with Nested Rotations using PCA algorithm, in most cases, performed better than the basic models. Finally, when we combined all of the above into a general model called Stacking/Fusion Feature Bagging with Rotation using a Logistic Regressor in semi-supervised mode and we got the best performance from all the previous techniques as expected. Ensemble techniques are known to greatly improve performance and solve many problems and our results showed that they can also help a lot in detecting anomalies.

Although these models are performing quite well in the anomaly detection problem in an unsupervised or semi-supervised environment they heavily depend on the aggregation function applied to the multiple scores we get at the final stage. Hence it is a valid question whether another function or even a non-linear model can boost the performance further. In [25] they did research on variants of majority voting and presented the two veto-based voting schemes that can combine multiple classifiers together based on their reliability of them. Another important point for future work is to test our model in higher dimensions. The two techniques we used Feature Bagging and Rotation are the core of our models. As we discussed earlier Feature Bagging is robust in high dimensions while Rotation injects diversity and boosts performance. Thus, we believe that as the dimensionality increases, the model should not be affected, but more tests with time series in higher dimensions need to be done to confirm this. Furthermore, as we already mentioned PCA seems to help a lot but a deeper investigation of the effect of PCA on the model ensemble is left as future work.

References

[1] M. Braei and S. Wagner, "Anomaly detection in univariate time-series: A survey on the state-of-the-art," *arXiv preprint arXiv:2004.00433*, 2020.

[2] A. B. Nassif, M. A. Talib, Q. Nasir, and F. M. Dakalbab, "Machine Learning for Anomaly Detection: A Systematic Review," *IEEE Access*, vol. 9, pp. 78 658–78 700, 2021.

[3] G. Mamalakis, C. Diou, A. L. Symeonidis, and L. Georgiadis, "Of daemons and men: reducing false positive rate in intrusion detection systems with file system footprint analysis," *Neural Computing and Applications*, vol. 31, pp. 7755–7767, 2019.

[4] I. Ullah and Q. H. Mahmoud, "Design and Development of RNN Anomaly Detection Model for IoT Networks," *IEEE Access*, vol. 10, pp. 62 722–62 750, 2022.

[5] P. Gogoi, D. K. Bhattacharyya, B. Borah, and J. K. Kalita, "A survey of outlier detection methods in network anomaly identification," *The Computer Journal*, vol. 54, no. 4, pp. 570–588, 2011.

[6] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, pp. 1–58, 2009.

[7] A. A. Cook, G. Mısırlı, and Z. Fan, "Anomaly detection for iot time-series data: A survey," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6481–6494, 2020.

[8] M. Markou and M. Singh, "Novelty detection: A review—part 1: Statistical approaches," *Signal Processing*, vol. 83, pp. 2481–2497, 12 2003.

[9] M. Markou and S. Singh, "Novelty detection: a review—part 2:: neural network based approaches," *Signal processing*, vol. 83, no. 12, pp. 2499–2521, 2003.

[10] S. Chauhan and L. Vig, "Anomaly detection in ecg time signals via deep long short-term memory networks," in *2015 IEEE Conference on Data Science and Advanced Analytics (DSAA)*. IEEE, 2015, pp. 1–7.

[11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] S. Mukhopadhyay and M. Litoiu, "Fault detection in sensors using single and multi-channel weighted convolutional neural networks," in *Proceedings of the 10th International Conference on the Internet of Things*, 2020, pp. 1–8.

[13] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," University of California, San Diego, Tech. Rep., 1985.

[14] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, "Lstm-based encoder-decoder for multi-sensor anomaly detection," *arXiv preprint arXiv:1607.00148*, 2016.

[15] D. Park, Y. Hoshi, and C. C. Kemp, "A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1544–1551, 2018.

[16] I. D. Katser and V. O. Kozitsin, "Skoltech anomaly benchmark (skab)," https://www.kaggle.com/dsv/1693952, 2020.

[17] A. Elhalwagy and T. Kalganova, "Hybridization of capsule and lstm networks for unsupervised anomaly detection on multivariate data," *arXiv preprint arXiv:2202.05538*, 2022.

[18] A. Lazarevic and V. Kumar, "Feature bagging for outlier detection," in *booktitle=11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining,*, vol. 21, 01 2005, pp. 157–166.

[19] J. Rodríguez, L. Kuncheva, and C. Alonso, "Rotation forest: A new classifier ensemble method," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, pp. 1619–30, 11 2006.

[20] C. C. Aggarwal, "Outlier ensembles: position paper," *ACM SIGKDD Explorations Newsletter*, vol. 14, no. 2, pp. 49–58, 2013.

[21] A. Zimek, R. J. Campello, and J. Sander, "Ensembles for unsupervised outlier detection: challenges and research questions a position paper," *Acm SIGKDD Explorations Newsletter*, vol. 15, no. 1, pp. 11–22, 2014.

[22] A. Chiang, E. David, Y.-J. Lee, G. Leshem, and Y.-R. Yeh, "A study on anomaly detection ensembles," *Journal of Applied Logic*, vol. 21, pp. 1–13, 2017.

[23] M. Munir, S. A. Siddiqui, A. Dengel, and S. Ahmed, "Deepant: A deep learning approach for unsupervised anomaly detection in time series," *IEEE Access*, vol. 7, pp. 1991–2005, 2018.

[24] M. Goldstein and S. Uchida, "A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data," *PloS One*, vol. 11, no. 4, p. e0152173, 2016.

[25] R. K. Shahzad and N. Lavesson, "Comparative analysis of voting schemes for ensemble-based malware detection," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, vol. 4, no. 1, pp. 98–117, 2013.