Unsupervised Abnormality Detection System for Autonomous Aerial Vehicles using synchronised IMU and Video data

Deciphi-IITK, Team ID: 26285, ICASSP Signal Processing Cup 2020

Nitish Vikas Deshpande, Afzal Rao, Jaya Srivastava, Adhiraj Banerjee, Prof. Dr. Vipul Arora

***Abstract*—Autonomous systems experience failure when the sensors record abnormal data which leads to the crashing of the decision making module if it is not capable of detecting abnormalities. The design of abnormality detection models becomes an important step in preventing the autonomous systems from malfunctioning. In this paper, we have thoroughly researched all possible steps involved in designing abnormality detection algorithms. The organisers of SP cup have provided us the dataset consisting of data from various sensors mounted on a drone. After identifying the key challenges like limited training data, non-uniformity in sampling of data from different sensors, low FPS of camera, absence of ground truth annotations, we have tackled every problem in a systematic way. We have used Isolation forest method, CNN based timeforecasting method and ARMA model based timeforecasting method to detect the abnormality. We compare these methods using the performance metric of Excess Mass Curves. The Isolation forest method performs better than the timeforecasting approach on the dataset provided by the organisers.**

***Index Terms*—Abnormality detection, Autoregressive Moving Average (ARMA), Convolution neural network, Excess Mass Curves, Isolation Forest, Timeforecasting**

I. Introduction

An abnormality can be defined as a value or a set of values which significantly deviate from the rest of the members of the set or the time series. Abnormalities can be of various kinds:

- Point Anomalies: A single instance of data is anomalous from the rest
- Contextual anomalies: This anomaly is context specific and found in time series data
- Time series discord is a subsequence which has maximum distance from its neighbour

Automatic detection of abnormalities is one of the key initial steps involved in the decision making of any autonomous system. Any spurious value in measurement of parameters like position, orientation, angular velocity or linear acceleration involved in an autonomous system may lead to wrong decision making. Anomaly/ abnormality detection has been widely studied and implemented in various sectors like robotics, automation in manufacturing, logistics and transportation, stock market, etc. In this paper we deal with the problem of abnormality detection in the field of autonomous aerial systems. However, the algorithms that have been used are also applicable in other fields with some minor modifications. This paper is organised as follows. Section II is a summary of the classification of abnormality detection algorithms. Section III describes the problem statement of the SP Cup challenge. Section IV is a detailed analysis of the key challenges. Section V presents our approaches in a systematic way which include pre-processing steps like resampling and interpolation, video feature extraction; and 3 approaches for abnormality detection with 1 approach based on isolation forest method and other 2 use the timeforecasting technique with a CNN based and ARMA based models. Section VI is devoted for describing the choice of Excess Mass curves as a performance metric for comparing the quality of unsupervised abnormality detection algorithms. Section VII compares the performance of the proposed methods. Section VIII discusses some limitations of the proposed approaches and lists down the scope of future work.

II. Literature Review for Anomaly Detection Methods

Anomaly detection algorithms can be classified as supervised, semi-supervised and unsupervised algorithms.

- Supervised algorithms: Data consists of fully labeled training and testing datasets. This is not a practical setup for anomaly detection as it is very difficult to label the abnormalities accurately.
- Semi-supervised algorithms: Data contains both training and testing datasets. However the training data contains only normal instances without any abnormalities.
- Unsupervised algorithms: This setup does not require any label and there is no distinction between the training and the testing datasets.

The tools, datasets, research papers and learning resources for anomaly detection have been compiled by Yue Zhao[1]. A comparative evaluation of unsupervised anomaly detection algorithms has been compiled in [1]. An excellent summary of unsupervised abnormality detection algorithms in unmanned aerial vehicles has been provided by the authors in [2]. They have classified the approaches as Clustering-based, Classification-based, Spectral-based, Model-based and Statistics-based.

[1]Anomaly detection resources by Yue Zhao : https://github.com/yzhao062/anomaly-detection-resources

TABLE I
DATA SET DESCRIPTION

File Name	Attribute				
	Or, Av, La	Mag Field	Temp	Press	Camera
Normal File 0	302	331	332	332	161
Normal File 1	125	136	135	135	51
Normal File 2	134	134	134	133	50
Normal File 3	150	149	149	149	55
Normal File 4	131	131	131	131	48
Normal File 5	142	141	141	141	53
Abnormal File 0	38	42	42	42	20
Abnormal File 1	129	128	129	128	44
Abnormal File 2	144	143	143	143	49
Abnormal File 3	146	145	145	145	40
Abnormal File 4	136	136	136	136	46
Abnormal File 5	157	156	156	156	51

III. DESCRIPTION OF PROBLEM STATEMENT

A ROS based dataset with data collected from various sensors like IMU, barometer, GPS and camera mounted on a drone was provided by the organisers. The data was recorded in different scenarios with separate .bag files for normal and abnormal instances. The task was to train our models on this dataset with IMU and camera synchronised data containing normal as well as abnormal datapoints and build smart prediction algorithms to detect abnormal behaviour that can be tested on new data files.

IV. IDENTIFICATION OF KEY CHALLENGES

We explored some visualisation tools like Rviz and Webviz[2] to get a feel of the challenges involved. We identified the following major challenges in the problem statement

- Lack of training data: The number of training data points were very less as is evident from table I
- Non-uniform sampling and different sampling rates for each sensor: The difference in time between two consecutive timestamps is not the same throughout the data and the sampling rate for each sensor is different i.e the total number of timestamps recorded from a particular sensor for a given duration is different as is evident from I
- Low FPS of camera data: The total number of frames captured by the camera is far low than the number of datapoints captured by IMU as is evident from I
- No ground truth labels/ annotations: The organisers did not provide any annotation for the data. Generally, for evaluation purposes, a ground truth file is provided so that the accuracy of the model can be calculated by comparing the predictions of the model with the ground truth file.

V. METHODOLOGY & IMPLEMENTATION DETAILS

A. Preprocessing

For preprocessing the data captured from the BAG file, we have followed [3] where the setup is very similar to the setup of the organisers by SP cup. They have used a smartphone mounted on the chest of a person and recorded the video data as well as the velocity, acceleration from the inbuilt IMU in the smartphone. The motivation for algorithm 1 was derived from this paper.

[2]Webviz is an browser based tool for visualization of content of a ROSBAG file Link: https://webviz.io/

```
Algorithm 1: Resampling and interpolation
input : BAG files (both normal and abnormal)
output: All attributes for a given file have equal
        number of timestamps. The result for each
        file is stored as a matrix D*M where D is
        the number of attributes and M is the
        number of timestamps. The number of
        timestamps M is different for each file but
        D is same for each file
for each BAG file do
    Extract every attribute as mentioned in table I
      using rosbag library in Python.
    Find the attribute $i$ which contains maximum
      number of datapoints
    For attribute $i$ find the mean sampling frequency
      $F_s$
    Create a new array of timestamps for attribute $i$
      with uniform spacing of $\frac{1}{F_s}$ between 2
      consecutive timestamps
    for each attribute do
        Use the new array of timestamps with
          uniform spacing and find the corresponding
          new value of the attribute using linear
          interpolation
    Store the new values for every attribute in a
    D*M matrix with D as number of attributes and
    M as number of timestamps
```

*1) **Resampling and Interpolation**:*

This is an important preprocessing step in the analysis of the data and the necessity of this step arises due to the challenge mentioned in sectionIV.

*2) **Video Feature Extraction Method**:*

Primer:

Key Points: Key Points are spatial locations, or points in the image that define what is interesting or what stands out in the image. One of the major aspects of a keypoint is that it is invariant to affine transformation or projective transformation on the original image, i.e. if an image shrinks/expands, is translated or is subjected to distortion, the set of key points of the distorted and the original image would be identical.

Descriptors: They are extremely important for matching keypoints between different images. They are primarily concerned with the scale and orientation of the keypoint. In a nutshell, they can be considered as a way to "compare" Key Points. Some important aspects of descriptors are:

- They are independent of the position of the Key Point that they are describing.
- They are robust against image transformations like change of contrasts, or change of perspective.
- They are scale independent.

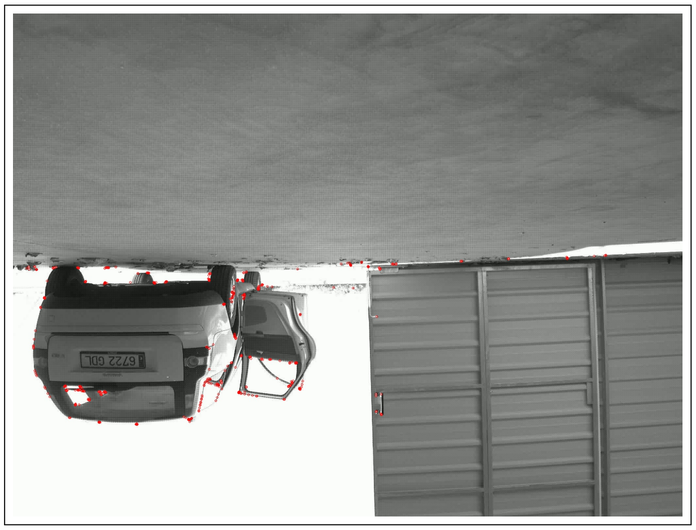

Fig. 1. Key Points (The key points are marked as red dots in the above picture of a car recorded from the drone camera)

Image Features: They are small patches that are useful for comparing similarities between images. An image feature is composed of a Key Point and its corresponding Descriptor.

ORB(Oriented FAST & Rotated BRIEF) [4]: It is a Feature Detection and Description algorithm that is basically a fusion of FAST(Features from Accelerated Segment Test) [5] keypoint detector and BRIEF(Binary Robust Independent Elementary Features) [6] descriptor with many modifications to enhance its performance.

Methodology:

From the original video, we extracted frames in the form of .jpg images every 0.05 seconds. This time interval was decided considering computational costs & time complexity of our methodology.

We computed the i$^{\text{th}}$ frame's similarity(α_{i}) with (i+1)$^{\text{th}}$ frame in terms of matching keypoints. The score that we defined for quantifying similarity was:

$$\alpha_i = \frac{No.of Matching Key Points}{Total No.of Key Points} \tag{1}$$

Considering computation costs and performance, we have used ORB(Oriented FAST and Rotated BRIEF) for feature detection, and through brute force matching, matched the Key Points of i$^{\text{th}}$ frame with the Key Points of the (i+1)$^{\text{th}}$ frame by employing hamming distance as a metric for comparison.

The scores generated by this method were then linearly interpolated over the timestamps provided in the raw images file.

B. Abnormality detection

1) Method 1: Time Forecasting using convolution neural network:

Motivation:

Usually deep learning methods are very data hungry i.e they require a lot of training data to train the neural network. The approach of deep learning is generally used when we have sufficient data. However, some architectures have been developed which can be trained well even with limited data. We derived the motivation for using a convolution neural network for timeforecasting and then using the predicted series to detect the abnormality from DeepAnT [7], which stands for Deep Learning Approach for Unsupervised Anomaly Detection in Time Series. This model can be trained on datasets with few number of training samples as the paper has implemented on Ionosphere datasets with 140 datapoints. The paper also mentions the advantage of using CNNs over LSTMs(Long Short Term Memory) networks which are more data hungry. The model achieves good generalisation capability due to its effective parameter sharing property.

Primer:

Convolution Neural Networks: CNNs have been widely used in Computer Vision applications. Any standard CNN architecture contains a sequence of layers like convolutional layers, max-pooling or average pooling layers, downsampling layers, fully-connected layers, etc. The weights of the network get trained by the standard back-propogation method. We have used CNNs on time-series data for forecasting values.

Methodology:

The 2 key steps involved in the timeforecasting module using CNNs are data preparation and building the neural network architecture.

- **Data Preparation:** The data preparation methodology is explained in algorithm 2. For our model we choose the history window ***Hw*** as 25, the prediction length ***Pl*** as 1 and the hop size ***H*** as 1. So the previous 25 datapoints are used to predict 1 future datapoint. We now have the input data matrix X and the target matrix y that can be fed to the neural network.
- **Architecture Summary:** The neural network was implemented in Keras library. The architecture is as follows:

```
model = Sequential()
model.add(Conv1D(filters=32,
kernel_size=4, activation='relu',
input_shape=(Hw, D)))
model.add(MaxPooling1D(pool_size=2))
model.add(Conv1D(filters=32,
kernel_size=4, activation='relu'))
model.add(MaxPooling1D(pool_size=2))
model.add(Flatten())
```

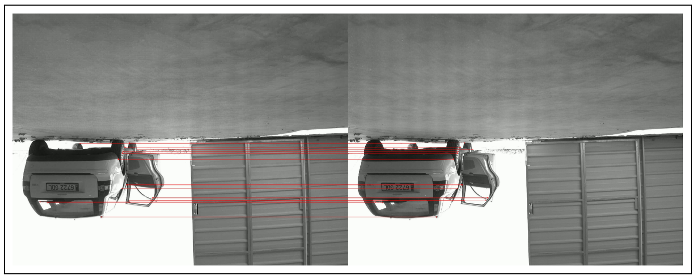

Fig. 2. Feature Matching for Computing Similarity

```
model.add(Dense(50, activation='relu'))
model.add(Dense(D))
```

The Mean Squared Error function is used as a loss function. When the model is trained, we store it in a .h5 file. The model is then tested on new files for timeforecasting.

The anomaly score is calculated by taking the euclidean distance between the predicted time series and the original time series. High values of Euclidean distance implies that there is a high chance that the instance is an abnormal instance. This anomaly score is a continuous function and can be regarded as a probability or a confidence value of the abnormality by mapping the euclidean distance in the range [0,1] using the logistic mapping function. More sophisticated anomaly scoring functions are possible for models based on multi-step forecasting and are explained in the scope of future work. However, since, we have used the prediction length ***Pl*** as 1, we restrict our anomaly scoring function implementation to the Euclidean distance measure. Further if a particular application requires the anomaly detection to be binary, one can use a threshold on the anomaly score and classify instances below threshold as normal and those above threshold as abnormal. The optimal threshold can be decided by using the Precision-Recall Curve method or the Area Under the PRC curve (AUC measure) method.

Algorithm 2: Data Preparation

```
input :  Resampled and interpolated data matrix as
         obtained from algorithm 1
output:  Input data to neural network X matrix and
         Target data to the neural network y matrix
Choose a suitable history window Hw, a suitable
  prediction length Pl and a suitable hop size H
for each attribute do
    Starting from time index 0, store the first Hw
      datapoints in row of X and the next Pl
      datapoints in row of y
    Shift the index pointer by H timesteps and
      repeat the above step till the index pointer
      reaches $M - Hw - Pl$ where M is the length
      of the timeseries of the input sequence
```

2) *Method 2: Time Forecasting using ARMA Model:* ***Motivation:***

We derived the motivation of using the autoregressive moving average model for timeforecasting and anomaly detection from [8] with an aim that the model needs to run in real time with minimal computational requirements. The paper has used methods like SARIMA and STL and compared their performance with deep learning models based on LSTMs.

Primer:

The ARMA model consists mainly of 2 key components:

- **Auto-Regressive Model:** Here the term “auto” stands for self. So auto-regressive means regressing from the same time series i.e a regression model which predicts the future values from the past values in the same sequence. $Z_t = \phi_1 Z_{t-1} + \phi_2 Z_{t-2} + \cdots + \phi_p Z_{t-p} + \alpha_t$
  Here “p” stands for the order of the AR model Z_t, α_t is the white noise, ϕ is the auto-regressive paramter.
- **Moving Average Model:** $Z_t = \alpha_t - \theta_1 \alpha_{t-1} - \cdots - \theta_q \alpha_{t-q}$
  Here α_t is the white noise, “q” stands for the order of the MA model Z_t, θ are the parameters of the model.

Methodology:

Time forecasting based on a given time series has been widely explored. In this work, we have designed a forecasting system which models the training data with an Auto Regressive Moving Average (ARMA) model. We have used the popular statistical library, pmdarima (pyramid-arima, equivalent of R’s auto.arima functionality) for ARMA modelling of each time series recorded by the whole system.

For tuning the parameters of our ARMA models, we have used the auto-arima function. As presented in this work [9], we can also formulate an optimization problem to get sparse solutions for the Auto Regressive coefficients and Moving Average Coefficients using the adaptive LASSO. We then designed a rolling forecaster of the IMU data which predicts the next instant based on the train values and present values (as recorded by the system).

Algorithm 3: Rolling Forecaster

```
input :  IMU data as recorded by the system
output:  IMU data estimate as predicted by the
         ARMA model
Initialize a list of name history composed of the
  training data.
Fit the best ARMA model for the IMU data with
  history.
for each attribute do
    Predict IMU estimate one step in the future.
    Add the IMU data recorded by the system to
      history.
    Fit the ARMA model which models the IMU
      data with history
    Store the predicted values in an array.
```

3) *Method 3: Isolation Forest:* ***Motivation:***

One of major challenges as mentioned above was the unequal sampling rates of all sensors present in the drone. To tackle the same, we had resampled the data & stored it for training our classifier.

Two of the most important properties of anomalies were that they were few in number & were comparably different from normal data points. Taking advantage of the same, they were hence easy to isolate from normal points.

Henceforth, Isolation forest, that employs this strategy for anomaly detection was employed for the given task.

Primer:

Principal Component Analysis: It is a procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables namely principal components.

Explained Variance Fraction: The fraction of variance explained by a principal component is the ratio between the variance of that principal component and the total variance.

Isolation Forest: [10] It is an unsupervised learning algorithm used for anomaly detection. It works by isolating anomalies, instead of conventionally profiling normal points. Abnormal points are isolated more easily than the normal points. Hence the number of iterations required to isolate abnormal points is far less than the number of iterations required to isolate normal points. This results in the normalized isolation number of normal points closer to 1 and that of abnormal points closer to -1. The isolation number can be mapped to the range [0,1] to indicate the probability of the confidence in abnormality by using some appropriate mapping function.

Algorithm 4: Isolation Forest Classifier

```
input  : d-dimensional data point & a positive
         integer n_iter
output: Isolation Score, a floating number
         between -1 & 1
Select a data point in the dataset ;
Declare a float one dimensional array A of size
 n_iter ;
while n_iter>0 do
    for Each feature in dataset do
        Set a range to isolate between minimum &
         maximum
    while Point is not isolated do
        Choose a feature randomly
        Pick a value that is within the range
        if Chosen value keeps point above then
            Switch minimum of range of feature to
             this value ;
        else
            Switch maximum of range of feature to
             this value ;
    isolation number = No. of times the loop
     is iterated ;
    Normalise isolation number b/w -1 & 1;
    A[n_iter] = isolation number;
    n_iter = n_iter - 1;
isolation number = mean(A);
return isolation number;
```

Methodology:

Out of the twelve abnormal & normal files, 5 abnormal files & 5 normal files were selected for training and the remaining 1 normal file & 1 abnormal file was used for testing the method. From the raw bag files, the following parameters were extracted:

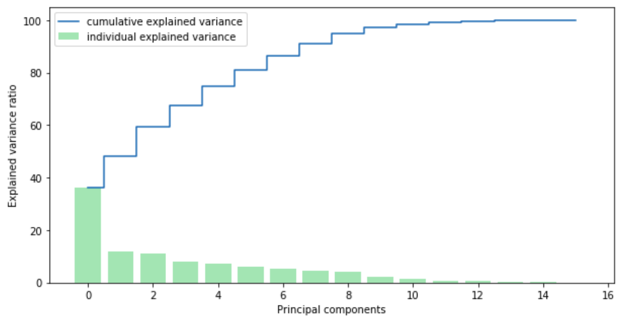

Fig. 3. Plot for Cumulative Explained Variance Ratio & No. of Principal Components

- Orientation (x,y,z,w) [from /mavros/imu/data]
- Angular Velocity (x,y,z) [from /mavros/imu/data]
- Linear Acceleration (x,y,z) [from /mavros/imu/data]
- Magnetic Field (x,y,z) [from /mavros/imu/mag]
- Static Pressure [from /mavros/imu/static_pressure]
- Frame by Frame Similarity Score [derived from /pylon_camera_node/image_raw]

All these parameters were extracted by employing the rosbag package available in Python. The Frame by Frame Similarity Score was derived from the Video Feature Extraction method that has been described earlier.

To remove any sort of redundancy between the variables and also improve the performance of our classification task, we decided to employ PCA, which is an unsupervised dimensionality reduction technique.

One of the major challenges in implementing the PCA model was to determine what number of principal components were required. This was determined by looking at the cumulative explained variance ratio as a function of the number of principal components as shown in fig 3 . It was observed that around 91% of the explained variance was contained within 8 principal components. Henceforth, the number of principal components chosen for our PCA model was 8, and it was implemented using the Python SKLearn Framework.

The reduced data points were then fed into an Isolation Forest Model, with the parameter of n-iter was set to 300 as mentioned in 4. This model was also implemented using the Python SKLearn Framework. The contamination parameter, that considers the proportion of outliers present in the training set, was set to 'auto'. It was used to define the threshold on the scores of the samples that were trained.
The PCA & Isolation Forest models trained were saved in .sav format using the Pickle library available in Python for future use.

VI. Evaluation schemes

An algorithm is as good as its evaluation scheme. We have devoted a considerable amount of time in searching for evaluation schemes for abnormality detection algorithms and

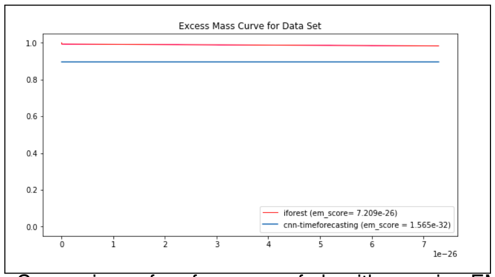

Fig. 4. Comparison of performance of algorithms using EM curve

hence have described it in this separate section. Generally, the performance metric used to evaluate and compare the quality of anomaly detection algorithms is the F1 score of the AUC measure. But this is only true if the ground truth annotations are included in the dataset. As was stated in section IV regarding the absence of ground truth data for the SP cup challenge, we initially tried developing our own interface to manually annotate the datapoints as normal and abnormal by visually inspecting the video and IMU data. However, manual annotations was a tedious task and it was not feasible to accurately annotate every datapoint. We came across the technique of Excess-Mass and Mass-Volume (EM and MV curves) in [11] which has recently become popular in literature in the evaluation of unsupervised anomaly detection algorithms. The speciality of this measure is that they do not require any ground truth label for computing this performance metric.

VII. Results and conclusions

Using the evaluation scheme described in Section VI, we have used Excess Mass curves for comparing our abnormality detection algorithms. Fig 4 shows that the EM score for the Iforest approach is larger than the EM score for the CNN based timeforecasting. The isolation forest algorithm performs better than deep learning approach. We conclude that the choice of algorithm depends on several practical constraints which include quantity and quality of the data, computing resources, feasibility of implementation, etc. For this dataset provided by the organisers, even simple approaches like Isolation forest give good results. For more complicated time-series data, one can use deep learning approaches.

VIII. Scope of future work

For video feature extraction, one of the approaches could be using optical flow methods as was used in [3]. If the FPS of camera were better, we would have used this approach to calculate the yaw, pitch and roll using optical flow method. The problem of limited data can also be tackled using data augmentation. We aim to incorporate data augmentation techniques as in [12] to enhance the quality of the data. The methodology of abnormality detection based on timeforecasting as described in V-B1 and V-B2 can be improved by combining the deep learning and statistical approaches as was done by the authors in [13]. For this problem, we have limited our analysis to the single step forecasting i.e the prediction length ***Pl*** was chosen as 1. However for more complicated problems, we can choose ***Pl*** more than 1. In such cases, some modifications need to be done in the anomaly scoring function. Instead of using a simple euclidean distance, a weighted euclidean measure can be used. The authors in [14] have proposed an innovative anomaly scoring technique which assigns weights to the previous frames that give prediction of the current frame. The weights are based on the anomaly score of the previous frames. We aim to incorporate the above modifications to make our model more robust to handle complicated cases of abnormalities.

Acknowledgment

We would like to thank Prof. Dr. David Martín Gómez(Intelligent Systems Lab, Carlos III University of Madrid) for organising this challenge and providing us an excellent opportunity to apply our theoretical knowledge on real world problems. We are also grateful to our faculty advisor Prof. Dr Vipul Arora (IIT Kanpur) for mentoring our team.

.

References

[1] 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.

[2] S. Khan, C. F. Liew, T. Yairi, and R. McWilliam, "Unsupervised anomaly detection in unmanned aerial vehicles," *Applied Soft Computing*, vol. 83, p. 105650, 2019.

[3] R. Bonetto, M. Soldan, A. Lanaro, S. Milani, and M. Rossi, "Seq2seq rnn based gait anomaly detection from smartphone acquired multimodal motion data," *arXiv preprint arXiv:1911.08608*, 2019.

[4] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International conference on computer vision*. Ieee, 2011, pp. 2564–2571.

[5] E. Rosten, R. Porter, and T. Drummond, "Faster and better: A machine learning approach to corner detection," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 1, pp. 105–119, 2008.

[6] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *European conference on computer vision*. Springer, 2010, pp. 778–792.

[7] 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.

[8] S. Lee and H. K. Kim, "Adsas: Comprehensive real-time anomaly detection system," in *International Workshop on Information Security Applications*. Springer, 2018, pp. 29–41.

[9] K.-S. Chan and K. Chen, "Subset arma selection via the adaptive lasso," *Statistics and its Interface*, vol. 4, no. 2, pp. 197–205, 2011.

[10] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 Eighth IEEE International Conference on Data Mining*. IEEE, 2008, pp. 413–422.

[11] N. Goix, "How to evaluate the quality of unsupervised anomaly detection algorithms?" *arXiv preprint arXiv:1607.01152*, 2016.

[12] K. Babaei, Z. Chen, and T. Maul, "Data augmentation by autoencoders for unsupervised anomaly detection," *arXiv preprint arXiv:1912.13384*, 2019.

[13] M. Munir, S. A. Siddiqui, M. A. Chattha, A. Dengel, and S. Ahmed, "Fusead: unsupervised anomaly detection in streaming sensors data by fusing statistical and deep learning models," *Sensors*, vol. 19, no. 11, p. 2451, 2019.

[14] W. Wu, L. He, and W. Lin, "Local trend inconsistency: A prediction-driven approach to unsupervised anomaly detection in multi-seasonal time series," *arXiv preprint arXiv:1908.01146*, 2019.