UNSUPERVISED ABNORMALITY DETECTION BY USING INTELLIGENT AND HETEROGENEOUS AUTONOMOUS SYSTEMS

Partho Ghosh[1]*, Md. Abrar Istiak*[1]*, Mir Sayeed Mohammad*[1]*, Swapnil Saha*[1]*, Ahsan Habib Akash*[1]*, Nayeeb Rashid*[1]*, Asif Shahriyar Sushmit*[2]*, and Taufiq Hasan*[2]

[1]Department of Electrical and Electronic Engineering (EEE)
[2]mHealth Research Group, Department of Biomedical Engineering (BME)
Bangladesh University of Engineering and Technology (BUET), Dhaka - 1205, Bangladesh.

ABSTRACT

Anomaly detection from drone flight data is challenging task. In this work we have proposed an ensemble of classical and neural network based approaches to automatically localise and quantify anomalies from drone flight data in both time and sensor domain. Our experiments showed that using the IMU sensor values is the most efficient way for this task. We have also devised a novel temporal annotation scheme for the flight data anomaly and found our method to be highly accurate in anomaly detection. Our weighted ensemble architecture of these two state-of-the-art method scored a **MSE** of **0.031** on the manually annotated IMU test data which is better than both the machine learning and the deep learning approaches of ours.

Index Terms— Isolation Forest,Combinational sensorial info, DevNet

1. INTRODUCTION

In real world datasets it is often important to identify data instances that are dissimilar to all other ones. Recognizing these outliers or anomalies and locating the triggers is vital for anaylizing data driven systems. The goal for *anomaly detection* is to determine all such instances based on data. [1]. Hawkins defines an outlier as an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism. At one side it is important to dissonance the normal incident or event from the anomalous incident. On the other hand it is also important to define the level of abnormality. Level of abnormality helps us understand at what sort of measures necessary to take during an anomalous event. Anomalies can be classified into the following three categories: **Point Anomaly**, **Contextual Anomaly** and **Collective Anomaly**. In our problem we were to find all three types of anomalies as well as the level of anomalies. For this challenge we were provided with the Inertial Measurement Unit (IMU) data and video data of a drone flight. The whole challenge was an Unsupervised challenge, so the data point ground truth were not provided. Through our extensive study on the IMU dataset and video data we manually annotated the data provided to us which was vital step towards developing our algorithm. Many algorithms for unsupervised anomaly detection have been proposed, which can be grouped into three main categories [1]. In practical applications, nearest-neighbor based algorithms seem to be the most used and best performing methods today. In this context, outliers are determined by their distances to their nearest neighbors, A very well known local algorithm is the Local Outlier Factor (LOF) [2].The second category is cluster based i.e.K-means clustering [3],Gaussian Mixture Model [4].The third category comprises of statistical methods, both using parametric and non-parametric models for anomaly detection i.e.Mahalanobis distance based [5],Histogram-based Outlier Score (HBOS) [6] which seems to be promising.Currently state of the art on the current trends for anomaly detection systems in UAV is Isolation Forest [7].As for our algorithm we tried both the traditional machine learning methods as well as Deep learning methods. Although deep learning and machine learning algorithm has been applied to successfully address many data mining problems, relatively limited work has been done for anomaly detection. One of main reasons being very few amount of labeled data available and also very the imbalance nature between normal and anomalous instances. Most of the existing methods depends on the feature representation based learning which most of the time lead to a sub-optimal anomaly scoring. In this paper we proposed an ensemble of two state-of-the-art anomaly detection algorithm among them one is a classical machine learning approach and another is a deep learning approach. The machine learning approach [] is based on the feature learning approach and deep learning approach [8] help the algorithm to differentiate between the normal and abnormal instances. Therefore our ensemble were able to generate optimum anomaly score in all perspective.

2. DATASET

This is dataset of Signal Processing Cup 2020. The participants were provided with rosbag files for 12 data samples; 6 normal and 6 abnormals. Each of the rosbag files contain data from a heterogeneous autonomous system where a drone surveil a car in real life. In normal data the drone hovers in a position without any significant movement. In abnormal data it is found to roll.

The rosbag files contains multimodal sensor info data and corresponding videos of low frame-per-second rate. 16 of the sensor data has been extracted to a csv file and the images were separately extracted to a folder. The CSV file and clips were used to develop the predictive models.

2.1. Sensors and Data Volume

The data-set for the problem statement contains 12 ROS-bag files in total, providing flight data of the heterogeneous system. Among the data-set, 6 bag files contain normal flight data, and 6 bag files contain a mix of normal and abnormal data. In accordance with the release sequence of the datasets, ROS-bag files will be referred in this document as dataset1 (first normal bag file), dataset2 (first abnormal bag file), datasetA1-datasetA5 (5 normal bag files), datasetB1-datasetB5 (5 abnormal bag files).

Table 1: Training/test splits in the datasets used

Sub-datasets	Normal	Abnormal
Number of Files	6	6
Image Instances	420	264
Sensor Data Instances	991	747

The ROS-Bag files provided primarily consist of numeric and image data. Numeric data has a sample rate of 3.9 Hz and image data has a sample rate of 2.0 fps with a resolution of 1536x2048. Position and linear velocity data is derived from local GPS sensor. Orientation, angular velocity, linear acceleration, temperature and pressure data is from the IMU sensor on the MAV. Voltage and current data comes form on-board current and voltage sensors.

Table 2: Dataset Description

ROS-Topic	Message
/mavros/global_position/local	pose.pose.position
/mavros/global_position/local	twist.twist.linear
/mavros/imu/data	orientation
/mavros/imu/data	angular_velocity
/mavros/imu/data	linear_acceleration
/mavros/battery	voltage,current
/mavros/imu/temperature_baro	temperature
/mavros/imu/static_pressure	fluid_pressure
/pylon_camera_node/image_raw/compressed	data

Table 3: Prominent Features

Features	Variable Names	Source
Position	pos_x,pos_y,pos_z	GPS
Orientation	ori_x,ori_y,ori_z,ori_w	IMU
Linear Velocity	vel_lin_x,vel_lin_y,vel_lin_z	GPS
Angular Velocity	vel_ang_x,vel_ang_y,vel_ang_z	IMU
Linear Acceleration	acc_x,acc_y,acc_z	IMU

Table 3 presents a list of variables referred along the document indicating the prominent features of the datasets.

2.2. Dataset Observation and Analysis

Initially the feature correlation heatmap (figure: 1) was generated to check whether the features used are correlated. The heatmap shows that the primary 16 features taken for analysis are mostly independent and can be analyzed separately.

Video playback of the ROS-bag files revealed the nature of abnormality as unstable flight conditions. In dataset2 the MAV flips over, and in datasetB1-datasetB5 the MAV rolls/flips in random directions. Because of this motion, linear acceleration value in the z-axis keeps shifting polarity due to the continuous rotation of gravitational acceleration vector with respect to body frame. In normal datasets, the z acceleration has a constant bias equal to gravitational acceleration. Similarly in case of orientation, the unit quaternion values cross zero multiple times in the abnormal datasets, but holds a biased value in normal datasets. Angular velocity data from the gyroscope sensor also had useful information which is not immediately apparent because of high sensor noise. The position and linear velocity data was from the GPS sensor. However in the given operating conditions, such data is unbounded and can have high deviations

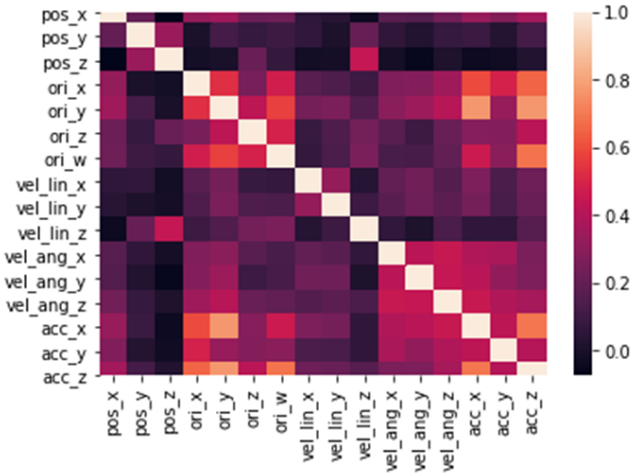


Fig. 1: Feature correlation heatmap for selected features

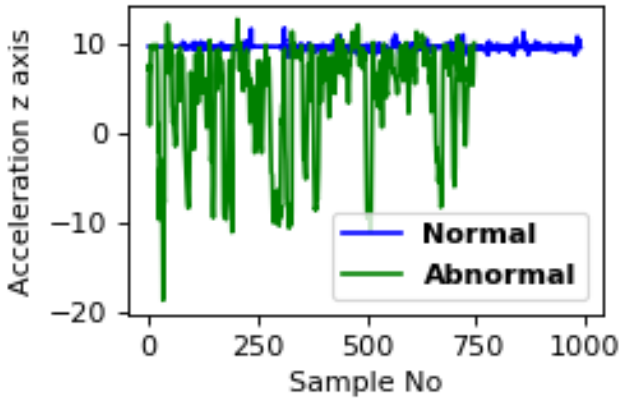
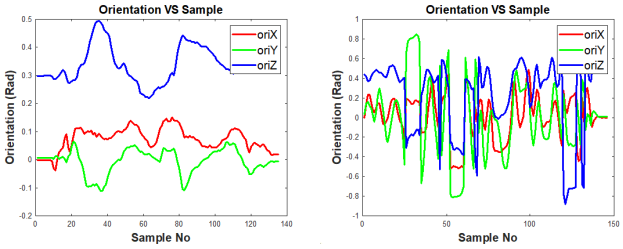


Fig. 2: Acceleration values for all datasets. Blue indicates normal and Red indicates abnormal bag data



(a) Normal (b) Abnormal

Fig. 3: x,y,z rotation values for normal and abnormal datasets

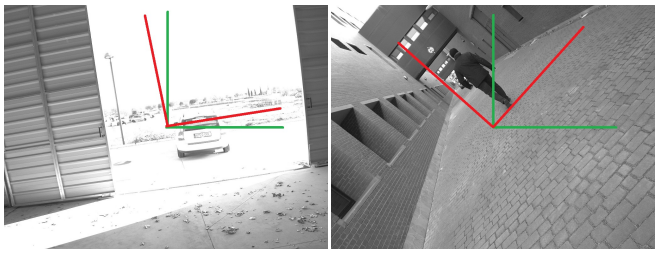
due to sensor biases and low resolution. Thus these two fields were unlikely to contain useful information for abnormality detection.

Pressure, temperature, voltage and current data are also less relevant factors in the type of abnormality as apparent from the datasets. Thus abnormality detection models were designed primarily based on IMU sensor and image data. Relevance of different features have been discussed further in latter sections. (Section 6.1)

2.3. Annotation

Original dataset was binary labeled as normal or abnormal but each sample in the time domain did not have any label.

Image: For deeper understanding of the dataset and tuning unsupervised network models, image data had been manually annotated by visual observation.



(a) Normal (b) Abnormal

Fig. 4: Red axes represent navigation frame and green axes represent body frame

By sequential visual observation of the images, the roll and pitch angle of the MAV was estimated in a range of -90 to 90 degrees, the level position valued 0 for both roll and pitch. The yaw angle was not taken into account as it did not have any clear relation with the type of anomaly. More deviation from level position indicates higher anomaly score. Motivation behind such error estimation is, from the camera feed of the MAV the roll and pitch angle are the most apparent features, and stable operation requires less deviation from level orientation. Putting equal weight on roll and pitch estimation, anomaly score was generated as

$$deviation = \sqrt{\frac{(roll/180)^2 + (pitch/180)^2}{2}}$$

IMU: Similar to the image data, the imu data was annotated with the x and y values of orientation. Orientation data was first filtered using a band-pass filter, then the difference between original orientation and filtered data was used as the deviation score. The pseudo-code is as follows:

```
X = orientationX;
frequency = 3.9;
fH = 0.0001; (Lower cutoff frequency)
fL = 0.05; (Upper cutoff frequency)
[BH,AH] = butter(1,2*fH/frequency,'high');
[BL,AL] = butter(1,2*fL/frequency,'low');
filteredX = filtfilt(BH,AH,X);
filteredX = filtfilt(BL,AL,X);
filteredX = real(filteredX);
filteredX = filteredX-mean(filteredX);
XX = (X-filteredX);
(similarly) YY = (Y-filteredY);
deviation = ((XX^2+YY^2)/2)^0.5;
```

Abnormality score: Finally, the anomaly score was generated using a sigmoidal function that was designed to map the deviation value of 0.1 to 0.5 (decition threshold) and 0.8 to 1.0 (maximum error) (figure: 6)
a = 5.5e-17, b = 4.6, c = 0.06, d = 1.05, m = 0.25;

$$abnormality\_score = d + \frac{a-d}{(1+(deviation/c)^b))^m};$$

Comparison: The abnormality score generated from images had a sampling frequency of 2 Hz and that generated from IMU data had a sampling frequency of 3.9 Hz. Image abnormality score was upsampled by a factor of 3.9/2 to match the IMU data rate. When compared, the two scores had a spearman correlation of 0.8729 and a pearson correlation of 0.8993. This indicates that the annotated scores are a good criteria for performance measurement of unsupervised abnormality detection model.

3. PROPOSED APPROACH

We have explored both state of the art traditional methods and state of the art neural network based methods for constructing a model for unsupervised classification of anomalous data. As our classical approach we have used isolation forest. For the neural network based approach, we have closely followed the works of DevNet [8]. Our proposed approach is an ensemble of the two methods. Figure 7 shows our overall approach.

There are four challenges that we addressed and analyzed each of the constituents of our solution.
i)**Anomaly scoring at each timestamp for the Inertial Measurement Unit(IMU) Data:** Since we were instructed to work only on IMU data and the image data so one of out vital work was to take the each instance we got form the IMU data and annotate a anomaly score to it.
ii) **Temporal localization of anomalous data points:** In a ROS-bag file all the timestamp was not annomaly even the provided dataset had some time instance that were not anomaly at the very beginning of the flight. So it was necessary for us to locate the timestamp that was causing the anomaly during a flight.
iii) **Sensor localization of anomalous data points:** During our work with the algorithm we found out that all the sensor data was not causing anomaly meaning all the sensor data was not giving anomalous information, some sensor were giving more anomalous data while other sensor were giving normal data. So we could realize that during an anomalous incident all the sensor was not equally responsible for anomaly to occur and therefore we tried to localize the sensor to determine which sensors were mostly responsible for anomaly to occur.
iv) **Quantifying anomaly (video, sensor and temporal instance):** In order to determine if our model were determining the anomaly point correctly we compare our model's anomaly score with the manually annotated data point score and generate the mean square error to quantifying the anomaly.

3.1. Isolation Forest

Isolation forest [9] explicitly isolates anomalies rather than profiles normal instances. By constructing tree structure isolation forest can isolate every instance which is called itree.Thus it calculate anomaly score.Repetitive partition is done after selecting random features and random split value which makes a tree like structure.Each tree is consist of differently sub-sampled data from the training data.Ensemble of all the itrees is the isolation forest.Based on tree depth of a instance travelled it can calculate anomaly score.

Given a sample of data X = {x1, ..., xn} of n instances.c(n) is the average of h(x) given n.h(x) of a point x is measured by the number of edges x traverses an itree from the main node until the traversal is terminated at an external node.So the anomaly score s of an instance

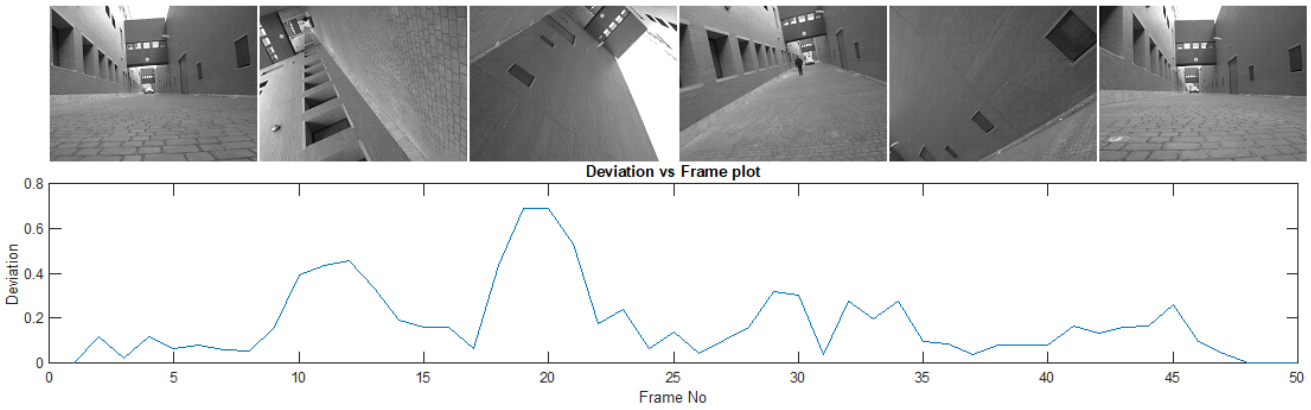


Fig. 5: Empirical deviation plot for an abnormal ROS-bag

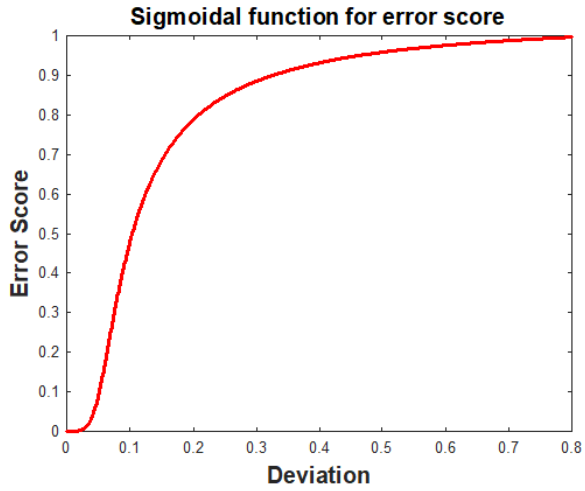


Fig. 6: Transfer function from deviation score to anomaly score

x given n is defined as :

$$s(x, n) = 2^{\frac{-E(h(x))}{c(n)}}$$

As isolating an instance is similar to unsuccessful search in Binary Search Tree thus average path length c(n) is measured as,

$$\text{c(n)} = \begin{cases} 2H(n-1) - \frac{2(n-1)}{m} & \text{if } n > 2 \\ 1 & \text{if } n = 2 \\ 0 & \text{if } n < 2 \end{cases}$$

Where, The harmonic number , H(n)=2(ln(n-1) + 0.572156649) - 2 (n-1)$\frac{}{n}$

3.2. DevNet

Deviation Network or DevNet is a deep learning based anomaly score learning algorithm which is introduced by Guansong Pan et al. [8] The network is called deviation network because the network learns the anomaly score in an end-to-end fashion by using a completely different type of loss function which is called the deviation loss. The network has two section, one is the anomaly scoring network which is the main network that generates the anomaly score

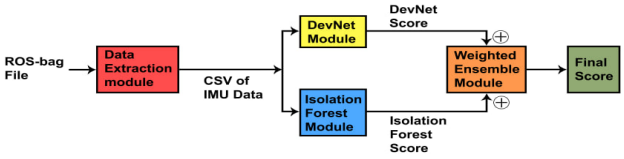


Fig. 7: **Overall workflow of our network**

and another is the reference score generator which basically fixes the mean(μ) and the standard deviation(σ) for the normal instances. So the proposed framework is instantiated by a Gaussian prior and a Z-Score based deviation loss in order to enable the direct optimization of anomaly scores with an end-to-end deep neural network.The whole architecture of the network and workflow of the network is shown in Figure 8.

DevNet was trained to learn the normal distribution which has a mean of 0 and standard deviation of 1. So theoretically DevNet normal score boundary was 0 to 1 and abnormal score boundary was 1 to ∞(if we take the absolute value of the network returned score). To project this score into the range of 0 to 1 we use a squashing function(shown below) so that 0 represent completely normal instance and 1 means completely abnormal instance and 0.5 means borderline of anomaly.

$$final\_score = \frac{1}{0.625 + e^{-|1.3x|}} - \frac{1}{1.625} \tag{1}$$

In equation (1) **x** represents the score returned by DevNet.

3.2.1. *Procedure of the Framework*

As shown in Figure 8, our framework consists of three major modules:

(1)We first use anomaly scoring network, i.e., a function ϕ,to yield a

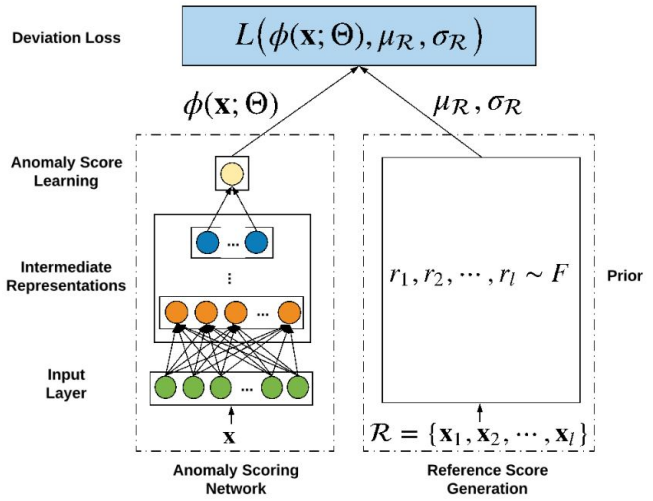


Fig. 8: **The proposed framework.** $\phi(X;\Theta)$ **is an anomaly learner with the parameters** Θ**.** μ_R **is the mean of the anomaly score of some normal objects, which is determined by a prior** F**.** σ_R **is standard deviation associated with** μ_R**. The Loss** $L(\phi(X;\Theta),\mu_R,\sigma_R)$ **is defined to guarantee that the anomaly scores of anomalies statistically significantly deviate from** μ_R **in the upper tail while enforce normal object to have anomaly score as close as possible to** μ_R**.**

scalar anomaly score for every given inputx.
(2)To guide the learning of anomaly scores, we then use a reference score generator to generate another scalar score termed as reference score, which is defined as the mean of the anomaly scores r1,r2,⋯,rl for a set of randomly selected normal objects, denoted as μ_R. The reference score μ_R may be either learned from a model or determined by a prior probability Function . The latter one is chosen so as to efficiently generate μ_R and obtain interpretable anomaly scores.
(3)Lastly $\phi(X)$, μ_R and its associated standard deviation σ_R are input to the deviation loss function L to guide the optimization, in which we aim to optimize the anomaly scores so that the scores of anomalies statistically significantly deviate from μ_R in the upper tail while at the same time having the scores of normal objects as close as possible to μ_R

3.2.2. How Does The Proposed Framework Address the Aforementioned Two Main Challenges of Deep Anomaly Detection?

The deviation loss-based optimization in the framework forces the normal objects cluster around F in terms of their anomaly scores but pushes anomalies statistically far away F, which well optimizes the anomaly scores and also empowers the intermediate representation learning to discriminate normal objects from the rare anomalies with different anomalous behaviors. In other words, the deep anomaly detector leverages a few labeled anomalies and the prior of anomaly scores to learn a high-level abstraction of normal behaviors, enabling it to assign a large anomaly score to an object as long as the object's behaviors significantly deviate from the learned abstraction of being normal. This offers an effective detection of dissimilar anomalies, e.g., anomalies due to different reasons or previously unknown anomalies; and in turn the optimization also requires substantially less labeled anomalies to train the detector.

3.2.3. End-to-end Anomaly Scoring Network

Let Qϵ R^M be an intermediate representation space, an anomaly scoring network $\phi(;\Theta)$:X $\rightarrow\Re$ can be defined as a combination of a feature representation learner $\psi(;\Theta_t)$:X $\rightarrow$ Q and an anomaly scoring function $\eta(;\Theta_s)$:Q $\rightarrow$R, in which $\Theta = \Theta_t, \Theta_s$. Specifically, $\psi(;\Theta_t)$ is a neural feature learner with HN hidden layers and their weight matrices Θ_t=$W^1, W^2, , W^H$,which can be represented as

$$q = \phi(x;\Theta_t) \tag{2}$$

where x$\epsilon\chi$ and q ϵ Q. Different hidden network structures can be used here based on the type of data inputs, such as multilayer perceptron networks for multidimensional data, convolutional net-works for image data, or recurrent networks for sequence data.$\eta(,\Theta_s)$:Q $\rightarrow\Re$ is defined as anomaly score learner which uses a single linear neural unit in the output layer to compute the anomaly scores based on the intermediate representations:

$$\eta(q;\Theta_s) = \sum_{i=1}^{M} w_i^0 q_i + w_{M+1}^0, \tag{3}$$

where qϵ Q and $\Theta_s = w^0(w_{M+1}^0$ is the bias term). Thus,$\phi(;\Theta)$ can be formally represented as

$$\phi(x;\Theta) = \eta(\psi(x;\Theta_t);\Theta_s), \tag{4}$$

which directly maps data inputs to scalar anomaly scores and can be trained in an end-to-end fashion.

3.2.4. Gaussian Prior-based Reference Scores

Having obtained the anomaly scores using $\phi(x;\Theta)$, a reference score $\mu_R\epsilon\Re$, which is defined as the mean of the anomaly scores of a set of some randomly selected normal objects $\Re$, is fed into the network output to guide the optimization. There are two main ways to generate μ_r: data-driven and prior-driven approaches. Data-driven methods involve a model to learn μ_r based on X, while prior-driven methods generate μ_r from a chosen prior probability F. The prior-based approach is chosen here because (i) the chosen prior allows us to achieve good interpretability of the predicted anomaly scores and (ii) it can generate μ_r constantly, which is substantially more efficient than the data-driven approach.
The specification of the prior is the main challenge of the prior-based approach. Fortunately, extensive results show that Gaussian distribution fits the anomaly scores very well in a range of data sets. This may be due to that the most general distribution for fitting values derived from Gaussian or non-Gaussian variables is the Gaussian distribution according to the central limit theorem.Motivated by this, we define a Gaussian prior-based reference score

$$r_1, r_2, ..., r_l \sim N(\mu, \sigma^2), \tag{5}$$

$$\mu_R = \frac{1}{l}\sum_{i=1}^{l} r_i, \tag{6}$$

where eachriis drawn from N(μ, σ^2) and represents an anomaly score of a random normal data object. We found empirically that DevNet was not sensitive to the choices of μ and σ as long as σ was not too large. We set $\mu = 0$ and $\sigma = 1$ in our experiments, which help DevNet to achieve stable detection performance on different data sets. DevNet is also not sensitive to l when l is sufficiently large due to the central limit theorem. l=5000 is used here.

3.2.5. Z-Score-based Deviation Loss

A deviation loss is then defined to optimize the anomaly scoring network, with the deviation specified as a Z-Score

$$dev(x) = \frac{\phi(x;\Theta) - \mu_R}{\sigma_R}, \tag{7}$$

where σ_R is the standard deviation of the prior-based anomaly score set,$r_1, r_2, , r_l.The deviation can then be plugged into the contrastive lo$ $1001[10] to specify our deviation loss as follows$

$$L(\phi(x;\Theta), \mu_R, \sigma_R) = (1-y)|dev(x)| + y.max(0, a - dev(x)),$$

where y=1 if x is an anomaly and y=0 if x is a normal object,and a is equivalent to a Z-Score confidence interval parameter. This loss enables DevNet to push the anomaly scores of normal objects as close as possible to μ_R while enforce a deviation of at least a betweenμ_R and the anomaly scores of anomalies. Note that if x is an anomaly and it has a negative dev(x), the loss is particularly large, which encourages large positive deviations for all anomalies.Therefore, the deviation loss is equivalent to enforcing a statistically significant deviation of the anomaly score of all anomalies from that of normal objects in the upper tail. We use a=5 to achieve a very high significance level (i.e., 5.73303e-07) for all labeled anomalies.

Similar to the contrastive loss, the deviation loss is monotonically increasing in—dev(x)— and is monotonically deceasing in max (0, dev(x)) so it is convex w.r.t. both cases. However, they are also very different, because the contrastive loss uses pairs of intra-class/inter-class data objects as training samples to learn a similarity metric,whereas our deviation loss is built upon the deviation function and dedicated to the direct learning of anomaly scores.

One problem for using loss Eqn is that we do not have the labeled normal objects. We address this problem by simply treating the unlabeled training data objects in U as normal objects. Our empirical results showed that DevNet and also its competing deep methods performed very well by using this simple strategy, even when there was a large anomaly contamination level(i.e., the pro-portion of anomalies in the unlabeled training data set U). This may be because anomalies are rare data objects and their impacts become very limited on the stochastic gradient descent-based optimization in these deep detectors. Therefore, this training strategy is used by DevNet and its competing deep methods throughout our experiments. This can be seen as training the model with noisy data sets.

4. IMPLEMENTATION

In our work we tried to localised the anomaly score in the time domain as well as in the sensor domain. To achieve this we first fit each timestamp of IMU data into our model in order to generate an anomaly score in the range of 0 to 1. In this scoring criteria 0 means totally normal instance and 1 means highest anomaly score possible. So in this scale 0.5 means borderline of anomaly. In order to localise the anomaly on sensor domain we manipulated our network to give results about which sensor data are more responsible for an anomaly ROS-bag file.

Since our dataset were very small we used an ensemble system to achieve the more generalized results as ensemble helped us reach to a more generalise and robust solution than a single state-of-the-art model could ever produce.

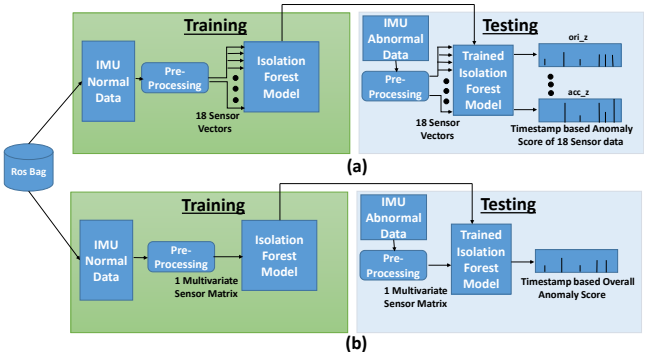


Fig. 9: Isolation forest implementation. (a)Univariate (b)Multivariate

4.1. Implementation details of Isolation Forest

4.1.1. IMU Data

Implementation of Isolation Forest is mentioned in detailed form in three part.

i)**Pre-processing**: As raw data of multi-sensor was given pre-processing was necessary, Firstly standardize features by the mean and scaling to its variance.

ii)**Feature Selection from Multi-Sensor Data**: By observing abnormal and normal data we see GPS related data hasn't fluctuated much as drone hovered at one place.Considering them redundant we discarded them also.Eventually selected 16 features which proved to enhance performance.We experimented different combination of sensor feature fusions to get the best result out from the dataset.

iii)**Training and Testing**: In classical machine learning Isolation Forest [9] plays a vital role in detecting anomaly [11].It is an ensemble of isolation trees and works extremely well on smaller dataset. As it is a different type of model that explicitly isolates anomalies rather than profiling normal instances in training stages it take two parameters:sampling size and number of tree.We take sampling size as mention in paper [11] and number of tree as 200 to reach convergence as paper suggests.Model was trained multivariately for unsupervised classification,temporal localization,quantization and univariately for sensor localization. Different approaches are summarized in figure 9

Testing can be done on different conditions such as more abnormal,more normal,both.In the case of more abnormal it becomes irrelevant distribution for isolation forest as it isolates based on distribution.It desires more normal than abnormal.So we increase the normal space by adding them normal data before tesing any condition.

4.1.2. Image

To detect anomaly we use image also and scheme is given in two part.

i)**Feature Extraction from Image**: We experimented different deep feature extractors such as NASNet Large [12],Xception [13],ResNet101 and 50 [14],InceptionResnetV2 [15] to extract feature from image.We find DenseNet201 [16] was able to extract significant feature than others feature extractors.After experimenting we find DenseNet 201 performed best among all other deep feature by seeing the binary unsupervised classification score.

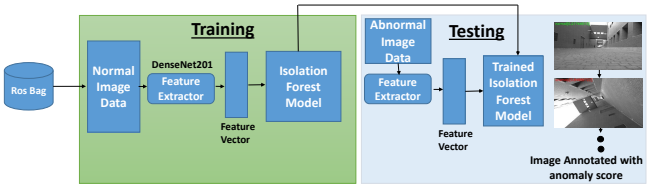


Fig. 10: Isolation forest implementation on Image.One sample of normal and one sample of abnormal is showed

ii)**Training and Testing**: The generated feature vector passed through isolation forest to fit model multivariately having similar parameter like in IMU to detect anomaly.

In testing phase feature was extracted similarly and passed through trained model to detect anomaly. The whole process is summarized in the figure 10.

4.2. Implementation details of DevNet

In the DevNet implementation we used three architecture to produce benchmark anomaly score. First was zero hidden layer meaning the input data is directly projected to a anomaly score, then a single layer architecture in which there was a single layer of 20 hidden neurons were used in between the input data and the output score. Then at last we used three hidden layer architecture in which the input data and out anomaly score had three hidden dense layer in between having 1000,250 and 20 neurons accordingly. All the hidden layer had ReLU activation and L2 regularization technique with a penalty value of 0.01. The output neuron had a Linear type activation and the optimizer for the model was RMSprop and loss function was deviation loss. Empirically we found that DevNet gives the best performance in single layer architecture. In the training of DevNet it was necessary to have some anomalous instance with all the normal instance in order for the network to differentiate between normal and anomalous instances. So to do this we used some annotated anomalous instances by seeing the video frame. The training was done in a balance batch fashion meaning each batch has the same amount of normal and anomaly instances. In order to achieve this the anomaly instances were replicated multiple times as it was necessary to achieve balance batch. In our training we used all the normal instances and 17 abnormal instances as it was made sure by reviewing the images of the abnormal data. Each epoch consist of 20 batches of balanced data and each batch consist of 512 instances and total of 100 epoch was done in a single run. We used a total of 20 runs at different random state to get a generalised model.

4.2.1. Image

The experiments done on the images were first we tried to fit the images to the network without any crop or image processing and tried to extract anomaly score but the model was not being able to differentiate normal and abnormal images and the accuracy was around 20%. Then we pass the images in a ResNet50 [17] feature extractor with it's imagenet [18] weight. The weights of the feature extractor was kept freeze. Then the extracted feature by the feature extractor was fit to the DevNet architecture which improve the result but increase the runtime exponentially. The ResNet50 give an accuracy of 58% and empirically we found out that the larger the feature extractor the greater was the anomaly score but it also increased the runtime. Decresing the image size reduce the runtime but that also decrease the accuracy significantly. Since we found out that the images were hugely co-related to IMU data therefore we drop images from our pipeline as it was increasing our pipeline's runtime exponentially. The way accuracy was calculated is described in **Section** 5

4.2.2. IMU data

For IMU data at first we tried with all the features, including all axis of position, orientation, linear velocity, angular velocity and acceleration.That gave us a maximum accuracy of 77.64%. From our dataset study we found out that the position and linear velocity data is quite irrelevant to scoring the anomaly instances. So we train the model by dropping the position and linear velocity data and it give an increase in the accuracy by almost 7%. It also gave an MSE of 0.05. The model score was highly co-related to the annotated score which is shown in figure 11 and the Pearson co-relation value was determined to be 0.908 for the dataset2 IMU data.

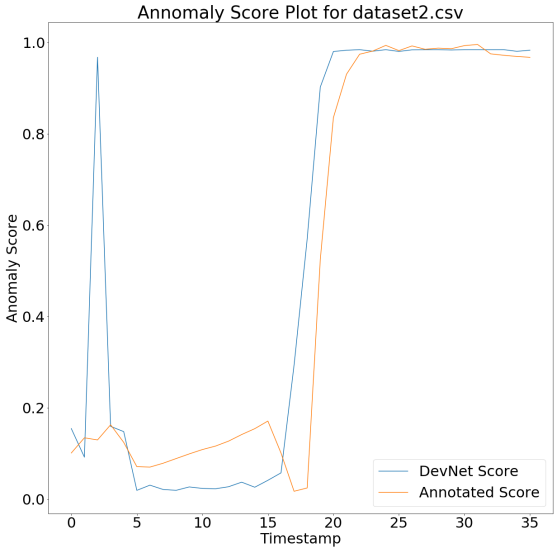


Fig. 11: **Annomaly Score for dataset2**

4.2.3. Sensor Localisation

In order to achieve sensor localisation we used an activation of the sensor to determine which sensor contribute more to the anomaly score. In a single ROS-bag file first we determine for which timestamp the model was generating anomaly score(score above or equal to 0.5) and then for that anomaly timestamp we determine which sensor is the most responsible for the high anomaly score or has the highest contribution in the score. This was done by using the activation map of the neurons. First we determine which neuron is more

activated in the hidden layer meaning which neuron has the significant high weightage to the anomaly score then we select top 5 most activated neurons among the 20 hidden layer neurons. Then we determine the top three neurons in the input layer which value made the hidden neuron get activated. These neurons directly corresponds to sensor value. This is how every sensor got scored and ultimately the top three sensors with the highest score got selected as the most responsible sensor to conduct anomaly in a g flight. So at the end of the ROS-bag the top three sensor that was frequently responsible for the anomaly to occur was returned by the network.

4.3. Ensemble Scheme

Since we had two novel state-of-the-art architecture and both was giving us good result. DevNet was giving a more better result than Isolation Forest. But since we were provided with very small amount of data it was never fully determined whether or not our model was giving a generalised result or a biased result. Therefore we decided to use both of the state-of-the-art architecture in an ensemble fashion. This will somewhat help us land to a more generalised solution. So we ensemble the two architecture score and weighted them in order to minimize the MSE from the annotated score. For this purpose we use a Linear Regression model to compute the best weight for which we can get satisfied results. The linear regression model was run for 2000 epoch with a learning rate of 0.001 and finally the weight that generated the lowest loss was selected.

$$final\_score = (\mathbf{a} \times DevNetScore) + (\mathbf{b} \times IsolationForestScore), \quad (8)$$

The **a** and **b** was found out to be **0.537** and **0.6861** accordingly by the linear regression model. The final score were clipped since theoretically it was possible that it might suppress the score boundary of 0 to 1.

5. EXPERIMENTS AND ANALYSIS

Experimental results are shown in 4 sections. i)Unsupervised Classification of Video: It denotes the binary classification over annotated data of all abnormal rosbag, for this we set a threshold for the anomaly score and annotated them in a binary fashion in such a way that 0 means normal and 1 means abnormal. The actual threshold was at 0.5. ii)Temporal Localization of Anomalous Point: Ability to do temporal localization, iii)Sensor Localization of Anomalous Data Points : As IMU sensor data was taken for analysis we localize the main feature which is mostly accountable for anomaly.Top three responsible features of each abnormal rosbag, iv)Quantifying Anomaly: An instance can be quantified by defining an anomaly score from model ranging from 0-1.It denotes the degree of anomaly a data possess.Here the threshold is 0.5 which means greater than 0.5 denotes anomaly.It is seen most of the anomalous data is scored greater than 0.5 and normal belong to the lesser side of 0.5.

5.1. Experiments done using Isolation Forest

5.1.1. IMU Data

We select different combinations of feature to get the best result from multivariate isolation forest.Experimental results on all given abnormal rosbag are illustrated as 4 challenges.

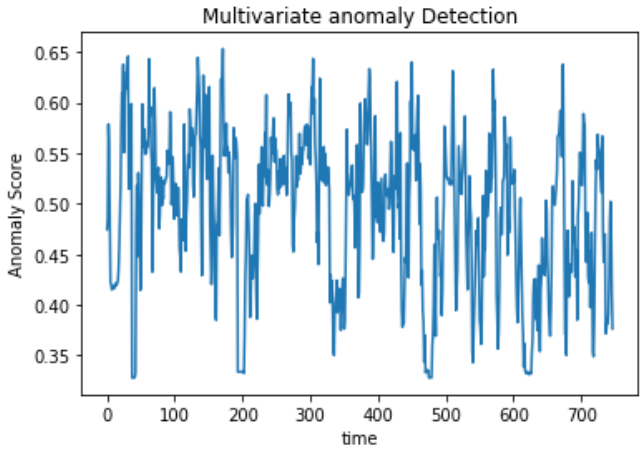


Fig. 12: Anomaly Score VS Time

i)**Unsupervised Classification of Video**: Results are given in table 4

ii)**Temporal Localization of Anomalous Point**: Isolation forest model can also temporally localize an anomalous or normal point.The figure 12 clearly explain the ability to localize a point anomalous or not with respect to time.

iii)**Sensor Localization of Anomalous Data Points**: Top 3 responsible features for anomaly are given in table 9. To find the top responsible features univariate isolation forest is used to detect anomaly score of each feature.Between them we find the top congested scored feature and consider them as most responsible in a timestamp.From an overall rosbag,instances possessing anomaly score greater than 0.5 are measured and most occuring features are given as final responsible features. In figure 13 we see the result of univariate anomaly score of selected 10 features over all abnormal rosbag.Y-axis denotes anomaly score and X-axis denotes timestamp.It is proof that our approach can localize sensorial info from similar kind of multimodal sensorial dataset.

iv)**Quantifying Anomaly** : From figure 12 we can see the quantization performance of isolation forest of all provided anomaly rosbag.Threshold is 0.5 which denotes the borderline to be anomalous suggested as paper [11].Evaluation metric to evaluate the performance of quantization is Mean square error with respect to annotated score in table 6.In paper it was described a specific condition where anomaly(outliers) will be less with respect to the quantity of normal data.But here testing scenario can be different so we observe a different result.So we can generalize all conditions in 3 category such

Table 4: Binary Accuracy of Isolation Forest on Different Conditions

Condition	Trained On	Accuracy
With all IMU features	All Normal	63.3%
Excluding positions	All Normal	69.5%
Excluding linear velocities	All Normal	64.7%
Excluding linear velocities,positions	**All Normal**	**76.2%**
Excluding linear velocities,positions	All Normal and some abnormals	74.6%

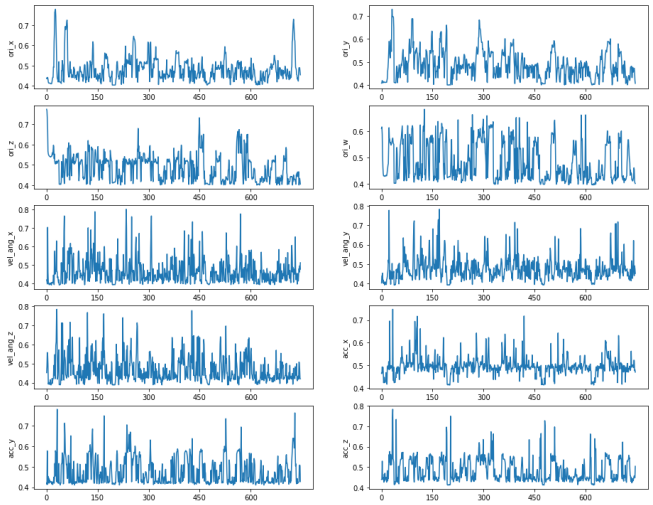


Fig. 13: Univariate Anomaly Score

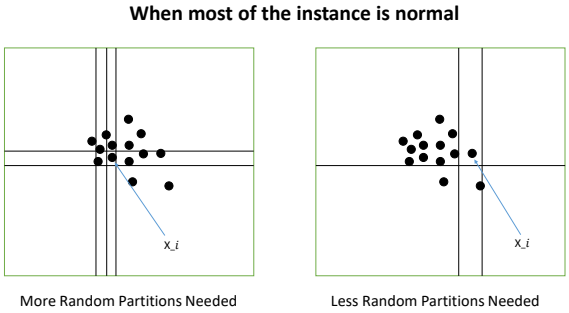


Fig. 14: Random Partition of Most Normal Instance

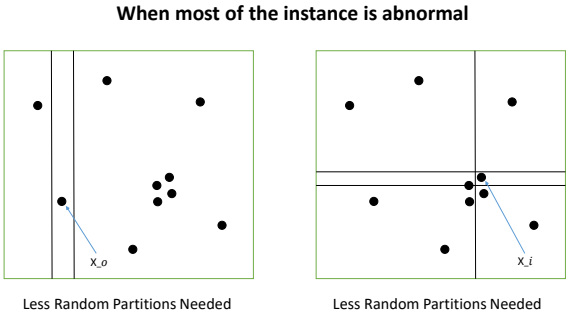


Fig. 15: Random Partition of Most Abnormal Instance

as **i)**when normal(inlier) is more than abnormal(outlier) **ii)**when approximately all are normal **iii)**when most are abnormal(outlier) and normal(inlier) is less in number.So in case (i) isolation forest work properly without any biasness in anomaly score.But in case (ii) there is little bit of biasness which affect some instances to score greater than 0.5 and make them anomalous.This case is illustrated in figure 14.To isolate a inlier in left side of figure 14 more partition is needed which means average of E(h(x)) is more and $2^{\frac{-E(h(x))}{c(n)}}$ scored less which is normal.But in right side of figure 14 isolating the inlier requires less partition as its little bit of outside of congested distribution.So average of E(h(x)) is less and $2^{\frac{-E(h(x))}{c(n)}}$ scored more which may cross 0.5 and making it as abnormal. In case (iii), left of figure 15 a outlier required less partition which makes anomaly score higher than 0.5 but in the right side of figure figure 15 it also requires less partition to isolate a inlier between more abnormal instances.So average of E(h(x)) is more than abnormal instances' average but not much as normal instance.So anomaly score reduces little but not much to cross threshold line 0.5.So it remains greater than 0.5 thus denoting anomaly.

To solve the problem of biasness in score when abnormal is more we fit provided normal data before test data to increase normal distribution and we use a squashing function to settle the bias and set mean at optimum position to cope with number of sample.The empirical squashing function is : $y = \mid 1.4x^3 - 2.3x^2 \mid$

Table 5: Top three responsible sensors of all abnormal rosbag

Rosbag Name	Sensors Name
Dataset2	ori_w, acc_z, ori_y
DatasetB1	acc_z, acc_x , ori_y
DatasetB2	acc_z, ori_x, ori_y
DatasetB3	ori_y, acc_z, acc_x
DatasetB4	ori_x, acc_z, ori_y
DatasetB5	acc_x, acc_z, ori_w

5.1.2. Image

Isolation forest can generate anomaly score frame by frame sequentially denoting the ability to localize temporally and quantifying as seen in figure 17a for normal image and 17b for abnormal image.Anomaly score is given at the top of each image.Evaluation metric to determine performance in image is Spearman correlation co-efficient between annotated data of image.A sample of anomaly score based on timestamp is given in figure 16.Here Spearman correlation co-efficient is 0.817.

For the image size of 700 px X 530 px and feature extractor DenseNet201,n_estimator=150 spearman correlation between annotated data and predicted score(average over all abnormal rosbag) is 0.737.

For extracting more feature quiver plot of optical frame was experimented but redundancy of arrow confuses model more to reduce

Table 6: MSE of Anomaly Score of Isolation Forest

Condition	Trained On	Accuracy
With all IMU features	All Normal	0.122
Excluding positions	All Normal	0.114
Excluding linear velocities	All Normal	0.120
Excluding linear velocities,positions	**All Normal**	**0.107**
Excluding linear velocities,positions	All Normal and some abnormals	0.108

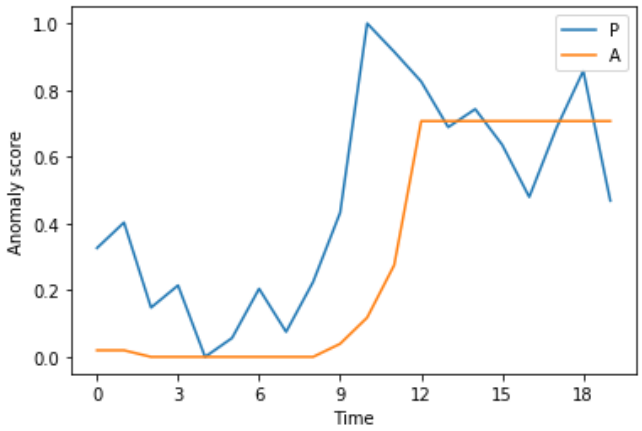


Fig. 16: Anomaly Score VS Frame Count.Here 'P' for predicted and 'A' for annotated score of dataset2

(a) Normal (b) Abnormal

Fig. 17: Anomaly Score of Images from Isolation Forest

correlation between annotated data.So quiver plot doesn't represent motion.From figure 19 we see the proof.

5.2. Experiments done using DevNet

As like Isolation Forest network we also generate and compare four type of evaluation metrices to select the best weight file for DevNet network.

5.2.1. Binary Accuracy of DevNet

The table 6 shows the binary accuracy of DevNet trained on all normal IMU instances and 17 abnormal instances and tested accuracy for only abnormal instances. To determine the optimum threshold for different implementation of DevNet we first determine the models ROC curve from the annotated binary classification. From that ROC curve we determine the equal error rate which is basically $1 - TruePositiverate = FalsePositiveRate$, at the threshold this condition was true we used that threshold point and finally any score above that score was denoted as the anomaly instance (i.e. class 1) and any score under the threshold was considered as normal instance (i.e. class 0). From there we compare with each instance predicted class with the annotated class and report it's accuracy. The ROC curve for our selected model is given in Figure 19

5.2.2. Time Localisation of DevNet

Figure 20 shows the time localisation of DevNet architecture on the test data. The Pearson co-relation for the all the test data point is

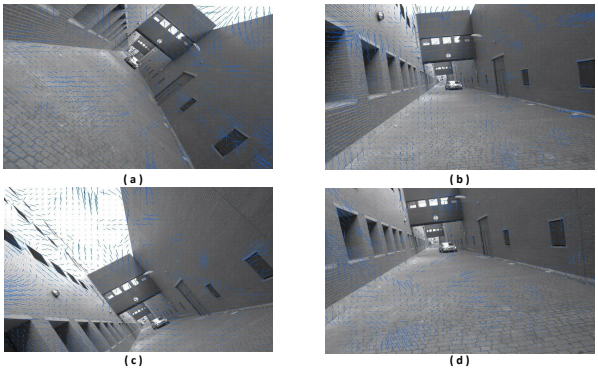


Fig. 18: Optical Flow of four successive frames

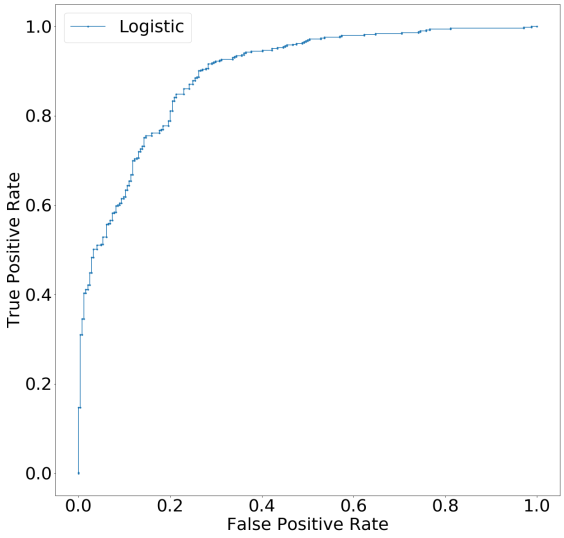


Fig. 19: ROC Curve of DevNet

found to be 0.778 .From the figure we could see that DevNet were able to localise the anomaly scores in the time domain appropriately.

5.2.3. Mean Square Error of DevNet

Table 7 shows the MSE score for different condition and model architecture on test data instances. To calculate the MSE score we first annotated all the The annotated data in the test data instance. Each annotated data was given an anomaly score according to observing it's orientation value in the range of 0 to 1. Then our network used this annotation as the ground truth and determine the data point to point error from its prediction and took square of that error and finally return the mean value of all the square . The lowest MSE give the best resulting model.

As evident from binary accuracy, MSE score and the high co-relation value with the annotated data it was evident that DevNet gave the best performance when it was trained without position and linear velocity data. Figure 21 shows that the best model was able

Table 7: Binary Accuracy of DevNet

Condition	Network Depth	Accuracy
With all IMU features	Single Hidden Layer	77.64%
With all IMU features	No Hidden Layerr	65.33%
With all IMU features	Three Hidden Layerr	62.92%
Excluding position feature	Single Layer	81.77%
Excluding linear velocities,positions feature	Single Hidden Layer	**84.61%**

Table 8: MSE Scores of DevNet

Condition	Network Depth	Accuracy
With all IMU features	Single Hidden Layer	0.089
With all IMU features	No Hidden Layerr	0.132
With all IMU features	Three Hidden Layerr	0.212
Excluding position feature	Single Layer	0.065
Excluding linear velocities,positions feature	Single Hidden Layer	**0.054**

to differentiate normal instance from anomaly instance and project their score in the range of 0 to 0.5 and project anomaly instances score in the range of 0.5 to 1 based on their level of abnormality where 0.5 represents as the borderline anomaly. **??**

As it was evident from the MSE table and the binary accuracy table DevNet with single hidden layer and without the position data and the linear velocity data it was able to perform the best and give the best results. Therefore that model was selected as the best model for our DevNet. This model was able to differentiate the normal instance and abnormal instances and score them in their region which is 0 to 0.5 as the normal instance region and 0.5 to 1 as the abnormal instance region. Visual representation of the model being able to differentiate between normal and abnormal region place the test data point accordingly can be seen in figure 21

5.2.4. Sensor Localisation of DevNet

Table 8 shows the sensor localisation of DevNet, meaning the top three sensor responsible to conduct anomaly in that IMU data in the test ROS-bag file.

5.3. Results from the ensemble

In the ensemble we took the weighted average score of the best model of DevNet and Isolation Forest. The best model of DevNet gave us a mean square error of 0.05 while the best model of Isolation Forest gave us a mean square error of 0.1. It was both satisfying result for our dataset but since it was trained on very small dataset we weren't completely optimistic about the result being a generalised rather a biased result. So we took a weighted average for both of the models returned score. Our evaluating metrics for the ensemble

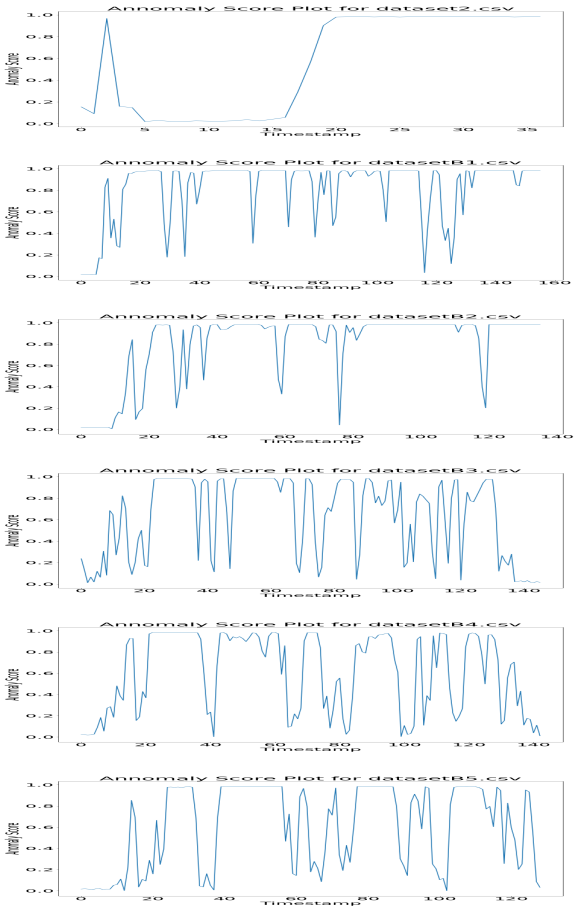


Fig. 20: **Anomaly score graph on test data instances**

was the final MSE score. The weight were generated by a linear regression method. Finally our ensemble land on a MSE of 0.031 which was lower than both the network and weight was computed in a way that made sure both model score is significant to the final score output of out network.

6. EMPIRICAL OBSERVATION

6.1. Feature Relevance

Due to the type of abnormality present in the datasets, only motion data was taken into account for the system. Motion data includes position, orientation, linear velocity, angular velocity and linear accel-

Table 9: Top three responsible sensors of all abnormal ROS-bag accoding to DevNet

Rosbag Name	Sensors Rank(from highest to lowest)
Dataset2	ori_x, vel_ang_y, acc_x
DatasetB1	ori_x, acc_z , vel_ang_y
DatasetB2	ori_x, acc_x, acc_z
DatasetB3	acc_z, ori_x, vel_ang_y
DatasetB4	acc_z, acc_x, ori_x
DatasetB5	acc_z, ori_x, acc_x

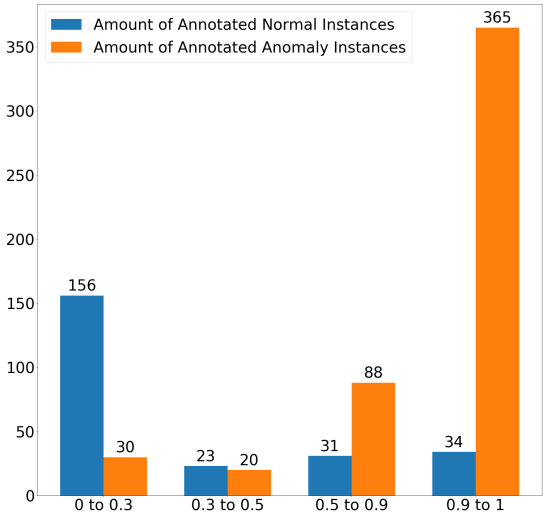


Fig. 21: **Dissonance between normal and anomaly instances**

Fig. 22: Dissonance between normal and anomaly instances

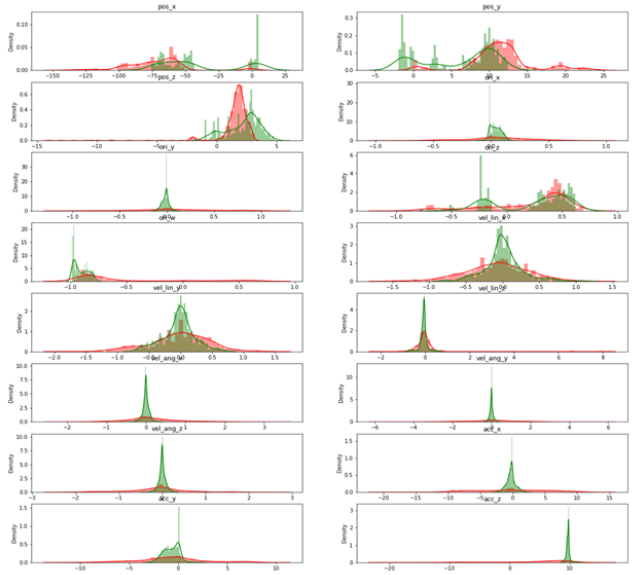


Fig. 23: Distribution plot of 5 features for normal(green) and abnormal(red) dataset

eration data. Empirical analysis of output from the proposed network model reveals that removing the position and velocity data increases the model performance.

Distribution plot 23 reveals that in case of normal datasets, distribution of the features have a low variance, but abnormal dataset has a greater variance and a flatter curve, only with the exception of position data. Position data has no clear distribution pattern. A plausible explanation is that positional data is naturally not bound in the dataset, and during runtime as seen from the video the drone position with respect to navigation frame does not vary much. Thus dropping positional x,y,z values gave a higher accuracy.

Using univariate isolation forest approach, a significant discrepancy was observed in linear velocity data as well 24 (Decision values for orientation clearly 1 for normal datasets and oscillates from 0 to 1 for abnormal datasets, but such characteristic isnt clear for velocity). This implies that linear velocity data is also an irrelevant feature for the given problem statement. This is also due to the fact that the MAV has no significant change in linear velocity over the normal or abnormal datasets. Another reasoning is that position and velocity data are either GPS data (low resolution) or derived from imu data, which can have large accumulated errors due to sensor bias and noise. [19]

6.2. Run-time Performance

Models trained on image in both Isolation forest 3.1 and DevNet 3.2 approaches are fairly complex and has a significant amount of run-time. However, model accuracy does not improve much with image included in the network. We have also observed that network generated abnormality score based on images is closely related to the manually annotated abnormality score explained in section 2.3. Later in that part, we have also shown that the annotated score again has a close correlation with the IMU orientation data. It can be easily explained as orientation is the most apparent feature of the images, and because of the abnormalities present in the datasets, most of the relevant information of images is present in the IMU orientation. As a result, it was safe to drop the image data from the final network, which not only gives a higher accuracy but also an overall faster system. To keep the system faster we keep the sensor localization duty from DevNet scheme in **Section** 5.2.4 as isolation forest takes extra time to run univariate anomaly detection.

7. FUTURE WORK

The proposed approach of the paper was driven by the IEEE Signal Processing Cup 2020 problem statement and given dataset. Consequently the workflow was hugely restricted by available data volume and quality. As mentioned in section 5.1.2, the quiver plot of the anomalous datasets gave poor results since the video frame rate is very low (2 Hz) and not optimum for optical flow based approach. However optical flow approach has significant importance in activity recognition i.e. abnormality recognition in autonomous systems. Similarly the IMU data rate was also very low (3.9 Hz) compared to several UAV datasets available such as The Zurich Urban Micro Aerial Vehicle Dataset [20], EuRoC MAV Dataset [21], The Blackbird Dataset [22]. These datasets include similar data (IMU, Image) and in a much larger quantity. Due to different flight mode, the larger datasets could not be integrated in our proposed method. However models built on a wide variety of flight modes would be more dynamic and robust. A time series feature extraction method was attempted while solving the problem statement using the python TsFresh module. The module extracts various features such as mean, variance, standard deviation, rms, zero crossing rate, derivatives, peak counts, DFT coefficients, wavelet coefficients by taking in windows of IMU data. By empirically choosing relevant features, designing another network would be possible. But this windowed feature extraction method was ultimately dropped due to the

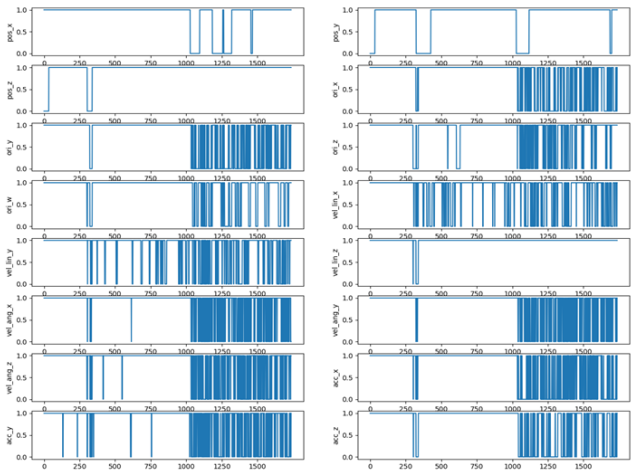


Fig. 24: Univariate isolation forest output for all datasets

requirement of time localizing abnormalities. Nevertheless the time series feature extraction has a huge potential.

A dynamic modelling of the MAV used in the experimentation would also provide a better abnormality detection network. Due to the low frame rate and short data-lengths, there were some confusions regarding the actual motion of the MAV. In some cases, the MAV showed maneuvers that are dynamically impossible to execute without crashing. Such high speed motions would be easier to interpret by a dynamic model of the system.

The network could also be improved by manually collecting more data similar to the provided dataset and introducing a wider variety of abnormalities to train the network.

Data augmentation methods can provide more training data to the network from the available datasets. Such data augmentation on time series data was proposed by KM Rashid et. al [23], O Steven et. al [24]. The proposed augmentation methods are rotation, permutation, additive noise, magnitude scaling, magnitude warping, time warping, cropping, time reversal, mirrored motion etc. However these augmentations were not included in the proposed method because augmenting the normal datasets could result in abnormal motion characteristics which would not be verifiable without a dynamic model. Again augmented IMU data can provide better results in abnormality detection.

8. ACKNOWLEDGEMENT

We would like to thank the department of BME and Brain Station 23 (Dhaka, Bangladesh) for supporting this research. The TITAN Xp GPU used for this work was donated by the NVIDIA Corporation.

9. REFERENCES

[1] V. Chandola, A. Banerjee, and V. Kumar, "Outlier detection: A survey," *ACM Computing Surveys*, vol. 14, p. 15, 2007.

[2] H.-P. Kriegel, P. Kröger, E. Schubert, and A. Zimek, "Loop: local outlier probabilities," in *Proceedings of the 18th ACM conference on Information and knowledge management*, 2009, pp. 1649–1652.

[3] A. Likas, N. Vlassis, and J. J. Verbeek, "The global k-means clustering algorithm," *Pattern recognition*, vol. 36, no. 2, pp. 451–461, 2003.

[4] D. A. Reynolds, "Gaussian mixture models." *Encyclopedia of biometrics*, vol. 741, 2009.

[5] S. Xiang, F. Nie, and C. Zhang, "Learning a mahalanobis distance metric for data clustering and classification," *Pattern recognition*, vol. 41, no. 12, pp. 3600–3612, 2008.

[6] M. Goldstein and A. Dengel, "Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm," *KI-2012: Poster and Demo Track*, pp. 59–63, 2012.

[7] 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.

[8] G. Pang, C. Shen, and A. van den Hengel, "Deep anomaly detection with deviation networks," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 353–362.

[9] 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.

[10] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2. IEEE, 2006, pp. 1735–1742.

[11] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation-based anomaly detection," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 6, no. 1, pp. 1–39, 2012.

[12] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8697–8710.

[13] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.

[14] S. Targ, D. Almeida, and K. Lyman, "Resnet in resnet: Generalizing residual architectures," *arXiv preprint arXiv:1603.08029*, 2016.

[15] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Thirty-first AAAI conference on artificial intelligence*, 2017.

[16] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[18] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[19] M. Kok, J. D. Hol, and T. B. Schön, "Using inertial sensors for position and orientation estimation," *arXiv preprint arXiv:1704.06053*, 2017.

[20] A. L. Majdik, C. Till, and D. Scaramuzza, "The zurich urban micro aerial vehicle dataset," *The International Journal of Robotics Research*, vol. 36, no. 3, pp. 269–273, 2017.

[21] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The euroc micro aerial vehicle datasets," *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

[22] A. Antonini, W. Guerra, V. Murali, T. Sayre-McCord, and S. Karaman, "The blackbird dataset: A large-scale dataset for uav perception in aggressive flight," *arXiv preprint arXiv:1810.01987*, 2018.

[23] K. Rashid and J. Louis, "Window-warping: A time series data augmentation of imu data for construction equipment activity identification," in *ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction*, vol. 36. IAARC Publications, 2019, pp. 651–657.

[24] O. Steven Eyobu and D. S. Han, "Feature representation and data augmentation for human activity classification based on wearable imu sensor data using a deep lstm neural network," *Sensors*, vol. 18, no. 9, p. 2892, 2018.