SPCUP 2020: IEEE Signal Processing Cup 2020

Daniele Scapin, Gabriele Ferraresso, Marco Perin, Piero Simonetto, Riccardo Lorigiola

***Abstract*—In this letter, we proposed our method to solve the problem of unsupervised abnormality detection. It involves four main steps : peaks and abnormalities during data acquisition saerch, anomaly check and system values update.**
First and second steps take place at the same time during program flow but they serch autonomously the anomaly. Peaks serch is implemented with the use of polyfit and Kalman evaluation, for prediction of the measures and calculation of the percentual change. Instead abnormalities during data acquisistion uses a modified version of the isolation forest to verify if during the measures are occurred discrepancies.
The third part leverages of the concept that if every sensor in the autonomous system reports an anomaly, it means that the behavior is not anomaly. The anomaly check uses a tree structure to get feedback from each sensor.
The system values update is used to improve respose time and to update the variables needed in other sections.

***Index Terms*—Abnormalities detection, peaks search, Kalman filter, isoaltion forest, tree structure**

I. Introduction

TO DO
(Descrizione dettagliata flow del programma, associata al flow diagram)

II. Peaks search: FindPeaksWrapper

$FindPeaksWrapper(...)$ is used to find noticeable variation in the sensors data flow. The program can be divided in four main elements: polyfit, Kalman filter, peak presence and variables update.

A. Polyfit

For each sensor, for each new data acquisition the variation between the measure and the prediction is checked by $poly\_fit(...)$ function (it uses $polyfit(...)$ and $polyval(...)$ Matlab functions). This evaluation is done only if the number of elements to analyse is greater than degree of polyfit evaluation plus three (in the program degree plus three). This restriction is caused by the necessity of the presence of one element to verify and of degree plus two elements for $polyfit(...)$(plus two and not one because so the parametric estimation has al least a degree of freedom).

B. Kalman Filter

The Kalman filter is used only when the cosidered data is space, velocity and acceleration (angular or linear) and the acceleration is almost constant. The Kalman filter needs the state transition matrix to be built (obtained from the differential equations of the model). In this case the model is a black box model thus it can't be described exactly. For these reasons the Kalman filter is used only on space, velocity and acceleration with uniform acceleration. The Kalman filter is used anyway because if the autonomous system makes continuous and without variations paths/movements, it will generate a more accurate evaluation than $poly\_fit$. The filter is activated by the evaluation of slope of regression line (the coefficient is generated inside $poly\_fit$ function). The covariance matrix ($Pn\_...$) is initialized as 10 $eye(...)$ and update during subsequent cycles. The covariance matrix of the measures (Q) is set by $var2\_error$ vector placed diagonally (the motivation of these setting is explained in the next sections).

C. Peaks Presence

The peak presence is verified by calculating if the percentage of change between values expected and measured exceeds delta $[\%]$. The value of delta is the maximum between a pre-set value (gap), the average percentage change of previous check and percentage error change of polyfit evaluation. This check is necessary because if signal noise is very loud, the function avoids reporting every measures as anomaly.

D. Internal variables update

Inside $FindPeaksWrapper(...)$ there are three values (for each type of signal) that are used both as input and as output ($Pn\_2, varp\_error, var2\_error$) because in each cycle they are updated. Pn_2 is necessary for Kalman filter evaluation and in each cycle it is updated whitin the function itself. $varp\_error$ is average percentage change between measure and prediction (error) (related to prediction). $var2\_error$ is average squared change between measure and prediction (error) and it is used in the covariance matrix of the measure. It is used as covariance matrix of the measure because assuming the noises as gaussian noises (processes is subjected to ambient, thermal and internal noise)

$$y_{measure} = y_{real} + e_{measure} \quad e_{measure} \sim N(0, \Sigma_{measure})$$
$$y_{estimate} = y_{real} + e_{estimate} \quad e_{estimate} \sim N(0, \Sigma_{estimate})$$

The measures of different axis are considered unrelated (covariance matrix is diagonal). Unidimensional estimation of variance

$$e = y_{measure} - y_{estimate} \qquad N(0, \theta)$$
$$e_i = y_{measure_i} - y_{estimate_i}$$
$$l_e(\theta) = -\log\left(\prod_{i=1}^{n} p_{e_i}(\theta)\right)$$
$$... \quad = \frac{n}{2}\log(2\pi) + \frac{n}{2}\log(\theta) + \sum_{i=1}^{n} \frac{e_i^2}{2\theta}$$
$$\frac{\partial l_e(\theta)}{\partial \theta} = \frac{n}{2\theta} - \sum_{i=1}^{n} \frac{e_i^2}{2\theta^2}$$
$$\hat{\theta} = \frac{\sum_{i=1}^{n} e_i^2}{n}$$

This competition is sponsored by the IEEE Signal Processing Society and MathWorks

All author is with the University of Padue, Padova, ITA

E. Input - Output

Detailed specification in $FindPeaksWrapper(...)$

Input

- t – time vector
- y – vector of values of data_type element
- data_type – type of data
- degree – max degree during poly fit evaluation
- num – number of elements evaluating during polyfit
- ap – maximum permissible percentage error
- gap_sva – max variation to identify a constant element

Output

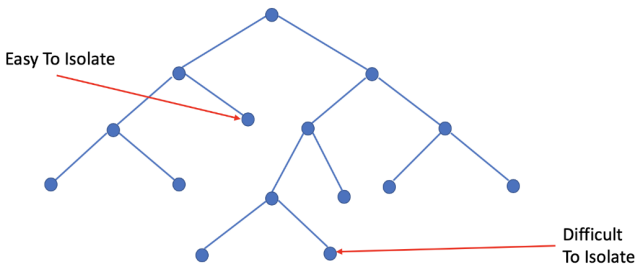
- already_analysed – true if all values of the corresponding data_type are already analysed
- anomaly – vector of all the anomaly of the corresponding data_type
- error – error of the last element ($y_{measured} - y_{predicted}$)
- y_next – prediction of the last element

III. Isolation Forest

$Isolationforest$ is a Statistical method to analyze small dataset and find anomalies. An anomaly is when an observation deviates so much from others. In this case, anomalies are observation that not correspond to the right functioning of the Drone. We used this algorithm to analyze data when the observation received are approximately constant, we took this decision because we want to exclude the anomalies that could be find on the right functioning of the Drone. For example if we are height parameter we don't add to isolation forest the observation of the drone when it's taking off, because there is a very high variation of the observation in a small time and this can be identified as a false positive anomaly. However, when the drone is flying, a high variation of the height in a small time is very likely an anomaly. This function can be divided in four main elements: $IsolationTree$, $iForest$ and $AnomaliesFinder$

A. Isolation Tree

This function randomly creates a binary tree from a given dataset of observation. It randomly take a subsample of the data. Our data are the observation and every observation can have one ore more dimension. The function randomly select a dimension and a value in that dimension and divides observation in two groups (observation with a greater or equal value and observation with lower value). This process is recursively repeated until all observation are isolated. Isolated observation are on the leaves of the tree.



B. iForest

This function creates a lot of different isolation trees that's why it's called forest. Every tree is different because it's randomly generated by $IsolationTree$ function, however the observations inside every tree are always the same. $iForest$ generate a forest that can be analyzed from $AnomaliesFinder$ function.

C. Anomalies Finder

This is the most important function of $IsolationForest$ because it's used to find anomalies. It takes the forest and calculates the average height of every observation and anomalies will have an average height lower than normal observations.

Now the anomaly score s of an instance x on a database of n instances is defined as :

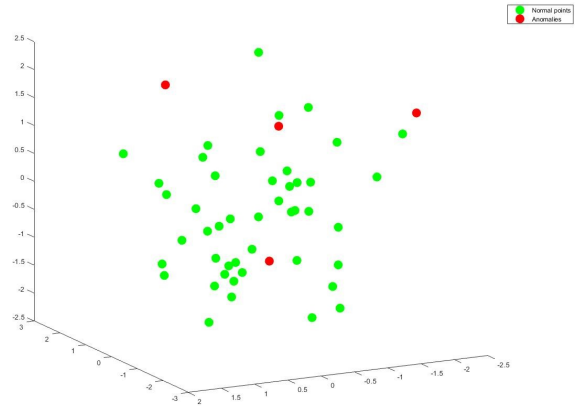
$$s(x,n) = 2^{\frac{E(h(x)}{c(n)}}$$

$$where \qquad c(n) = 2H(n-1) - \frac{n-1}{n}$$

$$and \qquad H(i) = \ln(i) + 0.5772156649$$

$s(x,n)$ give us an information about the observation:

- If $s(x,n)$ is lower than 0.5, x is a normal value
- If $s(x,n)$ is closer to 1, x probably is an anomaly observation
- If $s(x,n)$ is around 0.5 for all observation it's safe to assume that the dataset doesn't contain anomalies.



The main disadvantage of Isolation Forest is Swamping. Swamping is when normal instance are very closed to anomalies making them hard to be isolated. This problem can be minimized by subsampling the dataset.

D. Input

- NumTree: Number of trees inside the forest
- maxPoint: Maximum points inside every tree of the forest (for subsampling)
- sk: Anomaly threshold
- type: Type of the new element
- newEl: New point of the forest

E. Output

- Last: True if newEl is abnormal, false otherwise
- Abnormal: True if there are abnormalities, False otherwise
- posOfAnomaly: Index of abnormal points
- h: Average height of each point in the forest
- s: Anomaly score of each point in the forest

REFERENCES

[1] Isolation Forest.
[2] Hariri Forest.
[3] Swamping and masking in anomaly detection in isolation forest.
[4] Anomaly score of isolation forest.
[5] Anomaly detection with isolation forest visualization.

IV. ANOMALY CHECK

This part contains classes and functions to structure data in a prioritized way, to avoid unnecessary controls over sensor that are less likely to give data representing a true anomaly

V. SYSTEM VALUES UPDATE

A. To do