CASPER: Context-Aware Anomaly Detection System for Industrial Robotic Arms

Hakan Kayan
Department of Computer Science and Informatics
Cardiff University
Cardiff, United Kingdom
kayanh@cardiff.ac.uk

Omer Rana
Department of Computer Science and Informatics
Cardiff University
Cardiff, United Kingdom
ranaof@cardiff.ac.uk

Pete Burnap
Department of Computer Science and Informatics
Cardiff University
Cardiff, United Kingdom
burnapp@cardiff.ac.uk

Charith Perera
Department of Computer Science and Informatics
Cardiff University
Cardiff, United Kingdom
pererac@cardiff.ac.uk

***Abstract*—With the arrival of industry 4.0, industrial control systems are converted into "smart" industrial cyber-physical systems that depend on high interconnectivity enabled by ubiquitous applications. As these applications can significantly reduce maintenance and supervision costs, the integration of these applications is done with the "cost" being the focus overlooking the security aspect that suffers from the vulnerabilities that occurred due to increased attack surface. The adversaries aim to create physical alterations by exploiting these cyber vulnerabilities via so-called "cyber-physical" attacks. In this work, we introduce CASPER, a context-aware ubiquitous machine learning-based anomaly detection infrastructure that utilizes ubiquitous computing to detect anomalies of an industrial robotic arm. CASPER monitors the robotic arm's movements to ensure the arm follows a predetermined trajectory. The CASPER can reach an accuracy and F1 score of 97% which is promising for an industrial domain. We modify the joint velocity of an industrial robotic arm to create anomalies which we detect via CASPER.(Demo Video)**

***Index Terms*—anomaly detection, predictive maintenance, ubiquitous computing**

I. Introduction

The legacy industrial control systems (ICS) are being converted into industrial cyber-physical systems (ICPS) adopting a highly interconnected infrastructure enabled by the utilization of disciplines such as the Internet of Things (IoT), cyber-physical systems (CPS), and cloud computing. The increased attack surface is the main con of this increased automation, digitalization, and networking. The recent industrial cyber incidents [1] show that the adversaries targeting industrial systems cause physical impact by exploiting vulnerabilities occurring due to the integration of information technology (IT) systems into operational technology (OT) systems. The network-only cybersecurity mechanisms fail to detect physical alterations as the physical data generated on the edge can be spoofed by adversaries to misguide the monitoring systems.

The physicality of ICPS is examined under two mutually inclusive main disciplines: fault diagnosis, and predictive maintenance. The traditional methods apply statistical analysis to signals [2] generated by industrial equipment (e.g., current, and voltage). These methods have two big disadvantages: i) they require a deep knowledge about the system which is hard to obtain as current ICPS are heterogeneous, ii) they are not scalable as a change in the system requires redesigning the whole solution due to parameters being dependent. Data-driven approaches [3] are getting popular as they offer a solution to these problems by being dependent only on the data characteristics. In this work, we apply a data-driven approach to inertial measurement unit (IMU) data that we gather from an industrial robotic arm to detect movement-related anomalies via the open-source machine learning (ML) pipeline [4].

We make the following contributions: i) we propose an open-source[1] end-to-end context-aware anomaly detection system that includes edge (Nicla Sense ME), fog (human-machine interface (HMI) based on Raspberry Pi 4B (Pi-HMI)), and central nodes (local PC/cloud) (Fig. 1), ii) we introduce the CASPER dataset, iii) we test the proposed system on Universal Robots UR3e which is an industrial robotic arm.

II. System Architecture

We detect movement-based anomalies of an industrial robotic arm via a low-cost anomaly detection system (that we call CASPER) that utilizes Bluetooth Low Energy (BLE) to transmit the data from edge to fog. We attach an edge development board which can generate IMU data via built-in accelerometer, gyroscope, and magnetometer sensors. We teach the normal movement of an arm to a neural network model which utilizes 1D convolutional neural network (1D-CNN) layers by feeding non-anomalous IMU data during

[1]https://github.com/hkayann/CASPER-PerCom

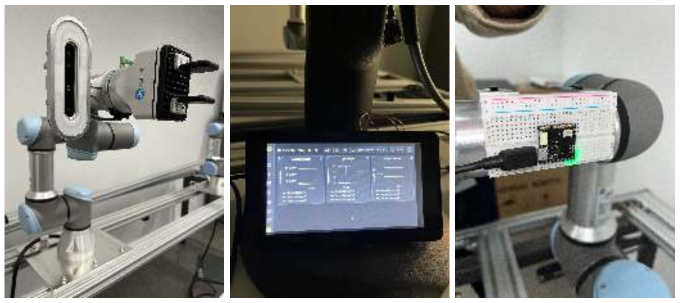
(a) UR3e (b) Pi-HMI (c) Nicla Sense ME

Fig. 1: The main system components.

training. We utilize Tensorflow (tensorflow.org) for this particular work.

A. System Components

Edge node. We use a Nicla Sense ME as an edge development board. It has a 64 MHz Arm Cortex microcontroller, built-in IMU (accelerometer, gyroscope, and magnetometer), gas, pressure, temperature, and humidity sensors, and a BLE chip. It is relatively small (22,86 x 22,86 in mm), micro USB-powered and programmable, operates in low power (15 mAh during the test). It also has a memory built-in but in this demo, as we transmit the IMU data over BLE we do not store any data on the edge. We run an Arduino sketch that generates and transmits IMU data from Nicla to fog node where we use Node-RED to receive such data. *Fog node.*We build an HMI based on a Pi (Pi-HMI). It displays the IMU data from the Nicla Sense ME and forwards it to a local PC (a data science workstation) where we store the data. *Central node.* The local PC that we use for training and testing has an NVIDIA RTX A6000 48GB GDDR6 GPU.

B. The Testbed and Use Case Scenario

We have two industrial robotic arms manufactured by Universal Robots (UR3e). While these arms are introduced as cobots they are capable of completing tasks standalone as well. These arms can perform many tasks [5] ranging from pick and place, and screw-driving to surgical operations. For this particular demo, we design a pick and place task as it is one of the most common processes seen in manufacturing systems. The task we implement is repetitive, continuous, has a certain frequency, and does not require any human interaction. Fig. 2 demonstrates the use case scenario.

III. Anomaly Detection System

This section demonstrates the proposed solution. The anomaly detection system we build depends on an ML model which is based on 1D-CNN layers. Usually, the Recurrent Neural Network (RNN) layers are applied to time series data as they have a "memory" [6]. However, the computational complexity of RNNs makes their usage questionable for resource-constrained environments. 1D-CNNs are also known to be effective against time series data thanks to strong 1D

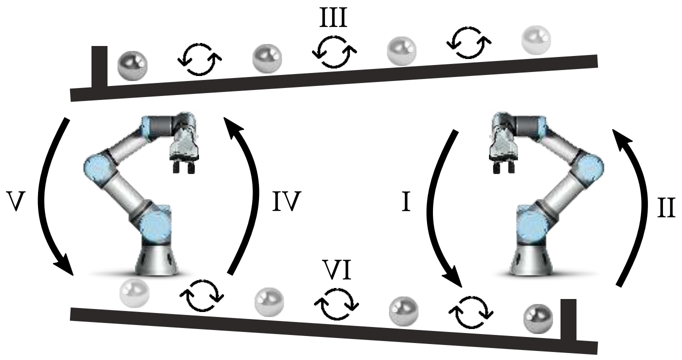

Fig. 2: I) The arm picks the ball, II) drops it to an inclined platform, III) the ball rolls down over the platform until it hits to a stopper, IV) then it is picked by other arm and so on.

temporally correlated structure [7]. For this particular demo, we utilize 1D-CNN as our future work aims to bring anomaly detection to the resource-constrained edge environment.

A. The Input Data Structure and the Dataset

The choice of an ML algorithm depends on the input data characteristics. In this work, we have 3 data sources where each of which is 3-dimensional as the accelerometer, gyroscope and magnetometer are 3-axis (x, y, z) sensors. Thus, in total, we have 9 individual inputs combined into one multivariate input. As we try to minimize the computational complexity, we do multivariate analysis [8]. We apply min-max normalization with the range of 0 to 1 to avoid the domination of higher-scale inputs. While applying normalization, min/max values are computed via train set only. The whole train set is non-anomalous while the validation and test sets include 50% of anomalous data. The test is 24 hours and the data frequency is around 20Hz.

B. The Machine Learning Model

CASPER utilizes 1D-CNN layers with ReLU activation function to extract exclusive features that occur thanks to high temporal correlation. We stack two 1D-CNN layers to access deeper features. They are followed by pooling layers to prevent overfitting while reducing the computational cost as our model is prone to overfitting due to input data being periodic while having a train set with zero anomalies. Then, we pass the output of pooling to flatten layer to convert the multi-dimensional output to two-dimensional. Finally, we output 9 predictions per given input. Fig. 3 demonstrates the structure of the model with given hyperparameters.

C. Creating Realistic Anomalies

The desired way of creating anomalies is to simulate a real cyberattack. However, due to the possibility of damaging high-cost equipment we choose to create controlled anomalies by changing the joint velocity of the arms. The arm consists of 6 joints in total. The arm moves its joints in synchronization via applying forward kinematics per a given "pose" parameter which includes the tool center point (TCP) and the joint angles.

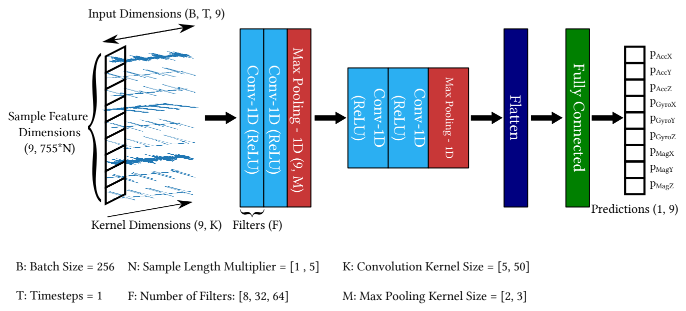

Fig. 3: The structure and the hyperparameters of model.

The anomalies we create range from 10% to 100% increase and 5% to 50% decrease in joint velocity where the default joint velocity is 1.04 rad/s ($\approx$ 60 deg/s).

D. Detecting Anomalies

Training: We train the model with non-anomalous data while applying grid search to hyperparameters to figure out the most efficient model. The initial hyperparameter to analyze is the input sample length. Due to high-frequency data, we utilize input windows rather than single data point inputs.

We apply autocorrelation to find an input window length with highest Pearson correlation coefficient (indicates periodicity). For the other hyperparameters (e.g., convolution kernel size, number of filters), we use the commonly accepted parameters. We decide the efficiency of the model by looking at train, validation, and test losses calculated via mean squared error (MSE). Fig. 4 presents the loss graph, Table I presents the best performing hyperparameters.

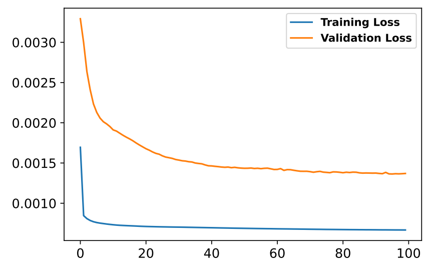

Fig. 4: The loss graph.

TABLE I: The Best Performing Hyperparameters

Number of Filters	Max Pooling Kernel Size	Convolution Kernel Size	Sample Window Length	Batch Size
32	3	5	755	256

Non-ML Baseline: We compare the model to a non-ML baseline to justify implementing an ML-based approach. We calculate the overall mean absolute error (MAE) of the sample windows shifted by the correlation coefficient. We see that the ML model beats the common-sense approach with a high margin.

Detecting Anomalies: To convert anomaly labels into anomalous windows, we count their number per window. We consider such a window as anomalous if more than half of it is formed from anomalous points. Then, we calculate absolute residuals per sample window. We decide the threshold via applying a grid search that maximizes the F1-score as our work is in an industrial domain where "false" conditions matter. Anything above the threshold is accepted as anomalous. We can see from the confusion matrix (see Fig 5) that the proposed model has around 99% accuracy and F1-score thus justifying the use of CNN for periodic time series data.

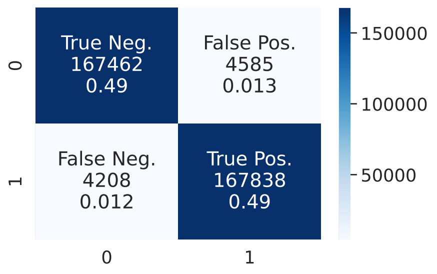

Fig. 5: The confusion matrix.

IV. Conclusion

Ubiquitous computing is revolutionizing many domains including ICPS as wireless communication technologies are becoming reliable. Anomaly detection is crucial for applying predictive maintenance. Thus the development of ubiquitous anomaly detection systems that can efficiently work in the resource-constrained environment is significant to build more secure ICPS. In this demo, we demonstrate CASPER which is a ubiquitous context-aware anomaly detection system that utilizes 1D-CNN layers. CASPER can detect movement-based anomalies with more than 97% accuracy. The future work includes adding edge anomaly detection capability.

References

[1] H. Kayan, M. Nunes, O. Rana, P. Burnap, and C. Perera, 'Cybersecurity of Industrial Cyber-Physical Systems: A Review', ACM Comput. Surv., vol. 54, no. 11s, pp. 1–35, Jan. 2022.
[2] Z. Gao, C. Cecati, and S. X. Ding, 'A Survey of Fault Diagnosis and Fault-Tolerant Techniques—Part I: Fault Diagnosis With Model-Based and Signal-Based Approaches', IEEE Trans. Ind. Electron., vol. 62, no. 6, pp. 3757–3767, Jun. 2015.
[3] S. Yin, S. X. Ding, A. Haghani, H. Hao, and P. Zhang, 'A comparison study of basic data-driven fault diagnosis and process monitoring methods on the benchmark Tennessee Eastman process', Journal of Process Control, vol. 22, no. 9, pp. 1567–1581, Oct. 2012.
[4] H. Kayan, Y. Majib, W. Alsafery, M. Barhamgi, and C. Perera, 'AnoML-IoT: An end to end re-configurable multi-protocol anomaly detection pipeline for Internet of Things', Internet of Things, vol. 16, p. 100437, Dec. 2021.
[5] M. E. Moran, 'Evolution of robotic arms', J Robotic Surg, vol. 1, no. 2, pp. 103–111, Jun. 2007, doi: 10.1007/s11701-006-0002-x.
[6] A. Sherstinsky, 'Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network', Physica D: Nonlinear Phenomena, vol. 404, p. 132306, Mar. 2020, doi: 10.1016/j.physd.2019.132306.
[7] Y. LeCun, Y. Bengio, and T. B. Laboratories, 'Convolutional Networks for Images, Speech, and Time-Series', p. 14.
[8] R. Wan, S. Mei, J. Wang, M. Liu, and F. Yang, 'Multivariate Temporal Convolutional Network: Deep Neural Networks Approach Multivariate Time Series Forecasting', Electronics, vol. 8, no. 8, p. 876, Aug. 2019.