University of Novi Sad
Faculty of Technical Sciences
Department of Production Engineering

14th International Scientific Conference

# MMA 2021

## FLEXIBLE TECHNOLOGIES


## PROCEEDINGS

Novi Sad
Serbia
September 23-25, 2021

PROCEEDINGS OF THE 14th INTERNATIONAL SCIENTIFIC CONFERENCE

**MMA 2021 - FLEXIBLE TECHNOLOGIES**

**Novi Sad, 2021**

*Publisher:* **FACULTY OF TECHNICAL SCIENCES**
**DEPARTMENT OF PRODUCTION ENGINEERING**
**21000 NOVI SAD, Trg Dositeja Obradovića 6**
**SERBIA**

*Organization of this Conference was approved by Educational-scientific Council of Faculty of Technical Sciences in Novi Sad*

*Editor:* Dr Rade Doroslovački, Full Professor, Dean

*Technical treatment and design:* Dr Milenko Sekulić, Full Professor
Dr Borislav Savković, Associate Professor
Dr Dragan Rodić, Assistant Professor
Dr Miroslav Dramićanin, Assistant Professor
M. Sc. Anđelko Aleksić, Teaching assistant
M. Sc. Nenad Kulundžić, Research associate

*Manuscript submitted for publication:* September 15, 2021
*Printing:* 1st
*Circulation:* 200 copies

*CIP classification:*

CIP - Каталогизација у публикацији
Библиотека Матице српске, Нови Сад

621.7/.9(082)
621.9(082)

**INTERNATIONAL Scientific Conference MMA 2021 - Flexible Technologies (14 ; 2021 ; Novi Sad)**
Proceedings / 14th International Scientific Conference MMA 2021 - Flexible Technologies, Novi Sad, September 23-25, 2021 ; [editor Rade Doroslovački]. - 1st ed. - Novi Sad : Faculty of Technical Sciences, 2021 (Novi Sad : FTN, Graphic Centre Grid). - V, 254 str. : ilustr. ; 30 cm

Tiraž 200. - Tekst štampan dvostubačno. - Bibliografija uz svaki rad. - Registar.

ISBN 978-86-6022-364-9

a) Производно машинство - Зборници. b) Метали - Обрада - Зборници
COBISS.SR-ID 45959689

*Printing by:* FTN, Graphic Centre
GRID, Novi Sad

*Financing of the Proceedings was sponsored by the Ministry of Education, Science and Technological Development of the Republic of Serbia and supported by the Provincial Secretariat for Higher Education and Scientific Research of AP Vojvodina.*

MMA 2021
FLEXIBLE TECHNOLOGIES

14th INTERNATIONAL SCIENTIFIC CONFERENCE
NOVI SAD, SERBIA, SEPTEMBER 23-25, 2021

**Nedeljković, D., Jakovljević, Ž.**

# IMPLEMENTATION OF CNN BASED ALGORITHM FOR CYBER-ATTACKS DETECTION ON A REAL-WORLD CONTROL SYSTEM

***Abstract:*** *The emergence of the Industry 4.0 concept leads to crucial changes in manufacturing by building advanced industrial systems and applications based on Cyber-Physical Systems (CPS), as the core of this approach. Using CPS, manufacturing assets are designed in the form of systems of systems through interconnection of smart devices with integrated computation and communication capabilities. System control logic is distributed over a large number of resources, and its performance is achieved through their coordinated work and ubiquitous communication raising the issue of cyber-attacks by malicious adversaries. Since cybersecurity within industrial control systems is safety related, it is necessary to timely detect cyber-attacks on industrial assets; for these purposes a number of different approaches have been developed. As a technique of choice, deep learning (DL) based methods emerge, providing good online performances. In this work, we focus on the implementation of a DL based cyber-attack detection algorithm on an electro-pneumatic positioning system containing smart sensor and smart actuator. In particular, we employ cyber-attack detection procedure based on 1D Convolutional Neural Network (CNN) at the local controller of the smart actuator. The implemented algorithm can successfully detect cyber-attacks in real-time, as will be experimentally demonstrated.*

***Keywords:*** *Industrial Control Systems, Cyber-Physical Systems, Convolutional Neural Network, Cybersecurity*

## 1. INTRODUCTION

Cyber-Physical Systems (CPS) become the core components of the Industrial Control Systems (ICS) not only at manufacturing shop floors, but also in power systems, water treatment plants, and other critical infrastructures [1]. CPS based smart devices (sensors, actuators...) integrate computational and communication capabilities into physical processes and enable distribution of control tasks, where the control system is realized through their intensive communication and interoperability. In this way Industrial Internet of Things (IIoT) is introduced, and consequently smart devices are often connected to the internet, instead of communication only within the industrial plant [2]. Widespread communication opens up various security related concerns such as the occurrence of malicious cyber-attacks that can compromise system operation or even endanger human life. Creating an intrusion detection method that provides ICS operation in a safe manner is a challenging task. Due to the inherent differences in IT systems and ICS, traditional IT detection mechanisms cannot successfully handle all security issues of ICS. For instance, noisy behavior of a physical process can result in a high rate of false positive and/or low rate of true positive attack detections [3].

A basic control loop of ICS employs SCADA (Supervisory Control and Data Acquisition) systems, controllers (PLCs, microcontrollers), sensors, and actuators to manage some physical process (Fig. 1). The controller interprets the sensor measurements $x_t$, and based on the control task transmits the commands $y_t$ to the actuator. The actuator executes corresponding action and closes the control loop.

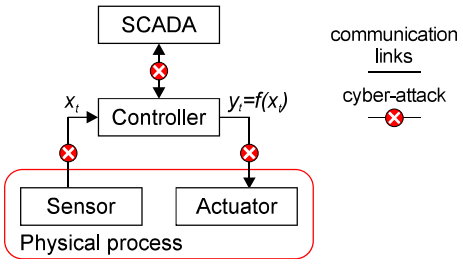


Fig. 1. Simplified scheme of ICS

Communication links sensor/controller and controller/actuator represent vulnerable points for cyber-attacks by different adversaries. To reach the desired goal, attackers maliciously modify the sensor output and/or actuator input signal through communication channels.

For the design of Intrusion Detection Systems (IDS) within ICS, data centric approaches are most frequently employed. Within these approaches, the behavior of the system in normal operation (without attacks) is modeled based on the data acquired from the system operating in isolated conditions. Once the model is generated, it is integrated in ICS on the receiving side and the attacks on the communication links that alter transmitted data are detected based on the discrepancy between modeled and received signal values. The model in normal operation can be created using different techniques such as dynamic neural network combined with integral sliding-mode attack compensator [4], support vector machines [5], recurrent neural networks [6], multi-layer perceptron [7], timed automata [8], Convolutional Neural Networks (CNN) [9].

In this paper we present a procedure and results of the implementation of a cyber-attack detection algorithm on ICS. In particular, we show how CNN

based algorithm for intrusion detection can be applied on a low level controller of CPS as a part of the electro-pneumatic positioning system. We chose 1D CNN as a suitable technique for IDS design due to its relatively simple configuration and real-time applicability.

The rest of the paper is organized as follows. In Section 2, electro-pneumatic positioning system is briefly described. Section 3 presents the developed 1D CNN based cyber-attack detection algorithm. The implementation of the algorithm on a real-world installation is presented in Section 4. Finally, conclusions and future work guidelines are provided in Section 5.

## 2. ELECTRO-PNEUMATIC POSITIONING SYSTEM

Electro-pneumatic positioning system DisEPP that we will consider in this paper consists of two CPS (Fig. 2): 1) a smart actuator (pneumatic cylinder controlled by electro-pneumatic air pressure regulator on one, and mechanically controlled air pressure regulator on the other side that enable the desired motion of the cylinder) augmented with local controller LC1 and 2) smart sensor (magnetic linear encoder placed along cylinder) augmented with LC2. Both LCs are based on ARM Cortex-M3 running at 96 MHz [10], extended with IEEE 802.15.4-compliant wireless transceiver Microchip MRF24J40MA [11]. The control task is distributed between LC1 and LC2, where LC1 has the desired trajectory at input and implements PID controller to compute the required motion of the cylinder based on the sensory signal. The information regarding desired motion is transferred to LC2 using IEEE 802.15.4 protocol, and this communication link represents a vulnerable point for cyber-attacks by different adversaries. The details regarding this system can be found in [12].

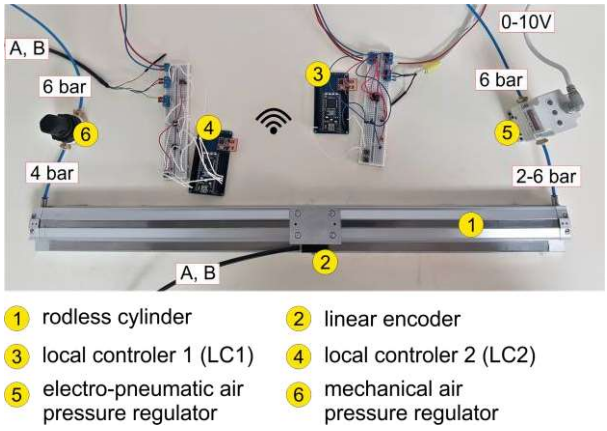


Fig. 2. Experimental setup of DisEPP

## 3. CYBER-ATTACKS DETECTION ALGORITHM

Our 1D CNN based IDS employs auto-regression, where the prediction of the current signal value $x_i$ is obtained taking into account the buffer of previous $z$ values $x_{i-z},\ldots,x_{i-1}$; in particular we use the buffer size of $z$=16. For model generation, the transmitted signal between LC1 and LC2 during normal operation was recorded using piston trajectory with positions of 50, 400, 250, 400, and 100 mm that was cyclically repeated. A total of 406,230 records were acquired. To mitigate the negative effects of abrupt changes in the signal, it is filtered using low-pass FIR filter. The whole dataset is divided into training, validation, and test part, with a ratio of 80/10/10%.

A number of different 1D CNN architectures for auto-regression were explored and the architecture consisting of nine layers presented in Figure 3 was chosen as appropriate. Network starts with two blocks containing two convolution layers, followed by a max pooling layer. The output from the max pooling layer is unrolled by flattening layer, which is connected with a dense layer. The final result is generated through the second dense layer at the end of the network. The number of filters in convolution layers is 8-16-16-32, where the filter size $m$ for all layers is 2. The downsampling rate $p$ for both max pooling layers is set to 2, with a stride $s$=2. The first dense layer involves 30 neurons, whereas the number of neurons in the second layer is determined by the network's output shape (in our case 1). Rectified linear unit (ReLU) activation function was employed in all convolutional layers. The model was trained in Python v3.8.5 using a Spyder with TensorFlow v2.3.0 in the background.

Online attack detection is done according to the following procedure. If a discrepancy between measured and estimated value exceeds the threshold ($T$) consecutively 15 times, the attack is present. The threshold value was experimentally determined and it is equal to 0.0051.

## 4. ALGORITHM IMPLEMENTATION ON A REAL-WORLD INSTALLATION

Before presenting the procedure for transformation of CNN based algorithm from TensorFlow platform to the local controller environment, we will briefly discuss the general structure of layers used in architecture from Fig. 3.

*Convolution layer,* as the basis of the CNN, performs 1D convolution utilizing a certain number of finite length filters. Output from the convolutional layer is calculated in the following way:

$$y_{i,j} = \sigma(\mathbf{w}_i * \mathbf{x}_j + b_i) \quad (1)$$

where $\mathbf{w}_i$ represents a matrix of filter coefficients and $b_i$ denotes bias of the $i$-th filter, $y_{i,j}$ is layer output and $\mathbf{x}_j$ its input vector. Operator * denotes convolution of two vectors - signal $h$ and filter $g$ and it is defined by:

$$(h * g)(k) = \sum_{k=0}^{m} h(k)\, g(m-k) \quad (2)$$

where $m$ denotes the size of filter. If the size of signal is $z$ with a filter length of $m$, after convolution an output vector of length $z$+$m$-1 is obtained. Nonlinear transformation and extraction of representative features from data in the convolution layer are performed using the activation function σ, usually ReLU.

*Pooling layer* downsamples the signal for a predefined number of samples by choosing one of $p$ samples according to the selected criterion (maximum, average value, etc.). Maximum pooling layer used in

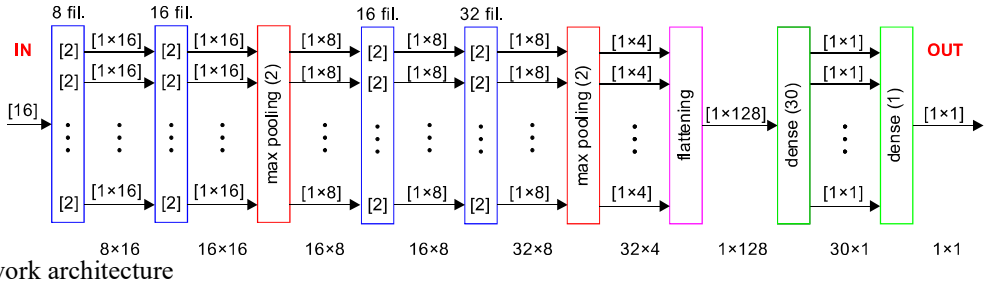

Fig. 3. Network architecture

our network can be described in the following form:

$$y_i = \max\left(x_{i\cdot s-s+1}, \ldots, x_{i\cdot s-s+p}\right), i \in \{1, \ldots, n/s\} \quad (3)$$

where $s$ represents stride.

*Flattening layer* converts the data of any format $x_{i,j}$ into a one-dimensional array $y_k$ and it is usually placed before a dense layer.

*Dense layer* learns nonlinear dependencies of data by connecting every neuron in one layer to all neurons in the next layer. The general form of the dense layer is written as follows:

$$y_i = \sigma\left(\sum_j W_{i,j}\, x_j + b_i\right) \quad (4)$$

where $W_{i,j}$ represents weight coefficient between $i$-th neuron of the current and $j$-th neuron of the previous layer. $x_j$ and $y_i$ denote layer input and output, $b_i$ is bias of the $i$-th neuron, whereas σ is the activation function.

Since the program implementation of max pooling, flattening and dense layers is straightforward, we will focus on the implementation of convolutional layers. The output from TensorFlow is a trained model composed of a certain number of parameters arranged in the predefined format. The parameters of the convolution layer $c$ are placed in a structure **S** with two members: (1) three-dimensional matrix $\mathbf{W}_c$ that includes weight coefficients and (2) bias vector $\mathbf{b}_c$.

$$\mathbf{S} = \begin{bmatrix} \mathbf{W}_c[m_c][i_c][f_c] \\ \mathbf{b}_c[f_c] \end{bmatrix} \quad (5)$$

The size of $\mathbf{W}_c$ is $m_c \times i_c \times f_c$ where $m_c$ represents filter size, $i_c$ is the number of input vectors, and $f_c$ denotes the number of filters in the current layer $c$. The order of the values in the filters is reversed (the last value is entered first), which must be taken into consideration during convolution calculation using (2).

The implementation of convolution layer that has one vector at input (such as the first layer in Fig. 3) is straightforward. However, when the input is not in the form of a single vector, the procedure becomes more complex and it is not easily observed from (1). In this case, one filter convolves all input vectors, sums the obtained values column by column, and makes one output vector. The same procedure is repeated for each filter in the layer and the output of the size $f_c \times z_c$ is created ($z_c$ is the length of input and $f_c$ the number of filters in the layer $c$). For example, for the second convolutional layer from Fig. 3 with 16 filters, the input consisting of 8 vectors of length 16 is transformed into an output of shape 16×16. Details regarding the implementation of the described procedure are given in Pseudocode 1.

The control task of both local controllers is implemented in C++ using Keil uVision5 environment [13]. Therefore, to apply the CNN based intrusion detection algorithm on LC1, we programmed every layer from the network with the elementary C++ functions.

Pseudocode 1: Convolutional layer implementation (the notation is explained in Table 2)

**for** $i$=1 **to** $f_c$ **do** // for all filters
  **for** $j$=1 **to** len($\mathbf{O}_{c-1}$[1]) **do** // for all input vectors
    **w**={$\mathbf{W}_c[m_c][j][i]$:$\mathbf{W}_c$[1][$j$][$i$]} // current filter
    // convolution start
    **for** $v$=1 **to** $z_c$ **do**
      **c**[$v$-1]=**w**[$m_c$]*$\mathbf{x}_c[j][v-m_c]$+…+**w**[1]*$\mathbf{x}_c[j][v]$
    **end for**
    **c**=**c**[$m_c$:end] // exclusion of the first $m_c - 1$ values
    // convolution end
    // putting convolution of input vector into matrix $\mathbf{I}_c$
    **for** $k$=1 **to** len($\mathbf{O}_{c-1}$[2]) **do**
      $\mathbf{I}_c[j][k]$=**c**[$k$]
    **end for**
  **end for**
  // column-wise summing of the inputs filtered by **w**
  **for** $q$=1 **to** len($\mathbf{O}_{c-1}$[2]) **do**
    **for** $h$=1 **to** len($\mathbf{O}_{c-1}$[1]) **do**
      **p**[$q$]+=$\mathbf{I}_c[h][q]$
    **end for**
    **p**[$q$]+=$b_c[i]$ // addition of the bias value
    **p**[$q$]=**p**[$q$]*(**p**[$q$]>0) // ReLU
    $\mathbf{O}_c[i][q]$=**p**[$q$]
  **end for** // one output vector is obtained
**end for**

| Label | Description | Label | Description |
|---|---|---|---|
| $\mathbf{W}_c$ | weight matrix for layer $c$ | $m_c$ | filter size in layer $c$ |
| $\mathbf{b}_c$ | bias vector for layer $c$ | **w** | current filter |
| $\mathbf{O}_c$ | output matrix from conv. layer $c$ | **c** | convolution vector |
| $f_c$ | no. of filters in layer $c$ | $\mathbf{x}_c$ | input vector |
| $z_c$ | input length in layer $c$ | **p** | current sum |

Table 2. Notation used in Pseudocode 1

The performances of the implemented IDS were validated using a number of attacks. Namely, after a time period in which the system operated in normal conditions, the real data on the LC2 were replaced with false data and transmitted to the LC1. The implemented algorithm successfully detected all attacks, without false positives. In this paper, we present the results of the detection of two attacks ($A_1$ and $A_2$), as shown in Fig. 4. In $A_1$ the signal value is sequentially fixed to 0.5-0.7-0.5-0.7 for certain time periods. On the other hand, $A_2$ presents a sequence of linear increase of signal values by 0.0025 followed by its decrease by 0.002, and another increase by 0.001; the signal is also contaminated with random noise in the range [0, 0.001] per sample, respectively. The moments of attack

detection are marked green dashed line in Fig. 4.

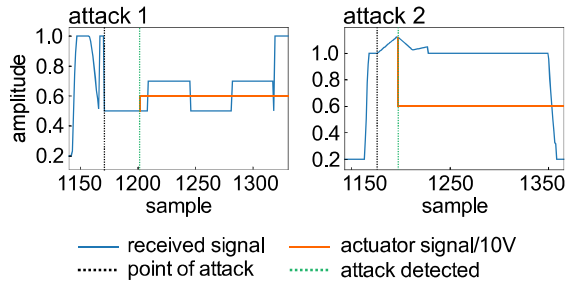


Fig. 4. The detected attacks on DisEPP

Certain actions should be taken to protect the system at the moment of attack detection. In our experiments, regardless the data received from LC2, LC1 sends a value of 0.6 to the actuator (orange line in Fig. 4), stopping the piston immediately. The application of the IDS algorithm did not cause any disturbances during normal system's functioning.

## 5. CONCLUSION

In this paper we have presented the implementation of 1D CNN based algorithm for detection of attacks on smart devices within ICS. In particular, we applied IDS on the low level controller of CPS in the electro-pneumatic positioning system. The procedure for transformation of CNN from the TensorFlow platform to the local controller environment is presented. Namely, built-in functions in TensorFlow have been replaced by elementary functions supported by C++.

The real-time detection performances of the implemented IDS were illustrated using examples of two attacks. The algorithm proved effective in detecting both attacks without false positives. The implementation of the attack detection mechanism did not generate any disturbances on the normal system operation. Our future research efforts will be directed to applying new methods for attack detection based on different deep learning techniques on DisEPP. Furthermore, a CNN-based algorithm will be implemented on more complex systems.

## 6. REFERENCES


[1] Monostori, L., Kádár, B., Bauernhansl, T., Kondoh, S., Kumara, S., Reinhart, G., Sauer, O., Schuh, G., Sihn, W., Ueda, K.: *Cyber-physical systems in manufacturing*, CIRP Annals, vol. 65, no. 2, pp. 621-641, 2016.

[2] Atzori, L., Iera, A., Morabito, G.: *The Internet of Things: A survey*, Computer Networks, vol. 54, no. 15, pp. 2787-2805, 2010.

[3] Khan, I. A., Pi, D., Khan, Z. U., Hussain, Y., Nawaz, A.: *HML-IDS: A hybrid-multilevel anomaly prediction approach for intrusion detection in SCADA systems*, IEEE Access, vol. 7, pp. 89507-89521, 2019.

[4] Huang, X., Dong, J.: *Reliable control of cyber-physical systems under sensor and actuator attacks: An identifier-critic based integral sliding-mode control approach*, Neurocomputing, vol. 361, pp. 229-242, 2019.

[5] Nedeljković, D., Jakovljević, Ž., Miljković, Z.: *The detection of sensor signal attacks in industrial control systems*, FME Transactions, vol. 48, no. 1, pp. 7-12, 2020.

[6] Inoue, J., Yamagata, Y., Chen, Y., Poskitt, C. M., Sun, J.: *Anomaly detection for a water treatment system using unsupervised machine learning*, 2017 IEEE International Conference on Data Mining Workshops (ICDMW), pp. 1058-1065, IEEE, New Orleans, 2017.

[7] MR, G. R., Somu, N., Mathur, A. P.: *A multilayer perceptron model for anomaly detection in water treatment plants*, International Journal of Critical Infrastructure Protection, vol. 31, art. no.100393, 2020.

[8] Mercaldo, F., Martinelli, F., Santone, A.: *Real-Time SCADA attack detection by means of formal methods*, 28th intern. conf. on enab. techn.: infrastr. for collab. enterp. (WETICE), pp. 231-236, IEEE, Napoli, 2019.

[9] Kravchik, M., Shabtai, A.: *Detecting cyber attacks in industrial control systems using convolutional neural networks*, 2018 Workshop on Cyber-Physical Systems Security and PrivaCy, pp. 72-83, ACM, New York, 2018.

[10] NXP Semiconductors N.V., 2009. LPC1769/68/66/65/64/63 32-bit ARM Cortex-M3 microcontroller.URL:https://www.nxp.com/docs/en/data-sheet/LPC1769_68_67_66_65_64_63.pdf

[11] Microchip Technology Inc. (2008, Jan.), "MRF24J40MA 2.4 GHz IEEE Std. 802.15.4TMRF Transceiver Module," [Online].Available:http://ww1.microchip.com/downloads/en/DeviceDoc/70329b.pdf

[12] Nedeljkovic, D. M., Jakovljevic, Z. B., Miljkovic, Z. D., Pajic, M.: *Detection of cyber-attacks in electro-pneumatic positioning system with distributed control*: 27th Telecommunications Forum (TELFOR), art. no. 8971062, IEEE, Belgrade, 2019.

[13] https://www.keil.com/download/, Accessed on: July, 2021.



***Authors:*** **Teach. assist. Dušan Nedeljković, Full Dr. Živana Jakovljević,** University of Belgrade, Faculty of Mechanical Engineering, Department of Production Engineering, Kraljice Marije 16, 11120 Belgrade, Serbia, Phone.: +381 11 3302-200, Fax: +381 11 3370364.
E-mail: dnedeljkovic@mas.bg.ac.rs; zjakovljevic@mas.bg.ac.rs



**ACKNOWLEDGMENTS:** This research was supported by the Science Fund of the Republic of Serbia, grant No. 6523109, AI-MISSION 4.0.
The research in this paper was supported by the Ministry of Education, Science and Technological Development of the Serbian Government, 451-03-68/2020-14/200105.