USING NEURAL NETWORKS WITH DATA QUANTIZATION FOR TIME SERIES ANALYSIS IN LHC SUPERCONDUCTING MAGNETS

MACIEJ WIELGOSZ [a,*], ANDRZEJ SKOCZEŃ [b]

[a]Faculty of Computer Science, Electronics and Telecommunications
AGH University of Science and Technology, al. Mickiewicza 30, 30-059 Cracow, Poland
e-mail: wielgosz@agh.edu.pl

[b]Faculty of Physics and Applied Computer Science
AGH University of Science and Technology, al. Mickiewicza 30, 30-059 Cracow, Poland
e-mail: skoczen@agh.edu.pl

The aim of this paper is to present a model based on the recurrent neural network (RNN) architecture, the long short-term memory (LSTM) in particular, for modeling the work parameters of Large Hadron Collider (LHC) superconducting magnets. High-resolution data available in the post mortem database were used to train a set of models and compare their performance for various hyper-parameters such as input data quantization and the number of cells. A novel approach to signal level quantization allowed reducing the size of the model, simplifying the tuning of the magnet monitoring system and making the process scalable. The paper shows that an RNN such as the LSTM or a gated recurrent unit (GRU) can be used for modeling high-resolution signals with the accuracy of over 0.95 and a small number of parameters, ranging from 800 to 1200. This makes the solution suitable for hardware implementation, which is essential in the case of monitoring the performance critical and high-speed signal of LHC superconducting magnets.

Keywords: Large Hadron Collider, LSTM architecture, signal modelling.

1. Introduction

The Large Hadron Collider (LHC) is a circular proton-proton collider located at the European Organization for Nuclear Research (CERN) on the border between Switzerland and France. It is the largest experimental instrument ever built (Brüning and Collier, 2007). The purpose of this enormous project is the verification of theories developed in elementary particle physics. The experiments conducted at the LHC generate a tremendous amount of scientific data which are later used in the analysis and validation of the physics models regarding the history of the universe and the nature of the matter. The investigated events are so rare that the collisions must take place every $25\,\mathrm{ns}$ to be able to capture the interesting phenomena.

On the other hand, the LHC itself is unique equipment which required the development of many innovations. As such, it is a subject of many research endeavors in the fields of engineering, technology and accelerator physics. It comprises many subsystems built with a multitude of devices installed inside the underground tunnel. The LHC tunnel is almost $27\,\mathrm{km}$ long and located $100\,\mathrm{m}$ underground. The radiation level in the tunnel excludes direct intervention during operation, making remote monitoring and control a necessity.

The main engineering effort is to maximize the availability of the machine while the high safety level is guaranteed. In consequence, there is a significant number of data streams generated by sensors and devices depicting various subsystem conditions. Gathered data triggers an interruption of the accelerator's operation immediately when a dangerous anomaly is detected. Then an analysis of the event must be performed off-line by experts using post mortem (PM) data. The automation and improvement of on-line analysis would be very beneficiary.

The research presented in this publication is especially crucial for the future upgrade of the LHC

*Corresponding author

(Apollinari *et al.*, 2017) and for the study of the next-generation circular accelerator named the Future Circular Collider (FCC) (FCC Study, 2019). The authors focus on a data stream coming from superconducting magnets operated at the LHC and introduce an architecture based on RNNs to model the magnets' work characteristics. The authors' long-term goal is to automate the task of determining the parameters of safe superconducting magnets' operation or at least to reduce the necessary expert involvement. It should be noted that specialists cannot be removed from the process of model creation. However, their work can be made easier by using the proposed system.

The initial version of this paper was presented at the 3rd Conference on *Information Technology, Systems Research and Computational Physics*, Cracow, Poland, 2018.

The publication structure is as follows. Section 2 gives a short overview of the LHC safety system and the data used for building the RNN-based model; Section 3 provides background information about time series analysis and RNNs. The proposed method is described in Section 4. Section 5 provides the results of the experiments. Finally, the conclusions of the research are presented in Section 6.

2. Protection of the Large Hadron Collider

The LHC accelerates two proton beams travelling in opposite directions (Evans and Bryant, 2008). The particles circle the $26\,658.883\,\mathrm{m}$ long beam pipe $11\,245$ times per second. Particle trajectories are formed by superconducting magnets working at a temperature of superfluid helium at about $1.9°\mathrm{K}$. Each of the eight sectors of the LHC comprises about 154 magnets. The magnets produce a magnetic field appropriate to bend the proton trajectory when they conduct an electrical current at the level of $13\,\mathrm{kA}$. This means that the energy stored in one sector is about $1.2\,\mathrm{GJ}$, which is sufficient to heat up and melt $1900\,\mathrm{kg}$ of copper. At a collision state, a separate particle has the energy on the level of $7\,\mathrm{TeV}$. This means that the beam of protons accumulates energy of $360\,\mathrm{MJ}$, which is equivalent to the energy for warming up and melting $515\,\mathrm{kg}$ of copper.

An energy corresponding to a fraction of $\approx 10^{-7}$ of the beam energy can quench a magnet when operated at full current. The quench is a phenomenon of leaving the superconducting state by a coil or a bus carrying strong electric current. A tremendous amount of heat is released, leading to a catastrophic accident.

A quench is a random event. The critical safety levels are, therefore, required to operate the LHC. As a result, the subsystem to protect against a consequence of this kind of event was built at the LHC, and it is permanently maintained and developed.
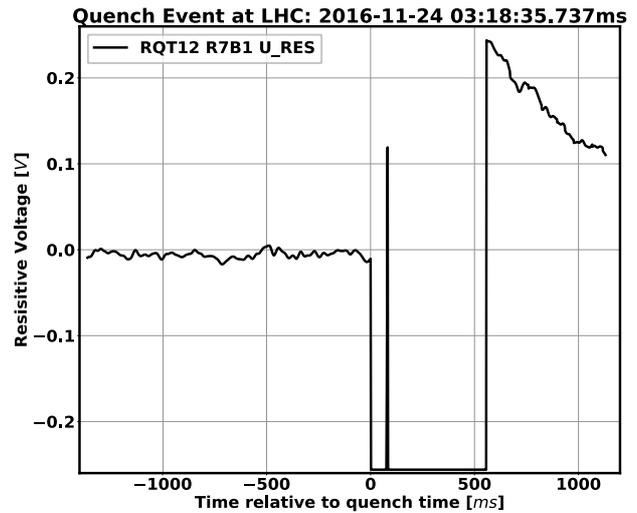

Fig. 1. Example of the PM data for one of 600 A magnets: RQT12.R7 for the coil around bean pipe B1. The curve represents time evolution of resistive voltage $U_{\mathrm{RES}}(t)$ around QUENCHTIME $= 24.1.2016\,3:18:25\,737\,\mathrm{ms}$. The zero on the timescale corresponds to QUENCHTIME. The voltage range of the ADC is from $250\,\mathrm{mV}$ to $-250\,\mathrm{mV}$. The sampling period is $2\,\mathrm{ms}$.

This system dedicated to ensure the LHC safety requirements is known as the machine protection system (MPS) (Wenninger, 2016; Bordry *et al.*, 2001; Schmidt, 2016). In general, it consists of two interlock systems: the power interlock system (PIS) and the beam interlock system (BIS).

The BIS is a superordinate system which collects signals from many sources. There are currently 189 inputs from client systems, ranging from the beam loss monitor (BLM) or the fast magnet current change monitor (FMCM) to the personnel access system. However, the most important and the most complex protection subsystem is the PIS, which ensures communication between systems involved in the powering of the LHC superconducting magnets. This includes power converters (PCs), the quench protection system (QPS), uninterruptible power supplies (UPSs), the emergency stop of electrical supplies (AUG) and the cryogenic system. When the quench protection system (QPS) detects a magnet quench, the power converter is turned off immediately. In total, there is an order of thousands of interlock signals.

When a failure is detected, the beams are dumped, the arrival of new particles is blocked, and a trigger for data acquisition is generated. It is a request related to many LHC systems for providing data that were recorded locally before the failure is detected. These data facilitate the understanding of the reasons for the failure. Each device inside these systems comprises
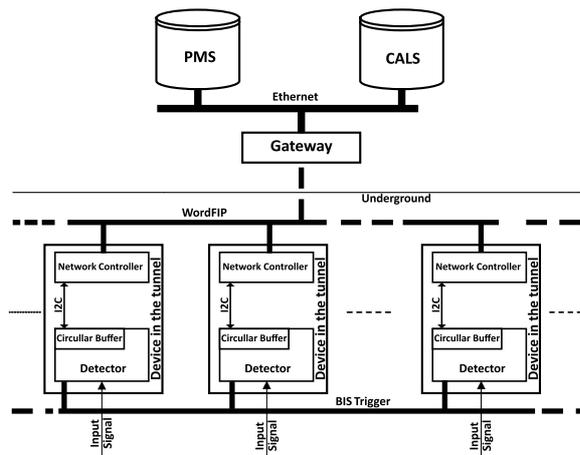
Fig. 2. Hardware path of signals from individual devices in the tunnel to the post mortem system. The thick line in the center marks an industrial data bus used for real-time distributed control WorldFIP.

a circular buffer which, at any time, serves current information about the protected component. In the particular case of a quench detector, the buffer contains a voltage time series acquired with a high resolution by an analogue digital converter (ADC) connected to a superconducting coil. At a trigger time, half of the buffer space is already filled with samples acquired before the event (quench) time. After the event time, the voltage samples are still recorded to fill the rest of the buffer space. Therefore, the buffer contains time series around the trigger time at both sides. An example of buffer contents is shown in Fig. 1. This kind of data is named *post-mortem* because it is recorded after the component ceased its regular activity.

The contents of the buffer are sent out by a network controller of the device over the field-bus to a gateway. Then the data is transferred to a database over an Ethernet network. The data transmission path is shown in Fig. 2. There are two storage systems for data: the post mortem system (PM system) and the CERN accelerator logging service (CALS). The former is used during failures and requested checks, and is a source of the data employed in the experiments presented in this paper. The latter is used to store the acquired monitoring data from any device permanently. Due to low resolution, this data is not used in this study.

The PM system is a diagnostic tool with the role of organizing the collection and analysis of transient data recorded during the time interval around a failure or a request sent by any device in the machine protection system (MPS) (Ciapala *et al.*, 2002). The primary purpose is to provide a fast and reliable tool for equipment experts and operation crews to help them decide accelerator operation can continue

safely or whether an intervention is required. When a failure (a beam loss or a magnet quench) happens, the individual devices' buffers are frozen and transmitted to the PM system for further storage and analysis (Ciapala *et al.*, 2002; Lauckner, 2001; Borland, 1998).

The architecture of the PM system is scalable, very flexible and dependable. A dynamic load of the system is balanced during data collection. Any device can dump the PM data transparently and without any additional configuration effort. The data storage is highly redundant and equipped with data consistency check. The data are stored in a file with a self-describing format known as the JavaScript object notation (JSON). Therefore, files can be processed later in any program.

Users can access the PM data utilizing a specially designed representational state transfer (REST) application programming interface (API). The aim is to serve multiple language technologies according to user preferences: Python, MATLAB, LabVIEW, C++, and Java. A user is not dependent on the data format or the file system. Direct extraction of only one signal from a big dataset is possible without the necessity of reading the entire set. The API can handle very complex queries.

2.1. Post-mortem signals. Among many kinds of data files stored within the PM system, this study concerns the data acquired during events happening within superconducting coils of one class of LHC magnets. The common feature of this class is the level of the electrical current flowing through their coils during normal operation. This level is $600\,\mathrm{A}$ and therefore the commonly used name of this class is "600 A magnets".

The PM javascript object notation (JSON) file for "600 A magnets" contains many self-descriptive fields with digital and analogue signals. The set of these signals monitors the complete system of supply and protection of one magnet in the LHC tunnel. Table 1 presents a list of signals which were the most important for this study.

The signals are measured using dedicated equipment named the quench detector (Skoczeń and Skała, 2009; Steckert and Skoczeń, 2017). All signals presented in Table 1 are converted to digital samples using 8-bit ADCs. The current I_{DCCT} flowing through the coil is measured with Hall sensor called a direct current current transducer (DCCT). Due to the superconducting state of the coil, the total voltage U_{DIFF} consists mainly of an inductive component (when current changes) and residual resistive voltage. When the current I_{DCCT} changes, the voltage U_{DIFF} grows. The quench detector has to distinguish between two reasons for increasing the voltage U_{DIFF}. In the first situation, only the inductive component of the voltage is huge while the resistive part remains zero. This state is normal during ramping up or down the current in the magnet. In the second situation, both the components are comparable,

Table 1. List of the most important time series stored in the PM JSON file of "600 A magnets."

Signal name	Symbol	Unit	Description
Coil current	I_{DCCT}	A	Electrical current flowing through superconducting coil
Total differential voltage	U_{DIFF}	V	Voltage measured between terminals of superconducting coil
Resistive voltage	U_{RES}	V	Resistive part of U_{DIFF} extracted by quench detector
Time derivative of current	I_{DIDT}	$A\,s^{-1}$	Derivative of current over time calculated by quench detector
Event time stamp	QUENCHTIME	ns	Unix time stamp (UTS) with augmented precision for a moment when U_{RES} exceeds threshold

or the resistive part is even higher. This situation is hazardous, and it signifies a loss of the superconducting state of the coil. The quench detector compensates the inductive part and compares the resistive part with a predefined threshold. The quench detector solves Kirchhoff's equations of the supplying circuit in order to extract the resistive voltage U_{RES}. During this process, it is necessary to calculate the time derivative of the current flowing through the coil I_{DIDT}. If the U_{RES} remains higher than a threshold for a time longer than a predefined value, the trigger signal is generated. This trigger suspends the operation of the whole accelerator. The time stamp of this event is stored as a QUENCHTIME field.

3. Time series analysis

The spatial and temporal components may characterize virtually all real-world phenomena. The spatial ones exist in space, and it is assumed that they are stationary, i.e., do not evolve in time, whereas the temporal ones unfold in time and have no spatial component. This distinction is, of course, an idealization since there are neither pure spatial nor temporal phenomena; most of them may be described as a mixture of those two different components.

3.1. Heuristic smoothing methods.
The basic smoothing methods are focused on the temporal component, and the spatial resolution of the input signal is limited to a single series. Building the best-suited heuristic smoothing model involves determining the type of model (whether the model must include the trend, seasonality or both) and then obtaining the best-fit parameters for the selected model.

3.1.1. Moving average.
Moving average is a simple method which uses several of the past samples to predict a single future value. The prediction is based on computing an average of values in the scope of a window. The quality of the prediction depends on the window size k, which is the only and main hyper-parameter,

$$\hat{y}^{(t+1)} = \frac{1}{k}\sum_{n=0}^{k-1} y^{(t-n)}, \tag{1}$$

where $y^{(t)}$ is the true observation, and $\hat{y}^{(t)}$ is the prediction for the same sample.

3.1.2. Exponential smoothing.
The exponential smoothing methods are a wide class of models with different assumptions and degrees of complexity that are derived from the common idea of creating forecasts. The common denominator of exponential smoothing is the attribution (exponentially) of decreasing weights to historical observations in determining the prognosis of a future observation.

Exponential smoothing models are based on a reasonable assumption that the future value depends not only on the last observed value, but on the whole of its series, and at the same time, the influence of older values is smaller than that of newer values.

Assuming $\hat{y}^{(0)} = y^{(0)}$, we have that

$$\hat{y}^{(t+1)} = \alpha y^{(t)} + (1-\alpha)\hat{y}^{(t)}. \tag{2}$$

Single exponential smoothing models are characterized by α, which controls the impact of past and current values on a predicted value. For $\alpha = 0$, only the current value is taken into account.

3.1.3. Double exponential smoothing.
Double exponential smoothing (Holt's method) models are characterized by two different parameters, α and β. In this approach, the parameter β controls the trend.

Assuming $\hat{y}^{(0)} = y^{(0)}$ and $\hat{y}^{(1)} = y^{(1)}$, we have

$$l^{(t)} = \alpha y^{(t)} + (1-\alpha)(l^{(t-1)} + b^{(t-1)}), \tag{3}$$

$$b^{(t)} = \beta(l^{(t)} - l^{(t-1)}) + (1-\beta)b^{(t-1)}, \tag{4}$$

$$\hat{y}^{(t+1)} = l^{(t)} + b^{(t)}, \tag{5}$$

which can be expanded as

$$\hat{y}^{(t+1)} = \alpha(1+\beta)y^{(t)} - \alpha y^{(t-1)} + (2-\alpha-\alpha\beta)\hat{y}^{(t)} - (1-\alpha)\hat{y}^{(t-1)}. \tag{6}$$

A further extension of Holt's method is triple exponential smoothing (Holt–Winter method) taking into account the data seasonality. The PM data analysis revealed no seasonal component, and therefore we decided not to include it in our experiments.

3.2. Recurrent neural networks.

In deep learning applications there is a well-established practice to use feed-forward neural networks (FNNs) and convolutional neural networks (CNNs) to address tasks dominated by a spacial component (Krizhevsky *et al.*, 2012). On the other hand, data which contain more temporally distributed information are usually processed by models built around recurrent neural networks. Of course, it is possible to treat time series signals as a vector of spatial values and use the FNN or CNN to classify them or do some regression (LeCun, 2015).

The voltage and current time series, which are used to train models described in this paper and make predictions unfold in time, and their temporal component are dominant. Therefore, a decision was made to use RNN networks. There are two fundamental equations, (7) and (8), which characterize computations of a recurrent neural network,

$$h^{(t)} = Q(W_{hx}x^{(t)} + W_{hh}h^{(t-1)} + b_h), \qquad (7)$$

$$\hat{y}^{(t)} = \sigma(W_{yh}h^{(t)} + b_y), \qquad (8)$$

where Q is an activation function, W_{hx}, W_{yh} and W_{hh} are weight matrices of input-hidden layer, hidden-output layer and recurrent connections, respectively, and b_h and b_y are vectors of biases.

One of the most efficient RNNs architectures is the long short-term memory (Graves, 2012; Morton *et al.*, 2016; Pouladi *et al.*, 2015; Chen *et al.*, 2016). Since its invention in 1997, the LSTM has been updated and modified (Greff *et al.*, 2015) to improve its modeling properties and reduce large computational demands of the algorithm. It is worth noting that the LSTM, as opposed to a vanilla RNN, is much more complex regarding the internal component constituting its cell—this complexity results in a long training time of the algorithm. Therefore, there were many experiments conducted with simpler architectures which preserve beneficial properties of the LSTM. One of such architectures is the GRU (Chung *et al.*, 2015), which is widely used in deep learning as an alternative for the LSTM. According to the recent research results, it even surpasses the LSTM in many applications (Chung *et al.*, 2014).

The LSTM internal structure is based on a set of connected cells, with three different gates in each one:

- *input* gate $i_c^{(t)}$ controls input activations into the memory element;

- *output* gate $o_c^{(t)}$ controls the cell outflow of activations into the rest of the network;

- *forget* gate $f_c^{(t)}$ scales the internal state of the cell before summing it with the input through the self-recurrent connection of the cell. This enables gradual forgetting in the cell memory.

In addition, the LSTM cell also comprises an input node $g_c^{(t)}$ and an internal state node $s_c^{(t)}$. The output of a set of LSTM cells is calculated according to the following set of vector equations:

$$g^{(t)} = \phi(W_{gx}x^{(t)} + W_{gh}h^{(t-1)} + b_g), \qquad (9)$$

$$i^{(t)} = \sigma(W_{ix}x^{(t)} + W_{ih}h^{(t-1)} + b_i), \qquad (10)$$

$$f^{(t)} = \sigma(W_{fx}x^{(t)} + W_{fh}h^{(t-1)} + b_f), \qquad (11)$$

$$o^{(t)} = \sigma(W_{ox}x^{(t)} + W_{oh}h^{(t-1)} + b_o) \qquad (12)$$

$$s^{(t)} = g^{(t)} \odot i^{(t)} + s^{(t-1)} \odot f^{(t)}, \qquad (13)$$

$$h^{(t)} = \phi(s^{(t)}) \odot o^{(t)}, \qquad (14)$$

where ϕ is a hyperbolic tangent. The hard sigmoid function controlling the gating mechanism is defined as follows:

$$\sigma(x) = \begin{cases} 0 & \text{if} \quad x \leq t_l, \\ ax + b & \text{if} \quad x \in (t_l, t_h), \\ 1 & \text{if} \quad x \geq t_h. \end{cases} \qquad (15)$$

The field of RNN-based methods for anomaly detection is growing very fast with the progress and discoveries in deep learning (DL). The basic concept of those methods uses original signal modeling.

The architecture of an LSTM-based anomaly detector which incorporates both the hierarchical approach and multi-step analysis was proposed in Malhotra *et al.* (2015), capitalizing on the property of generalization which results from stacking of several RNN layers. The Gaussian distribution of an error signal (the difference between predicted and real values) was used to decide if the prediction error signifies an anomaly.

There is also a whole branch of detectors which exploit the property of inconsistent signal reconstruction in the presence of anomalies (Marchi *et al.*, 2015a; 2015b). The authors trained the model of the autoencoder on regular data and set a threshold above which the reconstruction error is considered an anomaly. The papers deal with acoustic signals, but such an approach may be efficiently employed in other domain such as videos (Chong and Tay, 2017).

4. Proposed method

In the work of Wielgosz *et al.* (2017), experiments with the data collected from the CALS database were conducted using the setup presented in Fig. 3(a), which employed the RMSE measure. A great challenge in this approach is the lack of a strict reference threshold when detecting anomalies. In order to determine the error level, a group of experts must be consulted, and it is not always easy to set one. This difficulty exists because the RMSE
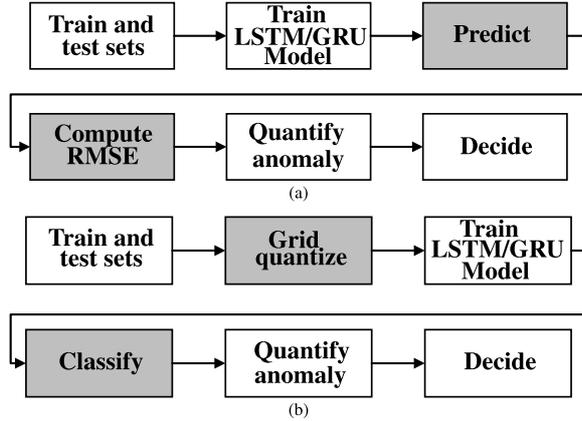
Fig. 3. Envisioned anomaly detection setups featuring the RMSE and prediction (a) as well as grid quantization and classification (b).

does not always indicate anomalous behavior well enough to quantify it correctly (Strecht *et al.*, 2015).

It is important to emphasize that this work does not address anomaly detection as such. The primary objective of this research is the examination of the proposed method based on a quantized grid as well as the analysis of its limitations.

Based on the experience from the previous work (Wielgosz *et al.*, 2017), a new experimental setup was designed. This new approach is shown in Fig. 3(b). It allows converting a regression to the classification. The main difference between the previously used approach and the proposed one is the introduction of a grid quantization and classification steps (see marked boxes in Fig. 3). Consequently, in the new approach, the train and test data are brought to several categories depending on the grid size. This transformation may be perceived as a specific kind of quantization since the floating-point data are converted to the integer representation denoted as categories in this particular setup.

After initial experiments, it was determined that utilizing the static grid does not yield the expected results due to a massive cardinality imbalance between classes. For example, when using a grid equal to 1024, the cardinality of the most numerous class was over 500 times higher than that of the second one. As a result, even with the accuracy near 100%, all samples were classified as belonging to a single (or at the very best two) class.

Analyzing these results, we decided to use a form of adaptive quantization to achieve more balanced classes, resulting in better resource utilization and more viable results. If n is the number of samples, $m \in \mathbb{N}$ is the number of classes (categories, bins) and $srt\_samples_i$ is the i-th sample in the ascending sorted array of all available normalized signal samples, the basic adaptive quantization can be described as the following mapping

from input space S_{norm} to quantized space S_{qa}:

$$S_{norm} \xrightarrow{\Pi_{qa}(m)} S_{qa} : \{0 \ldots m-1\}^{1 \times n}, \quad (16)$$

where $\Pi_{qa}(m)$:

$$\bigwedge_{x \in S_{norm}} \bigvee_{y \in S_{qa}} y = \begin{cases} edges_y \leq x \cdot m < edges_{y+1} \\ \qquad \qquad \text{if } x < 1, \\ y = m-1 \quad \text{if } x = 1, \end{cases} \quad (17)$$

$$edges : \bigwedge_{0 \leq k \leq m} edges_k = \begin{cases} 0 \quad \text{if } k = 0, \\ srt\_samples_{k \cdot \lceil \frac{n}{m} \rceil} \\ \qquad \text{if } 0 < k < m, \\ 1 \quad \text{if } k = m. \end{cases} \quad (18)$$

The subscript $k \cdot \lceil n/m \rceil$ is the index of the first sample in the k-th bin. For example, if

$$x = [0, .3, .9, .1, 0, .5, 0, .3, .1, .2, .3]$$

and the number of bins $m = 4$, then

$$srt\_samples = [0, 0, 0, .1, .1, .2, .3, .3, .3, .5, .9]$$

and $edges$ would be

$$edges_0 = 0,$$
$$edges_1 = srt\_samples_{1 \cdot \lceil \frac{11}{4} \rceil} = srt\_samples_3 = .1,$$
$$edges_2 = srt\_samples_{2 \cdot \lceil \frac{11}{4} \rceil} = srt\_samples_6 = .3,$$
$$edges_3 = srt\_samples_{3 \cdot \lceil \frac{11}{4} \rceil} = srt\_samples_9 = .5,$$
$$edges_4 = 1,$$

yielding the input vector after quantization:

$$y = [0, 2, 3, 1, 0, 3, 0, 2, 1, 1, 2].$$

In the actual experiments, a recursive version of the adaptive algorithm was used, ensuring there are no duplicated edges. The implementation is available online.[1] Example quantized series can be seen in Fig. 4, with especially illustrative panel (b), which shows the distribution of the grid bins across selected (marked with rectangle in panel (a)) part of the signal. It may be noticed that, out of five bins, bins 2 and 3 are the narrowest because a large number of the signal transitions occurring within the range covers by the bins. The bins' density is proportional to the level of signal variation, i.e., areas with greater variability are more densely covered with the grid.

The model is trained using historical data fed in portions. The next sample category is predicted based
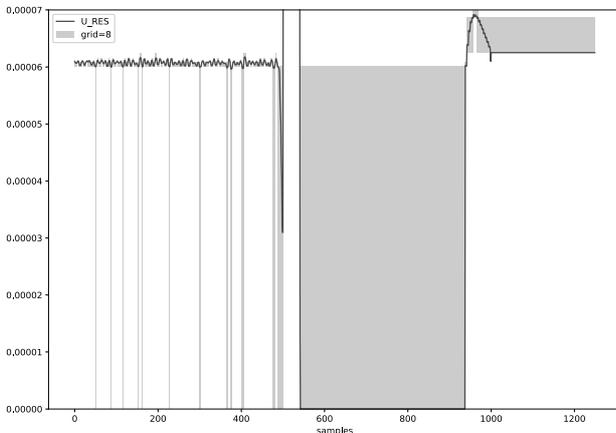
---

[1]https://bitbucket.org/maciekwielgosz/anomaly_detection.
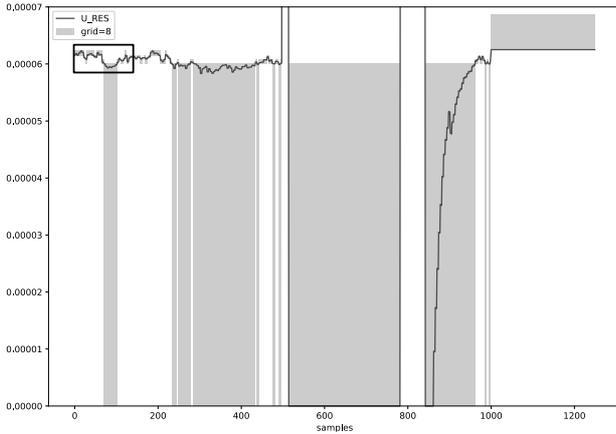
(a)
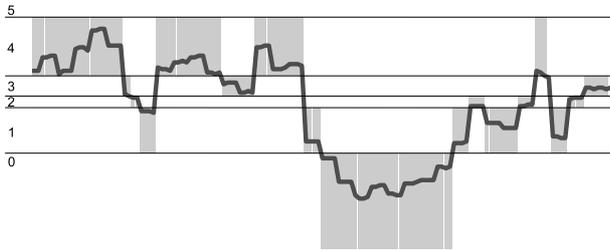
(b) zoomed-in view of the marked part of (a)

(c)

Fig. 4. Visualization of the recursive adaptive grid size $= 8$ for three example series. The line depicts the original signal, gray spans—grid bin indication for the corresponding original sample. Six of the calculated grid bins (see (b)) have edges with a value below 7×10^{-5}, so the y-axis is cropped accordingly to produce the visualization on (a) and (c).

on the number of previous steps equal to the look_back parameter. It is worth noticing that the increase in the grid size leads to an increase in the resolution and at the same time is more challenging for the classifier. Since the reliability of any further application (e.g., anomaly

detection) depends on the accuracy of the model, the experiments on hyper-parameters' influence on model performance for the PM data were conducted.

5. Experiments and a discussion

The main goal of the conducted experiments was the validation of the feasibility of the application of the proposed method for modeling the voltage time series of LHC superconducting magnets. A long short-term memory model was built with the Keras/Tensorflow libraries (Chollet, 2015). The number of model parameters increases slightly for bigger grids. The exact values can be seen in Table 2.

5.1. Dataset. All the data used for the experiments were collected from the CERN PM database using the PM JSON API written in Python. The procedure of data extraction from the PM database is composed of several steps, as presented in Fig. 5. A dedicated application and a set of parameters such as the signal name and number of time steps were used.

The signals concerning "600 A magnets" were collected for different time series: U_{RES}, U_{DIFF}, I_{DIDT} and I_{DCCT}. The modeling experiments were conducted using the U_{RES} signal only. Basic information about the data can be found in Table 3.

5.2. Quality assessment measures. Accuracy is used to compare the model performance for various grid sizes. It can be defined as

$$\text{accuracy} = \frac{1}{N} \sum_{i=0}^{N-1} 1\left(Y_i = \hat{Y}_i\right), \qquad (19)$$

where Y_i and $\hat{Y}_i$ are the i-th sample true quantized value and the one predicted by the model, respectively, and N is the dataset cardinality. Additionally, for baseline methods (see Section 3.1), the mean absolute error (MAE) and the root mean squared error (RMSE) were calculated:

$$\text{MAE} = \frac{1}{N} \sum_{i=0}^{N-1} |y_i - \hat{y}_i|, \qquad (20)$$

Table 2. Number of models' parameters depending on the selected architecture and grid.

grid	number of cells in LSTM layer(s)			
	16	32	64	64, 32
8	1288	4616	17 416	29 576
16	1424	4880	17 936	29 840
32	1696	5408	18 976	30 368
64	2240	6464	21 056	31 424

Table 3. Basic information about data used in the experiments. If a series was too short to be used in a particular setup (due to the required look_back), it was skipped.

dataset	no. of series	no. of series with QUENCHTIME field	total samples	total anomalous samples
training	31 496	13 613 (43.22 %)	39 509 987	10 272 823 (26.00 %)
validation	7874	3449 (43.80 %)	9 873 974	2 605 321 (26.39 %)
testing	9843	4293 (43.61 %)	12 349 311	3 241 540 (26.25 %)

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=0}^{N-1} (y_i - \hat{y}_i)^2}, \qquad (21)$$

where y_i and $\hat{y}_i$ are the i-th sample value and the one predicted by the model, respectively.

5.3. Results. The conducted experiments showed that even relatively small LSTM models are capable of modelling the U_{RES} signal (Table 4). The accuracy improves with increasing model complexity and longer look_back and decreases when a bigger grid is used. However, due to the most prevalent series length being 1250 samples (96.09 % of series were of this length), we decided not to use look_back longer than 256.

For comparison, the classical forecasting methods were employed, whose results are shown in Table 5. The best performance was achieved by utilizing exponential smoothing (2), resulting in an accuracy $\approx 2\%$ lower than the best of the tested LSTM models. In order to compare the RNN-based model using the proposed quantized regression scheme with baseline methods, their output (whose quality is usually determined based on the MAE/RMSE) was quantized and the accuracy was calculated (Table 5). Exponential smoothing with a grid of length 8 performs best among the analyzed baseline methods. However, it is still inferior to the best of the analyzed RNN-based models. Furthermore, the presented studies consider only a single signal; for more

Table 4. Model accuracy as a function of grid, look_back and the model architecture. All models were trained for six epochs with the batch size = 32 768.

cells	look_back				
	1	4	32	128	256
grid=8					
16	0.9394	0.9403	0.9448	0.9494	0.9558
32	0.9394	0.9428	0.9462	0.9502	0.9544
64	0.9394	0.9442	0.9518	0.9527	0.9554
64, 32	0.9394	0.9455	0.9540	0.9531	0.9611
grid=16					
16	0.8940	0.9098	0.9087	0.9203	0.9321
32	0.9034	0.9127	0.9137	0.9242	0.9361
64	0.9034	0.9191	0.9209	0.9265	0.9381
64, 32	0.9119	0.9214	0.9261	0.9285	0.9385
grid=32					
16	0.8533	0.8739	0.8765	0.8844	0.9059
32,	0.8569	0.8804	0.8807	0.8940	0.9131
64,	0.8657	0.8891	0.8921	0.9007	0.9177
64, 32	0.8874	0.8901	0.8957	0.9076	0.9200
grid=64					
16,	0.8236	0.8383	0.7713	0.8544	0.8318
32,	0.8264	0.8448	0.8448	0.8634	0.8836
64,	0.8295	0.8601	0.8587	0.8660	0.8924
64, 32	0.8604	0.8549	0.8656	0.8779	0.8938

complex cases composed of more signals such direct comparison would be challenging, since RNNs account for inter-dependencies between series as opposed to ES.
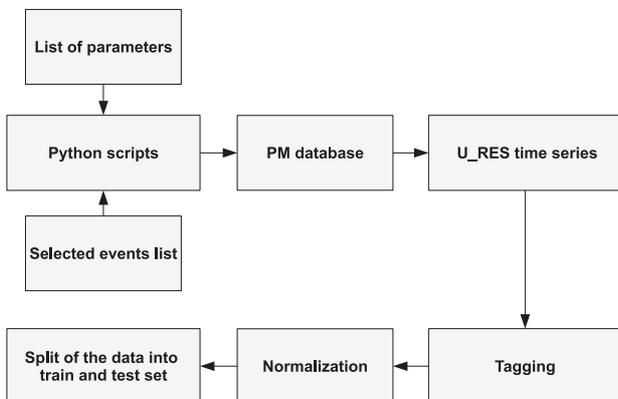
5.4. Anomaly detection with data quantization. For an anomaly detection task, if the model is trained only on non-anomalous data, it is enough to observe the predicted categories for several time steps. When it turns out that the predicted category differs from the actual one over the selected period, this means that an anomaly occurred (Fig. 6). The data expert has a much easier task in this case (when compared with the RMSE-based approach) because the only decisions required are about the grid size and the anomaly detection window, both of which are well quantifiable parameters.

The PM data we used were weakly labeled, with all samples occurring after QUENCHTIME marked as anomalous. Weakly labeling, as opposed to strict

Fig. 5. Procedure for the extraction of voltage time series with selected events from the PM database using U_{RES} as an example.

Table 5. Baseline results. The parameters α and β were minimized using a truncated Newton (TNC) algorithm, with the RMSE as the loss function. The baseline methods are described in Section 3.1.

Method	Parameters	MAE	RMSE	Accuracy			
				grid=8	grid=16	grid=32	grid=64
Moving average	window = 1	1.5253×10^{-17}	3.2137×10^{-16}	0.5727	0.5656	0.5406	0.5193
Exponential smoothing	$\alpha = 0.9476$	2.1296×10^{-5}	4.1668×10^{-3}	0.9433	0.9165	0.8919	0.8704
Double exponential smoothing	$\alpha = 0.9480$, $\beta = 0$	2.1293×10^{-5}	4.1672×10^{-3}	0.8066	0.7779	0.7530	0.7816

labeling, utilizes limited information regarding the modeled phenomenon as provided by experts in the field. In the case of the presented method and experiments, only the very moment of quench occurrence was marked by the experts (i.e., LHC technicians overlooking the data acquisition process). Consequently, it is not always clear where quenches end and if there are any other phenomena which overlap with the quench (e.g., a power converter
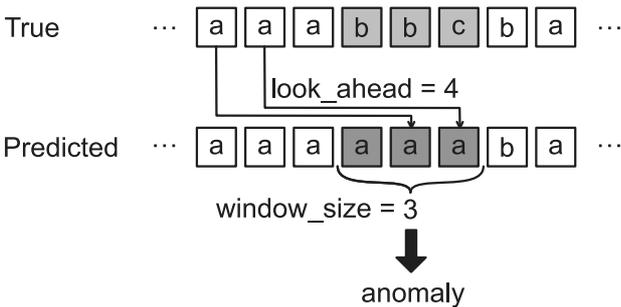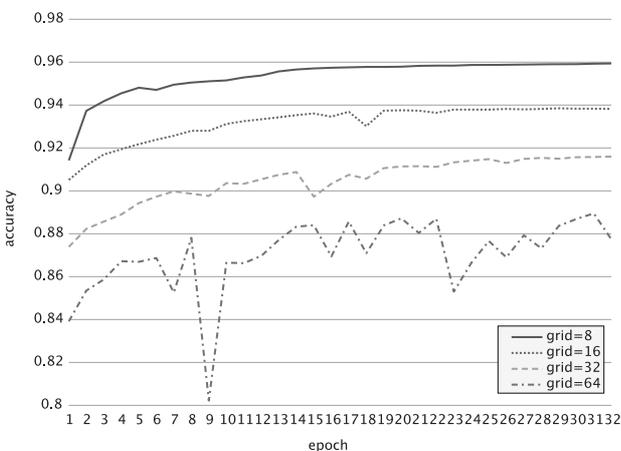

Fig. 6. Data flow diagram for the proposed method.

Fig. 7. Accuracy as a function of the number of epochs for models with cells = (64,32), look_back = 128 and the batch size = 32 768.

or monitoring apparatus distortion). In order to provide stringent labeling, close inter-team cooperation between technicians would be essential, especially concerning data interpretation. Such cooperation is very challenging, but it is gradually being introduced, which allows expecting a strictly labeled dataset to be ready in the future.

The analyzer part of the detector to work correctly requires a minimal grid size (e.g., grid ≥ 8) to allow the analyzer to capture tiny changes within the signal. On the other hand, increasing the grid size compromises the accuracy of the model. Consequently, the choice of the accuracy is a trade-off between the model and analyzer performance, which results in the effectiveness of the detector as such. The increase in the number of training epochs from 6 to 32 results in an accuracy better by $\approx 1\%$ for smaller grid sizes (Fig. 7). It can be concluded, however, that even the biggest of the tested models is too small to handle grid=64.

The early experiments with anomaly detection in the PM data resulted in a high rate of false positives (with the precision ≈ 0.43), indicating the need to enhance further the part of a system responsible for distinguishing between anomaly occurrence and model imperfection. Example visualization of anomaly detection results, using the model trained without data containing the QUENCHTIME field, is presented in Fig. 8.

It is worth noticing that the presented solution contains a module which enables the modeling of time series. In order to use it with more specific applications such as anomaly detection, the module is extended with the analyzer section which operates on top of the model. A more detailed explanation of this architecture was given by Wielgosz *et al.* (2018a) and is out of the scope of this paper.

Figure 8 shows the results of three different experiments. In the top one, the trigger for anomaly detection implemented in the analyzer was activated several times indicating a single false positive case, located just before the areas marked as an actual start of the anomaly (the gray span). It was assumed in the adopted quality measure scheme that multiple triggers within an area marked as an anomaly are counted as one,
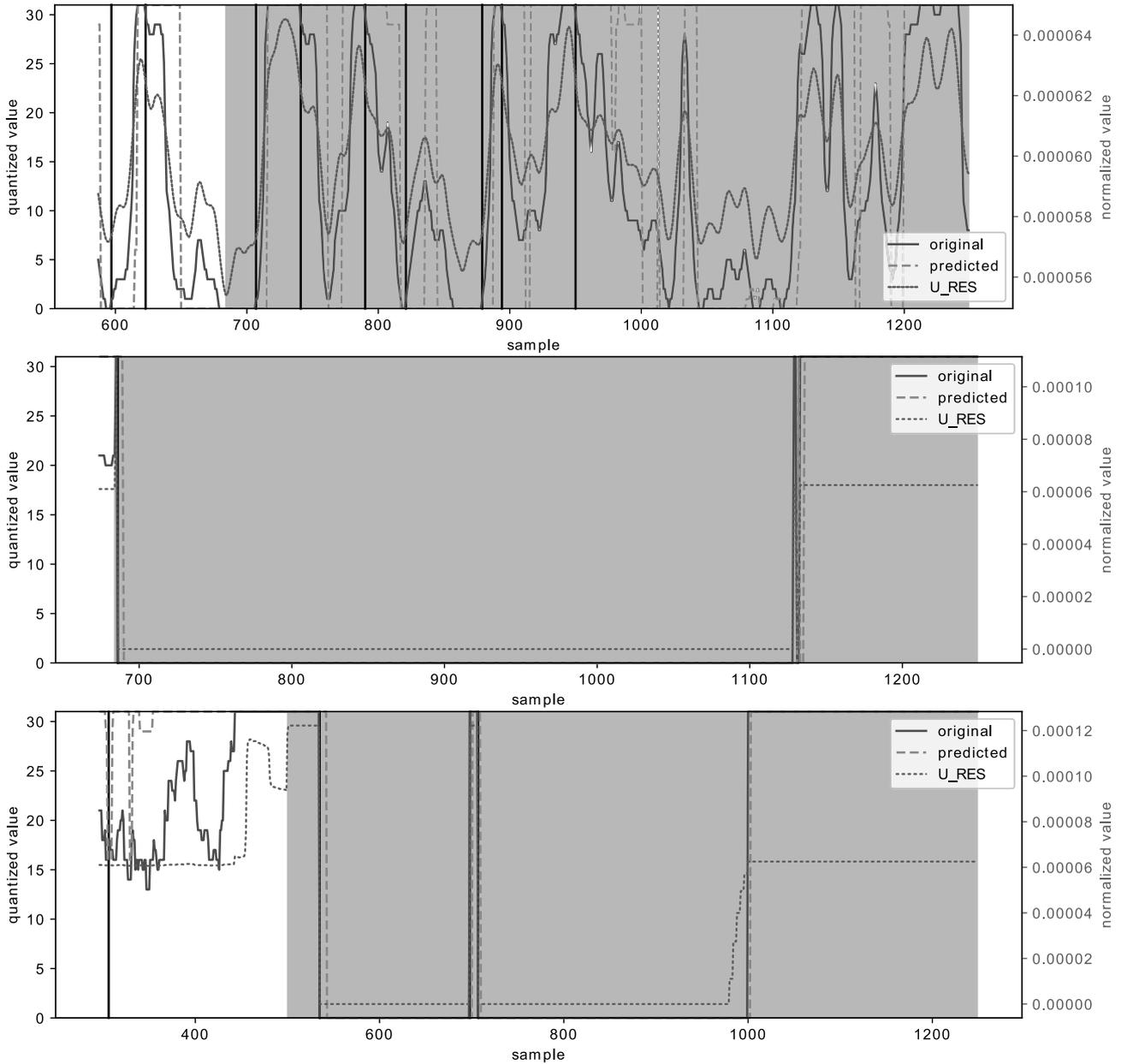
Fig. 8. Visualization of the proposed anomaly detector results for three example series using the LSTM model with two layers (64 and 32 cells), look_back = 256 and grid = 32. The gray span indicates the area labelled in data as an anomaly (from QUENCHTIME until the end of a series); the thick vertical black lines are the point detector marked as a start of anomaly (trigger points); the dotted grey line (and scale on the right) represents the original, normalized signal before quantization, while the solid and dashed lines represent the real and the predicted quantized signal, respectively.

and thus all the black lines (in the top panel of Fig. 8) are considered to be true positives. In the middle panel of the figure, the proposed system worked correctly, and the anomaly was accurately detected. The trigger point is in line with the beginning of the labeled area. In the bottom panel of Fig. 8, the anomaly detection module triggers too early, confusing distortions in the original signal with the start of the real anomaly. Further research is needed to check how weak data labeling affects those results and

whether the system is capable of confirming and detecting the existence of anomaly precursors, which at the moment would be labeled as part of the normal operation.

5.5. Hardware implementation considerations. To comply with the CERN magnets monitoring system requirements, the detector system, including the RNN, will need to be implemented in hardware using field-programmable gate arrays (FPGAs). Networks of

a similar size as proposed for the task were already implemented for application in speech recognition and were described by Chang *et al.* (2015), Han *et al.* (2017), and Lee *et al.* (2016). Differences in the described approaches yield varying results, with the single iteration execution time ranging from $\sim 16\,\mu s$ to $1\,ms$.

Due to hardware implementation requirements, the look_back value should be carefully analyzed. This analysis is especially pronounced when it comes to strong pipeline architecture of the hardware module, where large look_back may affect resource consumption.

In order to implement the hyperbolic tangent $\phi(x)$ and hard sigmoid $\sigma(x)$ functions in hardware, they can be approximated (near zero) using Taylor series (22–23). Such approximation is possible because after normalization the values rarely will be outside the $[-1, 1]$ range,

$$\phi(x) = x - \frac{x^3}{3} + \frac{2x^5}{15}, \qquad (22)$$

$$\sigma(x) = \frac{1}{2} + \frac{x}{4} - \frac{x^3}{48} + \frac{x^5}{480}. \qquad (23)$$

6. Conclusions and future work

This work extends the existing investigations (Wielgosz *et al.*, 2017; 2018a; 2018b; 2018c; 2020) using higher resolution data and more diverse models. The importance of the subject grows because the project High Luminosity LHC (HL-LHC) enters its engineering phase (Apollinari *et al.*, 2017). The goal is to increase the rate of particle collisions (luminosity). What is more, there is a dawn of the next huge CERN project named the Future Circular Collider (FCC) (FCC Study, 2019). The circumference of the ring is predicted to be $80\,km$ or even $100\,km$. Consequently, there is a need for a system capable of monitoring and prediction of malfunctions.

The authors showed that the developed system supersedes the state-of-the-art conventional methods, despite a lack of exhaustive hyper-parameter optimization of the RNN-based model.

In the future, the authors are going to investigate the feasibility of implementing a predictive model on FPGAs. Performing computations on a PC works well for the validation of the idea, but requirements of control systems like the QPS are rather *hard real-time*, which PC systems are incapable of doing.

Acknowledgment

References

Apollinari, G., Béjar, A.I., Brüning, O., Fessia, P., Lamont, M., Rossi, L. and Tavian, L. (2017). *High-Luminosity Large Hadron Collider (HL-LHC): Technical Design Report V. 0.1*, Yellow Reports: Monographs, CERN, Geneva, `http://cds.cern.ch/record/2284929`.

Bordry, F., Denz, R., Mess, K.-H., Puccio, B., Rodriguez-Mateos, F. and Schmidt, R. (2001). Machine protection for the LHC: Architecture of the beam and powering interlock system, *LHC Project Report 521*, CERN, Geneva, `https://cds.cern.ch/record/531820`.

Borland, M. (1998). A brief introduction to the SDDS toolkit, *Technical report*, Argonne National Laboratory, Lemont, IL, `https://ops.aps.anl.gov/SDDSIntroTalk/slides.html`.

Brüning, O. and Collier, P. (2007). Building a behemoth, *Nature* **448**(7151): 285–289.

Chang, A.X.M., Martini, B. and Culurciello, E. (2015). Recurrent neural networks hardware implementation on FPGA, *arXiv* 1511.05552 [cs.NE].

Chen, X., Liu, X., Wang, Y., Gales, M.J.F. and Woodland, P.C. (2016). Efficient training and evaluation of recurrent neural network language models for automatic speech recognition, *IEEE/ACM Transactions on Audio, Speech, and Language Processing* **24**(11): 2146–2157.

Chollet, F. (2015). Keras, GitHub repository, `https://github.com/fchollet/keras`.

Chong, Y.S. and Tay, Y.H. (2017). Abnormal event detection in videos using spatiotemporal autoencoder, *in* F. Cong *et al.* (Eds), *Advances in Neural Networks, ISNN 2017*, Springer International Publishing, Cham, pp. 189–196.

Chung, J., Gülçehre, Ç., Cho, K. and Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling, *arXiv* 1412.3555[cs.NE].

Chung, J., Gülçehre, Ç., Cho, K. and Bengio, Y. (2015). Gated feedback recurrent neural networks, *arXiv* 1502.02367[cs.NE].

Ciapala, E., Rodríguez-Mateos, F., Schmidt, R. and Wenninger, J. (2002). The LHC post-mortem system, *Technical Report LHC-PROJECT-NOTE-303*, CERN, Geneva, `http://cds.cern.ch/record/691828`.

Evans, L. and Bryant, P. (2008). LHC Machine, *Journal of Instrumentation* **3**(08): S08001.

FCC Study (2019). *Future Circular Collide, Conceptual Design Report*, Yellow Reports: Monographs, CERN, Geneva, `https://fcc-cdr.web.cern.ch/`, (in press).

Graves, A. (2012). *Neural Networks*, Springer, Berlin/Heidelberg.

Greff, K., Srivastava, R.K., Koutník, J., Steunebrink, B.R. and Schmidhuber, J. (2015). LSTM: A search space odyssey, *ArXiv* 1503.04069 [cs.NE].

Han, S., Kang, J., Mao, H., Hu, Y., Li, X., Li, Y., Xie, D., Luo, H., Yao, S., Wang, Y., Yang, H. and Dally, W. B.J. (2017).

ESE: Efficient speech recognition engine with sparse LSTM on FPGA, *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (FPGA'17), Monterey, CA, USA*, pp. 75–84.

Krizhevsky, A., Sutskever, I. and Hinton, G.E. (2012). ImageNet classification with deep convolutional neural networks, *in* F. Pereira *et al.* (Eds), *Advances in Neural Information Processing Systems 25*, Curran Associates, Inc., Red Hook, NY, pp. 1097–1105.

Lauckner, R.J. (2001). What data is needed to understand failures during LHC operation, *CERN-SL-2001-003*, CERN, Geneva, pp. 278–283, https://cds.cern.ch /record/567214.

LeCun, Y. (2015). Deep learning of convolutional networks, *2015 IEEE Hot Chips 27 Symposium (HCS), Cupertino, CA, USA*, pp. 1–95.

Lee, M., Hwang, K., Park, J., Choi, S., Shin, S. and Sung, W. (2016). FPGA-based low-power speech recognition with recurrent neural networks, *ArXiv* 1610.00552 [cs.CL].

Malhotra, P., Vig, L., Shroff, G. and Agarwal, P. (2015). Long short term memory networks for anomaly detection in time series, *23rd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, ESANN 2015, Bruges, Belgium*, pp. 89–94.

Marchi, E., Vesperini, F., Eyben, F., Squartini, S. and Schuller, B. (2015a). A novel approach for automatic acoustic novelty detection using a denoising autoencoder with bidirectional LSTM neural networks, *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brisbane, Australia*, pp. 1996–2000.

Marchi, E., Vesperini, F., Weninger, F., Eyben, F., Squartini, S. and Schuller, B. (2015b). Non-linear prediction with LSTM recurrent neural networks for acoustic novelty detection, *2015 International Joint Conference on Neural Networks (IJCNN), Killarney, Ireland*, pp. 1–7.

Morton, J., Wheeler, T.A. and Kochenderfer, M.J. (2016). Analysis of recurrent neural networks for probabilistic modelling of driver behaviour, *IEEE Transactions on Intelligent Transportation Systems* **PP**(99): 1–10.

Pouladi, F., Salehinejad, H. and Gilani, A.M. (2015). Recurrent neural networks for sequential phenotype prediction in genomics, *2015 International Conference on Developments of E-Systems Engineering (DeSE), Dubai, UAE*, pp. 225–230.

Schmidt, R. (2016). Machine protection and interlock systems for circular machines—Example for LHC, *Yellow Report CERN-2016-002*, CERN, Geneva, pp. 319–341.

Skoczeń, A. and Skała, A. (2009). Commissioning of quench protection instruments in the LHC superconducting circuits, *Electrical Review* **85**(7): 73–77, (in Polish).

Steckert, J. and Skoczeń, A. (2017). Design of FPGA-based radiation tolerant quench detectors for LHC, *Journal of Instrumentation* **12**(04): T04005–T04005.

Strecht, P., Cruz, L., Soares, C., Mendes-Moreira, J. and Abreu, R. (2015). A comparative study of regression and classification algorithms for modelling students' academic performance, *Proceedings of the 8th International Conference on Educational Data Mining, EDM 2015, Madrid, Spain*, pp. 392–395.

Wenninger, J. (2016). Machine protection and operation for LHC, *Yellow Report CERN-2016-002*, CERN, Geneva, pp. 377–401.

Wielgosz, M., Mertik, M., Skoczeń, A. and Matteis, E.D. (2018a). The model of an anomaly detector for HiLumi LHC magnets based on recurrent neural networks and adaptive quantization, *Engineering Applications of Artificial Intelligence* **74**: 166–185.

Wielgosz, M. and Skoczeń, A. (2018). Recurrent neural networks with grid data quantization for modeling LHC superconducting magnets behavior, *Contemporary Computational Science*, AGH-UST Press, Cracow, p. 240.

Wielgosz, M. and Skoczeń, A. (2020). Recurrent neural networks with grid data quantization for modeling LHC superconducting magnets behavior, *in* P. Kulczycki *et al.* (Eds), *Information Technology, Systems Research and Computational Physics*, Advances in Intelligent Systems and Computing, Vol. 945, Springer, Cham, (in press).

Wielgosz, M., Skoczeń, A. and De Matteis, E. (2018b). Protection of superconducting industrial machinery using RNN-based anomaly detection for implementation in smart sensor, *Sensors* **18**(11), Article ID: 3933.

Wielgosz, M., Skoczeń, A. and Mertik, M. (2017). Using LSTM recurrent neural networks for detecting anomalous behavior of LHC superconducting magnets, *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **867**: 40–50.

Wielgosz, M., Skoczeń, A. and Wiatr, K. (2018c). Looking for a correct solution of anomaly detection in the LHC machine protection system, *International Conference on Signals and Electronic Systems (ICSES), Cracow, Poland*, pp. 257–262.

Maciej Wielgosz received his PhD (with honors) in 2010 in high-performance reconfigurable computing from the AGH University of Science and Technology, Cracow, Poland. He is currently an assistant professor in the Department of Electronics, AGH-UST, and works in the Academic Computing Center CYFRONET. His primary area of research interests is cognitive computing, with the emphasis on real-time anomaly detection.

Andrzej Skoczeń is an assistant professor in the Particle Interaction and Detection Techniques Department at the Faculty of Physics and Applied Computer Science of the AGH University of Science and Technology in Cracow. He received his PhD in physics in the field of semiconductor devices in 1993. Currently his activities focus on digital design, including ASIC or FPGA devices. His cooperation with CERN concerns the design of instruments for quench protection systems (QPS) of the LHC.