Annales Univ. Sci. Budapest., Sect. Comp. 56 (2024) 69–82

THEORETICAL BACKGROUND OF NUMERICAL-BASED INPUT DATA EXTENSION AND RELIABILITY ANALYSIS FOR MACHINE LEARNING

Gergely Bencsik and Zoltán Pödör

(Szombathely, Hungary)

Communicated by Attila Kiss

(Received August 9, 2023; accepted January 7, 2024)

Abstract. Nowadays, Artificial Intelligence (AI) and Machine Learning (ML) models are becoming increasingly widespread in practice and every area. The central issue of AI and ML is to create the best model based on the available input data. Building and choosing the best model are often not trivial tasks. These models usually work well if the quantity and the quality of the training datasets are appropriate. Thus, the process of the model building and the raw data itself interact: the characteristics of the data determine the possibilities of the model building. However, data are not always available in large volume. Although smartening industrial, or any other environments up has been going on in recent years, it is still a problem when the data are very 'homogeneous', i.e., there is a relatively large amount of data, but much of it is irrelevant from the point of view of AI or ML modelling. Thus, false models can be built that do not work properly, especially if there are changes in the characteristics of the data. In this paper, the theoretical background of two new, self-developed methods are presented which are able to systematically increase the volume of the input data and examine the reliability and the stability of the applied analysis method. These two methods are able to work together in a framework and can extend the use and the applicability of AI models with new, systematically generated datasets.

Key words and phrases: Data science, ECReMIT, random correlations.
2010 Mathematics Subject Classification: 68P99.
The research was supported by the project No. 2019-1.3.1-KK-2019-00011 financed by the National Research, Development and Innovation Fund of Hungary under the Establishment of Competence Centers, Development of Research Infrastructure Program funding scheme.

https://doi.org/10.71352/ac.56.069

1. Introduction

Artificial Intelligence (AI) and Machine Learning (ML) models based on statistics, Data Mining techniques, Big Data and Internet of Things (IoT) are often referred to as Data Science. The application of Data Science methods - in almost all aspects of life - are becoming increasingly dominant [11], [13], [16]. By 2030, specialists expect a major scientific breakthrough that will affect the industry, and, with it, many jobs as well [7], [14], [18]. Economic implications, together with the previous ones, could transform the whole of economics and society as we know it [1]. Along with this, the application of Data Science models is not a trivial process, each problem may require a different approach and conditions. Therefore, in recent years, newer and newer models have been created in order to be as accurate as possible [12], [19].

A given Data Science problem can be divided into several sub-processes, such as data pre-processing, feature selection, choosing of data science model, which have been summarized by several authors [5], [9]. However, in practice, it is a hard, maybe even impossible task to define a general process for different problems. Nowadays, the data is often generated by sensors and the measured datasets are collected, transformed, and stored in a database. Collecting and storing a large amount of data technically is not a problem today, there are many hardware and software solutions utilize for these tasks. However, the reliable and efficient processing of this data is still a big challenge.

The (industrial) applications of Data Science methods consist of two main steps. The first step is always a general statistical analysis of the data series to get a general overview of them, and some kind of pre-processing or transformation steps to increase the data quality. The result of this step helps us to define the adequate analysis method in the next phase. The second step, which we often call modelling, is the implementation of the Data Science method itself on the pre-processed input dataset. As indicated earlier, these two steps cannot be truly separated, as the characteristics and quality of the input data will influence the choice of the method. Based on the 4V (volume, variety, velocity and veracity) definition of Big Data, we often consider collected data that have no effect on the value of the dependent variable we are looking for. In practice, we have a lot of data, but these datasets are not always suitable for explaining the dependent variable. We cannot create new data besides what we already obtained in the past. However, we can generate new, derived datasets from the available collected data, thus expanding the range of the relevant input data.

In this paper, we present the theoretical background and description of two new methods that extend the applicability of Data Science methods by generating data series derived from the originally measured sensor data in industrial environments. The first new method is the ECReMIT (Extended Cyclic Re-

verse Moving Intervals Technique) method, which uses time series to produce the new derived time series described above. This new dataset can be the input, or the training set of the applied analysis method. However, the systematic extension of data series can inevitably create a data environment where correlations or, in the case of AI and ML models, high accuracies can be achieved by chance. The larger the size of the input, the higher the probability of finding a good relationship. The increased input data space increases the probability for finding random relationships or producing good accuracies. Random Correlations (RC) aims to reduce correlations due to randomness and analyze the reliability and stability of the applied analysis methods and the results. Our main goal is to establish the extension opportunities of Artificial Intelligence and Machine Learning models and to test their stability at the data level, but the analytical method can be basically any method. The two methods are independent of the model built, so we will use the word modelling as a generic term hereafter. In this paper, our main aim is to introduce these two methodologies and to show how they can be linked into a complex framework.

2. Theoretical background of the methods

In data analysis, connected to a measurement we often have only one input time series, X which includes the measured data. The ECReMIT is responsible for the generating of the whole, complex dataset according to the parametrization by the user based on X. The output dataset of ECReMIT is the new, extended input of the data analysis. The examination of the reliability and the stability of these analysis methods and their results is the task of RC. In this chapter we provide the details, the theoretical background and notations of these two methods.

2.1. ECReMIT

The Extended CReMIT (ECReMIT) provides the opportunity to generate new, derived time series from X, based on the periodicity of it. The basic CReMIT method [15] was developed to improve the efficiency and completeness of time series analysis methods by systematically increasing the set of possible input data. The main idea behind CReMIT was a window-based technique to create new time series using special aggregation, transformation functions on the data in windows over the original dataset.

However, the basic CReMIT method uses the whole, available time series X to create the new, secondary time series (STS). Nevertheless, sometimes we do not want to use the whole length of the time series, but only a continuous subpart of it. Because this approach gives the opportunity to examine the changes in time series and in the examined relationships. The evolutionary and

moving interval techniques in Fig. 1 make it possible to use only a given part of X, but in a systematic way. The essence of the moving interval technique is that the length of the examined interval is always fixed, and the starting point is moved forward by one period in each iteration step. In the case of the evolutionary technique the starting point is fixed, and the interval length is increased by one period step by step [4]. Both two techniques use only a continuous part of the whole, original time series in a given step.

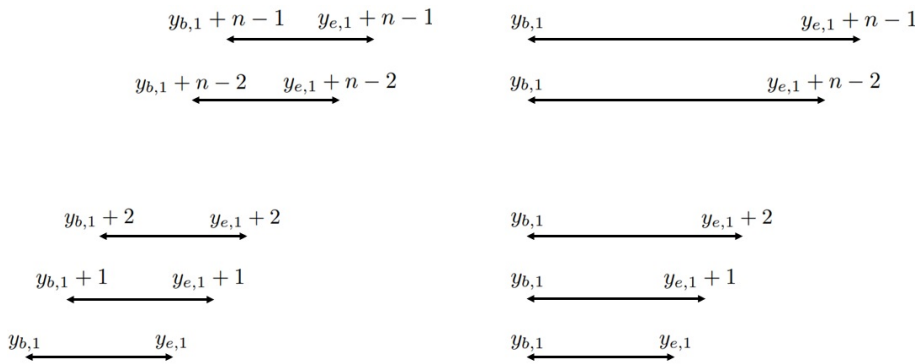

Figure 1. (a) Moving interval technique (b) Evolutionary technique

These techniques allow us to further increase the number of input datasets, but in a systematic way. Using Machine Learning algorithms, it is important to create as big a learning dataset as possible. The ECReMIT method helps us to create a huge number of input time series based on the original one, which can be the input set of the Machine Learning algorithms to improve, optimize the accuracy of the results.

Let $X = \{x_1,\ x_2,\ \ldots,\ x_{i-1},\ x_i,\ x_{i+1},\ \ldots,\ x_n\}$ a time series with the next four, given properties:

1. P is the periodicity of X,
2. n is the length of X,
3. x_1 is the chronologically latest value in X,
4. NP is the exact number of periods in X (e.g., if we have monthly data in X, then NP is equal with the number of whole years in X).

The ECReMIT method has six, user defined input parameters to determine the running conditions of the algorithm. These parameters determine the number and properties of the created new, secondary time series (STS):

1. The starting point index, SP $(1 \leq SP \leq P)$ defines the starting point x_{SP} of the algorithm in SP, this value will be the beginning of the first window over SP. In practice, SP is usually much smaller than n.

2. The maximum window width, MWW defines the maximum size of the windows over X. There is a running index in the algorithm, the actual window width, AWW and it covers the interval $[1,\ MWW]$.

3. The maximum time shifting value, MTS defines the maximum distance between (X_{SP}) and the first windows over X. There is a running index in the algorithm, the actual time shifting value, ATS and it covers the interval $[0,\ MTS]$.$ATS = 0$ means that the starting point of the first window is the index SP.

4. The transformation function, TRF. It defines the method to create the values of the new time series based on the windows values over X. TRF can be a simple *average*, *sum*, *min*, *max*, or more complex transformations, it depends on the given examination.

5. The number of the examined periods, NEP. This parameter allows us not to take the whole time series with NP into account, but only fewer periods. This provides the opportunity to examine the possible temporal changes in X according to the evolutionary or moving interval techniques. When the evolutionary technique is used then the value of NEP is increasing in each main iteration step by one. When the moving interval technique is used then the value of NEP is constant in whole process.

6. A binary variable to define that the algorithm uses the evolutionary or the moving interval technique, T. If $T = 0$ then the evolutionary if $T = 1$ then the moving interval technique is used in the algorithm.

According to these six input parameters the algorithm uses the next values to create the windows and STS-s:

1. The beginning period, BP. It is the first, chronologically the latest period, which is considered during the actual window creation over X. The initialization value of BP is 1, it is the first period in X. When the evolutionary technique is used then the value of BP is constant.

2. The ending period, EP. It is the last, chronologically the oldest period, which is considered during the actual window creation over X. The initial value of EP is $BP + NEP$. When moving the interval technique is used then the value of BP increases in each main iteration step by one period.

3. The number of all STS-s,

$$NSTS\,(MWW,\ MTS,\ NEP) =$$
$$= MWW \times (MTS + 1) \times (NP - NEP + 1)\,,$$

where z is the index of a given STS and

$$1 \leq z \leq NSTS\,(MWW,\ MTS,\ NEP)\,.$$

4. A new STS is defined by the next five, user defined parameters,
$$STS_z\,(SP,\ AWW,\ ATS,\ NEP,\ TRF).$$

5. The length of STS, or with other words, the number of the opened windows over the time series
$$NOW = [(n - (SP + AWW + ATS - 1))\,/P] + 1,$$
where [ ] is the entire function, but only between BP and EP.

Based on the six user defined parameters ECReMIT creates the above described five parameters in each iteration step and creates all of the STS-s on the output. The STS-s are stored in a matrix $MoSTS$ (*Matrix of Secondary Time Series*), where each row is according to a new, derived STS. STS_z is stored in the z^{th} row of $MoSTS$. The size of matrix $MoSTS$ is $NSTS\,(MWW, MTS, NEP) \times NoW$, with the user defined parameters, it is:

$$(MWW \times (MTS + 1) \times (NP - NEP + 1)) \times$$
$$\times\,([(n - (SP + AWW + ATS - 1))\,/P] + 1).$$

The actual length of an STS depends on the parameters, the empty cells are filled with NA (Not Applicable) values in the matrix. In a given window, we usually create the sum or the average of the values in the window, which is a simple elementary operation. The time complexity of the algorithm ECReMIT is the linear function of the above-mentioned parameters. The basic CReMIT method was applied by many practical problem connected to forestry and climate change [6], [8], [10].

2.2. Random correlations

As mentioned before, the goal is always to find some kind of relationship between data, or to automate natural or industrial processes based on data that we have processed with learning or analysis algorithms. However, despite the research being conducted about the same question in the same field under identical conditions the results can be conflicting. Random Correlations as a method seeks to find answers to such contradictions. The main idea behind the Random Correlations is to aim to discover the false relationships that appear among the methodologically correct results. Sometimes the collected data, based on their length and the applied transformations on them, can cause some relationship without practical meaning. There are several methods to

test the reliability of the results which create for example R^2 and various other statistical values. In the case of Machine Learning, many similar properties can be found, such as accuracy or $F1$ score which calculates the harmonic mean of precision and recall scores of the models or ROC (Receiver Operating Characteristic) curve which graphically illustrates the performance of the model at various thresholds [17]. The theory of Random Correlations does not replace the methods used to gain the above-mentioned properties, its aim is to eliminate the randomness of the results, thus increasing the reliability of the models. The main difference between reliability analyses and the Random Correlations is the way in which "bad" results are approached. If the reliability analysis of the results is adequate, we are more likely to believe that the relationship we are looking for really exists. The Random Correlations technique, on the other hand, assumes that, under certain circumstances, despite the good reliability value of the results their existence might be untrue.

The secondary datasets generated by the ECReMIT method are stored in the matrix $MoSTS$ and each row z in $MoSTS$ contains a new derived datarow. This extension of the data can create an environment where the models built are highly accurate, but in reality, this accuracy is the result of chance. As more and more data sets are produced, the accuracy of the models built may inevitably increase. To filter this random property, we compute the variance D_z for each datarow z. Let denote by c the constant step to produce all possible values that can be measured given the $MoSTS$ element and the variance D_z according to the following steps:

1. The vectors $VV_{z,w}$ ($Values\ of\ Values$) are produced based on the following rule:

$$\begin{aligned} VV_{z,w} = \ & [vv_{z,w}] = \\ & = [mosts_{z,w}] - D_z + i \times c_z\left[w\left[z\right]\right],\ i = 0,\ 1,\ 2,\ \ldots,\ k_z\left[w\left[z\right]\right], \\ & k_z\left[w\left[z\right]\right] = D_z/c_z\left[w\left[z\right]\right],\ D_z mod c_z\left[w\left[z\right]\right] = 0. \end{aligned}$$

2. Based on values of vectors VV, all combinations are produced. We go through the indices of the vectors VV systematically. In each iteration, changing the index of one vector VV and leaving the others unchanged will result in the next tertiary data series.

Since the rows of the $MoSTS$ are not the same length, the lengths are stored in vector w. Similarly, the constants are different for each element of each row and they are stored in vectors c_z. The constants are freely defined by the user, but the given variance D_z must be divided by it without a residue, otherwise the last element will not be obtained properly. All combinations as the tertiary data series for all data series z are stored in $MoRC_z$ ($z^{th}\ Matrix\ of\ RC$). Each data row of the matrix $MoSTS$ is replaced by the corresponding tertiary

data row $MoRC_z$ systematically. The general generation process of matrices $MoRC$ can be seen in Fig. 2.

In step S, the replacement is performed for each data row z and then the model is built based on matrix $MoSTS$ containing the new, replaced data rows. At each step S, we record the value of the variable measuring the accuracy or some kind of goodness-of-fit entity of the model. After step S, if these values have small variance, i.e., if the variance is below a certain level α, the model is stable. If α is large, the model is not stable, i.e., a small change in the original data can lead to large model degradation. The α allows not only to filter out randomness, but it also allows us to see the impact of the possible original data value changes on the model itself.

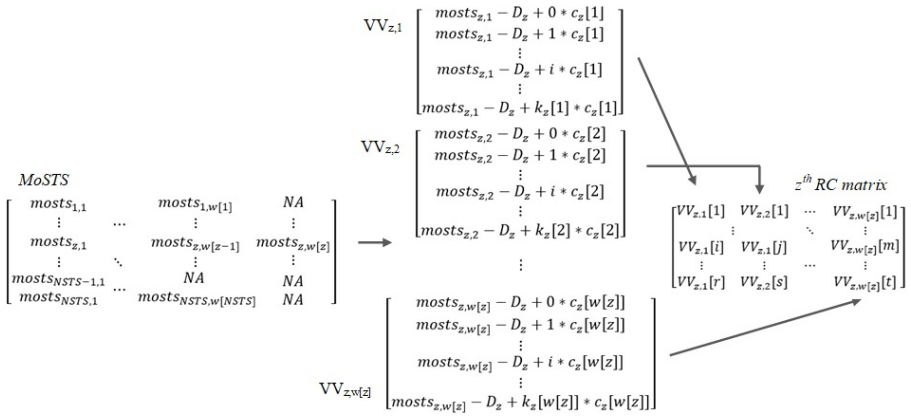

Figure 2. Basic generation process of $MoRC$ matrices

S is also an input parameter. To eliminate randomness, all possible combinations of all possible exchanges of all data series z would have to be generated, and the model would have to be built in each iteration. This is a very large data space and although the method is highly parallelizable, it is still infeasible in human time sometimes. The newly created datasets allow deeper analysis, but the operational complexity is exponential. Just generating the possible measured values for each value of the data series z and the constant c_z is a computationally expensive task, then, the combination phase has exponential time complexity. Thus, the computational demand should be reduced. A trivial reduction of the computational demand is if the parameter value of S is smaller and the step $c_z\,[w\,[z]]$ is larger. A further reduction is to look for correlation between the tertiary data series of each $MoRC_z$. In this case, classes denote by $C_{z,1},\ C_{z,2},\ \ldots,\ C_{z,t}$ are created and it is true that all data series within a class are correlated. We take the data series in order, the first one clearly generates class $C_{z,1}$. If the next data series is correlated to the previous one, it

is placed in class $C_{z,1}$, otherwise class $C_{z,2}$ is created with one element. Then, we do not need to perform all possible swaps, but only form a combination of data series representing classes.

From the RC point of view, data stability can also be examined in terms of time. In this case, the NEP parameter of ECReMIT is systematically chosen so that one or p pieces of data are always omitted backwards along the time axis. This is done as long as the omission of data is meaningful or until a certain predefined l limit is reached, below which no more data can be examined. In each iteration, we train a model and then use an α to evaluate the variance of these models. If α is large, then the accuracies or the model goodness-of-fit ratios are highly deviated, the model is not considered stable, and the data suggest that there is a high probability that the model will probably not perform well later. The limit of the α is also defined by the user. The numerical results of RC related to ANOVA (Analysis of Variance) and regression techniques were showed out before [2], [3].

3. Data process pipeline

The introduced methods are implemented as part of a complex data processing pipeline. The Fig. 3 shows the main concept of the data handling process.

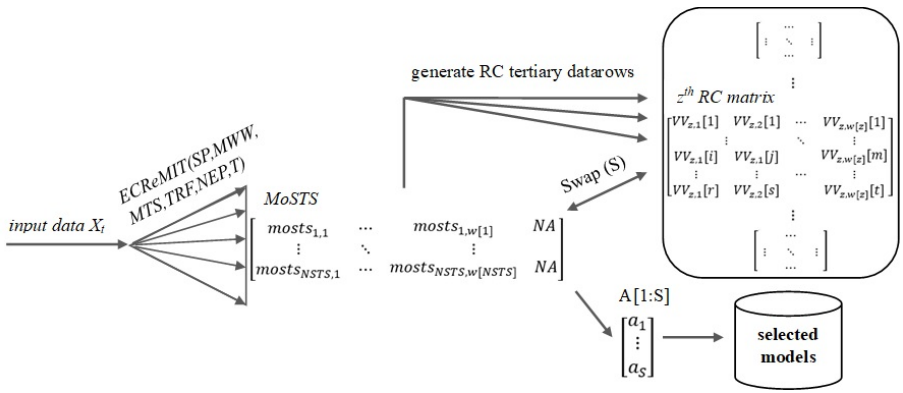

Figure 3. Concept of the pipeline

The pipeline starts with the original time series, X. A complex analysis process contains the next main steps:

1. ECReMIT module gets the input dependent variable X, and generates the new, derived time series, $STS\,(SP,\ AWW,\ ATS,\ TRF)$ from the dependent variable (s) and stores them in matrix $MoSTS$.

2. All the time series from $MoSTS$ can be used as training set to build the data analysis model (m).
3. RC module verifies the stability of these models with generating tertiary data rows based on the $MoSTS$ rows.
4. Swap $MoSTS$ data rows for $MoRC$ data rows. Accuracies or some similar entities are notified in vector A and model selection are made based on the limit α.

The implementation of the framework can be different, adjusted to the given research, but pseudo code can help to implement the framework easier. Thus, the main steps of the framework's algorithm pseudo code according to the notations of the definitions is introduced.

Algorithm 1 ECREMITRC

```
1:  procedure ECREMIT(SP, MWW, MTS, P, NEP, T and Function
                      TRF)
2:      initialization parameters SP, MWW, MTS, P, NEP, T and
        Function TRF
3:      MoSTS := empty matrix
4:      z := 0
5:      BP := 1
6:      EP := BP + NEP
7:      if (T == 0) then                          ▷ evolutionary technique
8:          for i := 1 to (NP − BP + 1) do
9:              NEP := EP − BP + 1
10:             for AWW := 1 to MWW do
11:                 for ATS := 0 to MTS do
12:                     z := z + 1
13:                     NoW := [(n − (SP + AWW + ATS − 1)) /P] + 1
                                between BP and EP
14:                     STS (SP, AWW, MTS, NEP, TRF)_z := empty array
15:                     for t := 1 to NoW do
16:                         open the window over X according to SP, AWW,
                            ATS and P
17:                         use TRF over the window's values
18:                         create STS (SP, AWW, MTS, NEP, TRF)_z [t]
19:                     end for
20:                     MoSTS [z,] := STS (SP, AWW, MTS, NEP, TRF)_z
21:                     EP := EP + i
22:                 end for
23:             end for
24:         end for
25:     end if
```

```
26:     if (T == 1) then                          ▷ Moving interval technique
27:         NEP := EP − BP + 1
28:         for i := 1 to (NP − BP + 1) do
29:             for AWW := 1 to MWW do
30:                 for ATS := 0 to MTS do
31:                     z := z + 1
32:                     NoW := [(n − (SP + AWW + ATS − 1)) /P] + 1
                                   between BPandEP
33:                     STS (SP, AWW, MTS, NEP, TRF)_z := empty array
34:                     for t := 1 to NoW do
35:                         open the window over X according to SP, AWW,
                            ATS and P
36:                         use TRF over the window's values
37:                         create STS (SP, AWW, MTS, NEP, TRF)_z [t]
38:                     end for
39:                     MoSTS [z, ] := STS(SP, AWW, MTS, NEP, TRF)_z
40:                 end for
41:             end for
42:             BP := BP + i
43:             EP := EP + i
44:         end for
45:     end if
46: end procedure
47: procedure RC(NSTS, c[ ], α)
48:     for z := 1 to NSTS do
49:         A_z := AVERAGE (MoSTS [z, ])
50:         D_z := DEVIATION (MoSTS [z, ])
51:         for w := 1 to w[z] do
52:             DEFINE (c_z [w])
53:             VV_{z,w[z]} := CREATEALLVALUES (D_z, c_z [w])
54:         end for
55:         MoRC [z, ] := CREATEALLCOMBINATIONS (VV_{z,w[z]})
56:         NoC [1] := 1                      ▷ number of classes and members
57:         for i := 1 to p do
58:             for j := 1 to NoC [p] do
59:                 if CORRELATE (MoRC_z) then
60:                     ADD (MoRC_{z,i}, C_{z,j})
61:                     NoC [i] := NoC [i] + 1
62:                 else
63:                     p := p + 1
64:                     NoC [p] := 1
65:                     CREATE(C_{z,p}, MoRC_{z,i})
```

```
66:                 end if
67:             end for
68:         end for
69:     end for
70: end procedure
71: procedure EVALUATION(S, NSTS, α)
72:     A := empty vector
73:     for i := 1 to S do
74:         for z := 1 to NSTS do
75:             SWAP(MoSTS[z,], MoRC_z, RANDOM(C_{z,p}),
                        RANDOM(C_{z,NoC[p]}))
76:             A[i] := MODEL(MoSTS)
77:             MODELS SELECTION(α)
78:         end for
79:     end for
80: end procedure
```

4. Conclusion

Data Science algorithms require a lot of input data. We have to use more data than we may reasonably require in classical statistics. Estimating the amount of data needed for Machine Learning models is critical in any data science project. When determining the volume of data necessary for an ML model, factors such as the type of problem being solved, the complexity of the model, the quality and accuracy of the data, and the availability of labeled data all come into play.

We usually need thousands of examples in training dataset. Ideally, tens or hundreds of thousands for "average" modelling problems. Millions or tens-of-millions for "hard" problems like those tackled by deep learning. Keep in mind that Machine Learning is a process of induction. The model can only capture what it has seen. If your training data does not include enough and adequate cases, they will very likely not be supported by the model.

The ECReMIT method is able to create a big size input time series from only one long enough time series based on the periodicity of it. It significantly increases the size of the training data in a systematic way. The RC technique always provides a kind of measurement based on derived data that could have been measured at all, and based on these values, it takes measurement inaccuracies and randomness into account. RC tries to explain to what extent the change of data, i.e., randomness, can affect the subsequent use of the model. In this article, our goal was to introduce these two new methodologies and to establish the theoretical background of these two algorithms' connection into one framework.

Our aim in the future is to build these methodologies into a pipeline, that covers the entire analysis process from the prepared basic input to the reliability of the obtained results. This pipeline, according to ECReMIT and mainly RC creates a huge number of training sets. The parallel processing gives an opportunity to use and run this pipeline and in the near future the quantum computers can be the solution for the significant computing capacity problem.

References

[1] **Al-Sarawi, S., M. Anbar, R. Abdullah and A.B. Al Hawari,** Internet of things market analysis forecasts, 2020—2030, in: X.-S. Yang (Ed.) *Fourth World Conference on Smart Trends in Systems Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, (London 2020), IEEE, London, 2020, 449–453.

[2] **Bencsik, G. and L. Bacsárdi,** Novel methods for analyzing random effects on ANOVA and regression techniques, *Advances in Intelligent Systems and Computing*, **416** (2016), 499–509.

[3] **Bencsik, G. and L. Bacsárdi,** New methodology to analyze the random impact level of mathematically proved results, in: A. Szakál (Ed.) *15th International Symposium on Computational Intelligence and Informatics (CINTI)*, Budapest 2014, IEEE, New York, 2014, 1365–1374.

[4] **Biondi, F. and K. Waikul,** DENDROCLIM2002: A C++ program for statistical calibration of climate signals in tree-ring chronologies, *Comp. Geosci*, **30** (2004), 303–311.

[5] **Chen, Z., S. Zhu, Q. Niu and T. Zuo,** Knowledge discovery and recommendation with linear mixed model, *IEEE Access*, **8** (2020), 38304–38317.

[6] **Csóka, G., A. Hirka, L. Szőcs, N. Móricz, E. Rasztovits and Z. Pödör,** Weather-dependent fluctuations in the abundance of the oak processionary moth, Thaumetopoea processionea (Lepidoptera: Notodontidae), *European Journal of Entomology*, **115** (2018), 249–255.

[7] **Denny, J.C. and F.S. Collins,** Precision medicine in 2030—seven ways to transform healthcare, *Cell*, **184** (2021), 1415–1419.

[8] **Führer, E., M. Edelényi, L. Horváth, A. Jagodics, L.T. Jereb, Z. Kern and I. Szabados,** Effect of weather conditions on annual and intra-annual basal area increments of a beech stand in the Sopron Mountains in Hungary, *Időjárás/Quarterly Journal of the Hungarian Meteorological Service*, **120** (2016), 127–161.

[9] **Huber S., H. Wiemer, D. Schneider and S. Ihlenfeldt,** DMME: Data mining methodology for engineering applications – a holistic extension to the CRISP-DM model, *Procedia CIRP*, **79** (2019), 403–408.

[10] **Janik, G., Z. Pödör, A. Koltay, A. Csókáné Hirka, J. Juhász, G. Kovács and G. Csóka,** Effects of meteorological and site parameters on the health status of beech (Fagus sylvatica L.) forests in Hungary, *Energy Reports*, **5** (2019), 1365–1374. *Acta Silvatica et Lignaria Hungarica*, **16** (2020), 67–78.

[11] **Javaid, M., A. Haleem, S.R. Pratap and R. Suman,** Artificial intelligence applications for industry 4.0: A literature-based study, *Journal of Industrial Integration and Management*, **7** (2022), 83–111.

[12] **Oreshkin, B.N., D. Carpov, N. Chapados and Y. Bengio,** N-BEATS: Neural basis expansion analysis for interpretable time series forecasting, arXiv preprint (2020), https://arxiv.org/pdf/1905.10437.pdf

[13] **Özdemir, V. and N. Hekim,** Birth of industry 5.0: Making sense of big data with artificial intelligence, "The internet of things" and next-generation technology policy, *OMICS: A Journal of Integrative Biology*, **22** (2018), 65–76.

[14] **Palomares, I., et al.,** A panoramic view and swot analysis of artificial intelligence for achieving the sustainable development goals by 2030: progress and prospects, *Applied Intelligence*, **51** (2021), 6497–6527.

[15] **Pödör, Z., M. Edelényi and L. Jereb,** Systematic analysis of time series – CReMIT, *Infocommunication Journal*, **VI** (2014), 16–22.

[16] **Sircar, A., K. Yadav, K. Rayavarapu, N. Bist and H. Oza,** Application of machine learning and artificial intelligence in oil and gas industry, *Petroleum Research*, **6** (2021), 379–391.

[17] **Sokolova, M., N. Japkowicz and S. Szpakowicz,** Beyond accuracy, F-Score and ROC: A family of discriminant measures for performance evaluation, AI 2006: *Advances in Artificial Intelligence*, **6** (2006), 1015–1021.

[18] **Wu, F., et al.,** Towards a new generation of artificial intelligence in China, *Nature Machine Intelligence*, **2** (2020), 312–316.

[19] **Yu, D., Y. Wang, H. Liu, K. Jermsittiparsert and N. Razmjooy,** System identification of PEM fuel cells using an improved Elman neural network and a new hybrid optimization algorithm, *Energy Reports*, **5** (2019), 1365–1374.

G. Bencsik and Z. Pödör
Eötvös Loránd University
Szombathely
Hungary
bg@inf.elte.hu and pz@inf.elte.hu