OINOS, an application suite for the performance evaluation of classifiers

Emanuele Paracone
Dept. of Civil Engineering and Computer Science
University of Rome "Tor Vergata"
Rome, Italy
emanuele.paracone@gmail.com

***Abstract*—The last few years have been characterized by a big development of machine learning (ML) techniques, and their application has spread in many fields. The success of their use in a specific problem strongly depends on the approach used, the dataset formatting, and not only on the type of ML algorithm employed. Tools that allows the user to evaluate different classification approaches on the same problem, and their efficacy on different ML algorithms, are therefore becoming crucial.**
In this paper we present OINOS, a suite written in *Python* and *Bash* aimed to the evaluation of performances of different ML algorithms. This tool allows the user to face a classification problem with different classifiers and dataset formatting strategies, and to extract related performance metrics. The tool is presented and then tested on the classification of two diagnostic species from a public electroencephalography (EEG) database. The flexibility and ease of use of this tool allowed us to easily compare the performances of the different classifiers varying the dataset formatting and to determine the best approach, obtaining an accuracy of almost 75%.
OINOS is an open source project, therefore its use and sharing are encouraged.

***Index Terms*—Machine learning, Classification, EEG.**

I. Introduction

In recent years we have witnessed the development of new machine learning (ML) techniques and the improvement of the existing ones, and their application has expanded in many fields [1]–[6]. Contemporarily, Python programming language has seen a surge in popularity across the sciences and in particular in neuroscience [7] for reasons which include its readability, modularity, and the large libraries available. Python's versatility is today evident in its range of uses.
With the aim of carrying out classification, regression and / or clustering on a specific problem, it is useful to evaluate the performances of different ML tools and the different dataset formatting strategies, for studying their behaviour with respect to the different scenarios.
In this work we present *OINOS*, a suite for the evaluation of classifier performances, composed of a set of modules for the comparison of ML algorithms with respect to different dataset partitioning strategies. OINOS is written in Python and Bash, and implemented as an applicative for the execution of multithreaded benchmark.

In order to give an example of application, we considered electroencephalography (EEG) data related to the problem of alcoholic prediction, i.e., the classification between patients suffering from alcoholism and healthy patients, based on EEG times series of a second of brain activity. Such dataset has been chosen for the high prediction complexity and because the data is publicly available (at https://kdd.ics.uci.edu/databases/eeg/eeg.html). In this problem the ML tools learn to glean the correlations among the fluctuation of brain signals obtained from the different channels and their dependance on the subject's pathological state.
The use of a custom dataset partitioning procedure allowed us to find satisfactory performances without the need to overload the data preprocessing. OINOS has simplified us to find alternative approaches to train the classifiers.

The analyzed data belongs to a test concerning 122 subjects. From each of them it has been collected a set of 120 trial. Each trial consists in the measurement of 1 sec. of EEG signals caught from 64 electrodes placed on the subject's scalp. During the trials, the subjects were exposed to three kind of stimuli: une single image, two matching images or two non-matching images alternately. Since subjects belongs to the 2 category *alcoholic* and *non-alcoholic* and the stimuli to the 3 kind *single*, *matching* and *non-matching*, the EEG data have been labeled through those 2 cohordinates (e.g. if a trial has been caught from an alcoholic patient while he was looking to a non-matching couple of figure, the trial label will be alc-non-matching).

II. Execution

Here we describe the structure of the presented tool and its operation modes.
The algorithms and the logic underlying the classification processes of OINOS are implemented by the libraries *scikit-learn* [8]–[10].
The component modules are:

1) main: the entry point of the suite. This block is responsible for the execution and the orchestration of single modules;
2) OINOS core: the main component. It implements the logic of comparison among different ML algorithms;

3) datalogger: the module for the output management and the experiment reports.

A. Use

In order to start OINOS it is necessary to execute the starter present in the root directory of the project through the command: $ ./start.
In this way the program will return:

```
================
= OINOS V1.0 =
================
Select an option:
1. learn from the 'Alcoholic' dataset from UCI Knowledge Discovery in Databases
2. learn from the 'Wrist' dataset from NeuCube
[1,2, quit]:
```

From this menu it is possible to select on which dataset the prediction algorithms must be tested.

B. Alcoholic

By selecting the first option, OINOS will acquire the datasets of the database of the *UCI Knowledge Discovery in Databases*. Before to start the execution, OINOS will ask to the user:

1) to specify the dataset among those available;
2) to specify the destination path for the output
3) to specify which portion of the data (i.e., *ratio*) with respect to the overall dataset, will be used for testing.
4) to specify the number of executions of the same prediction test. The importance of this setting is notable because once fixed the cardinality of the training and test sets (through the ratio), the related elements will be randomly selected; of course at each run the performances will vary depending on the specific training set of each experiment and it may be reasonable to study them in a statistical sense.

At the end of this configuration phase the comparison between the prediction algorithms is executed.

a) Dataset description: The dataset shown by the *start* are named with a suffix that indicates the dataset cardinality. For example, if *data_100* is selected by the user, it will be executed the prediction on a sample of 100 elements.
If a *ratio* of 0.2 is specified, the prediction will be redistributed using 80 elements as training set and the remaining 20 as test set.

b) Execution: During the execution, messages of four comparison categories are printed on the standard output; at each category corresponds a different classification to be presented to the prediction algorithms:

1) *alcoholic-control*: the EEG time-series of the dataset are classified as pertaining to alcoholic (*alcohol*) or healthy (i.e., *control*) patients;
2) *single-matching-non matching images*: the EEG time-series pertain to participants which a single flickering image (*single*), two identical alternate images (*matching*) or two different alternate images (*non matching*) are shown;
3) *single-matching-nonmatching images for alcoholic and control*: the intersection of the two previous classifications is considered (alcoholic patient watching a single image, alcoholic patient watching two identical images, etc.);
4) *alcoholic-control extended*: the six classes of the previous step are considered and projected to the two classes *alcoholic* and *control*.

The execution goes through these categories in four phases, showing the results of the test on the screen in terms of:

- classification (*ALC-CTRL*, *SGL-MATCH-NONMATCH*, *ALC-CTRL/SGL-MATCH-NONMATCH*, *ALC-CTRL_EXT*, respectively)
- overall cardinallity of the dataset (for example 100 for $data\_100$)
- cardinality of the test set (for example 20 for $ratio = 0.2$)
- type of classifier under testing
- *accuracy* of the classifier, computed as

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

- *precision* of the classifier, computed as

$$Pr = \frac{TP}{TP + FP} \tag{2}$$

- *recall* of the classifier, computed as

$$Rec = \frac{TP}{TP + FN} \tag{3}$$

- *F1 Score* of the classifier, computed as

$$F1 = 2 \cdot \frac{Pr \cdot Rec}{Pr + Rec} \tag{4}$$

  where TP stands for true positive, TN for true negative, FP for false positive and FN for false negative. A TP is an outcome where the model correctly predicts the positive class; similarly, a TN is an outcome where the model correctly predicts the negative class. A FP is an outcome where the model incorrectly predicts the positive class, and a FN is an outcome where the model incorrectly predicts the negative class.

c) output: When the execution is finished, the output will be avaible at the path specified during the configuration phase:

- a microsoft excel file (.xlsx) with the report, as described above;
- a figure with the comparison between the different accuracy values.

C. Unattended mode

With this option it is possible to directly call the Python relative sources.
This allows the user the execution in *unattended mode*, useful for the implementaton of custom procedures and benchmarks. The related scripts are $./bin/oinos.py$ and $./bin/wrist.py$

respectively; the switch -h enables the *help*, which returns the following information to the user:

```
$ ./bin/oinos.py -h
usage:
$ python bin/main.py -d <data_path> -r <testing data
ratio> -o
<output path>
example:
$ ./bin/oinos.py -d data_100 -r 0.3 -v -o out
```

```
$ ./bin/wrist.py -h
usage:
$ python bin/main.py -r <testing data ratio> -o <output
path>
example:
$ ./bin/oinos.py -r 0.3 -v -o out
```

This menu makes possibile to select the dataset on which the prediction algorithms have to be tested.

III. Dataset: alcoholic

A. Dataset interpretation

This dataset comes from the database of the *Knowledge Discovery in Databases Archive* of the *University of California, Irvine* [11], that is part of a bigger dataset on the detection of genetic predisposition of human beings to the alcoholism [9].

In our case, the gathered data comes from an experiment conducted on 122 subjects, each of which underwent 120 trials of the same task.
The task consists in a second of EEG activity recorded while the subject is asked to watch alternatively:

- a single image (case identifiec as *single*, i.e., *SNGL*)
- two identical images (*matching*, i.e., *MATCH*)
- two different images (*non matching*, i.e., *NONMATCH*)

For each presented stimulus, ten trials of a second of activity, recorded by 64 electrodes, have been gathered. Electrodes were located on the head of the subject, to record fluctuations of postsynaptic activity [12], sampled at 256 Hz. We implemented our comparisons between the classifiers by considering the four classifications described in the section II-B0b. In the next section we will show the strong points and the advantages of this approcach.

B. Classification: a bottom up approach

Using the metadata of the experiment, the samples have been subdivided with respect to the type of stimulus given to the subject (SNGL, MATCH, NONMATCH) or on the bases of the type of subject (alcoholic, control). Different tests have been done to evaluate the performances of the classifiers considered, using different configurations (only subject type, only stimulus type, combined).
Unfortunately, neither classifiers who achieved greater success during repeated runs were able to achieve satisfying performances.

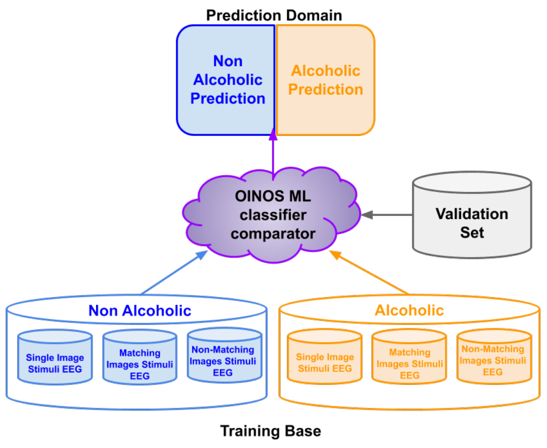

Fig. 1. Base aproach: OINOS compare several runs for each classification algorithm; each run takes as learning base a subset of the original base by considering only the label about alcoholism and ignoring the stimuli; the prediction domain is *alcoholic - control*.

Therefore we implemented a different method. In addition to the three types of prediction described above, we implemented - *alcohol control extended*, i.e., *alc-ctrl_ext* - able to project the classifications obtained by the combined configuration (six classes, one for each combination of pathology and stimulus) in the two classes *alcoholic* and *control*; we therefore classified the data in the most stringent way to go back into abstraction and generalize the final solution.

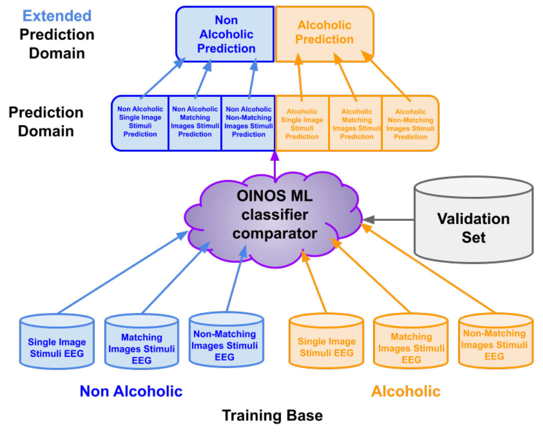
Extended Prediction Domain
Non Alcoholic Prediction
Alcoholic Prediction
Prediction Domain
Non Alcoholic Single Image Stimuli Prediction
Non Alcoholic Matching Images Stimuli Prediction
Non Alcoholic Non-Matching Images Stimuli Prediction
Alcoholic Single Image Stimuli Prediction
Alcoholic Matching Images Stimuli Prediction
Alcoholic Non-Matching Images Stimuli Prediction
OINOS ML classifier comparator
Validation Set
Single Image Stimuli EEG
Matching Images Stimuli EEG
Non-Matching Images Stimuli EEG
Single Image Stimuli EEG
Matching Images Stimuli EEG
Non-Matching Images Stimuli EEG
Non Alcoholic
Alcoholic
Training Base

Fig. 2. Extended aproach: OINOS compare several runs for each classification algorithm; each run takes as learning base a subset of the original base by considering both the label about alcoholism and stimuli; the prediction domain is made of all combination between*alcoholic-control* and *single-matching-nonmatching*; a further phase merges the obtained predictions into the *alcoholic extended* and *control extended* sets.

This new way to predict the classes *alcoholic* and *control* has significantly improved the performances of the classifiers, in the way we give illustration below.

C. Results

We have conducted a benchmark of 100 consecutive run to analyze the performances of the following classifiers:

- K Nearest Neighbors
- Linear SVM
- RBF SVM
- Linear SVC
- Gaussian Process
- Decision Tree (with max depth = 5)
- Decision Tree (with max depth = 10)
- Random Forest
- Gradient Boosting Classifier
- Neural Net
- Ada Boost
- Naive Baies
- Linear Discriminant Analysis
- QDA

After the execution of the run, an average for each one of the metrics has been done. Although the performances are not very good, it has to be noted that the introduction of the approach *acc-ctrl extended* has significantly affected the performance of some of the classifiers.

The two types of classifications *alcoholic - control* (in blue) and *alcoholic -control extended* (in orange) have been compared, underlining the benefits of the latter approach. The results are summarized in figures 3, 4, 5, 6.

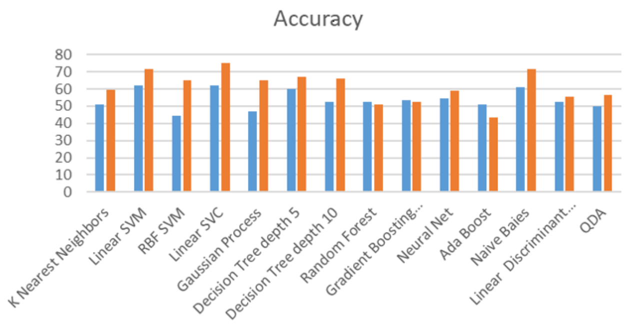

Fig. 3. Accuracy results are summarized, taking in account the considered classifiers. In orange are shown the results obtained by adopting the bottom-up approach whereas the blue ones shows the result of the "normal" aproach.

Among the different classifiers tested, it is worth highlighting the cases *Linear SVC* and *Neural Network*. Their classification for *alcoholic control* were just above the average of the other classifiers, with accuracy and precision near the 60%. Such performances have considerably improved with the extended approach.

The *University of California Knowledge Discovery Database Archive* (*UCI KDD Archive*) openly shares different datasets with the aim of make them usable for Machine Learning research (http://kdd.ics.uci.edu/). In the website different dataset are available, indexed for typology and semantic area. EEG data selected for our study are categorized into the section *Time Series - EEG* (http://kdd.ics.uci.edu/databases/eeg/eeg.data.html).

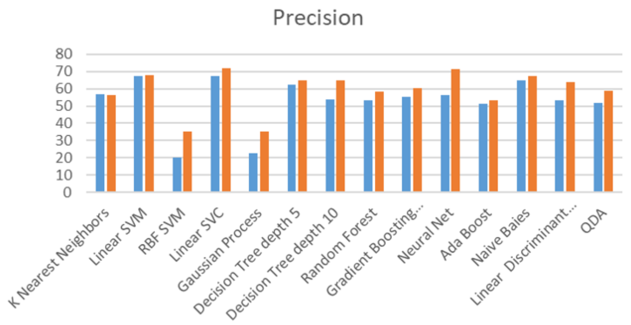

Fig. 4. Precision results are summarized, taking in account the considered classifiers. In orange are shown the results obtained by adopting the bottom-up approach whereas the blue ones shows the result of the "normal" aproach.

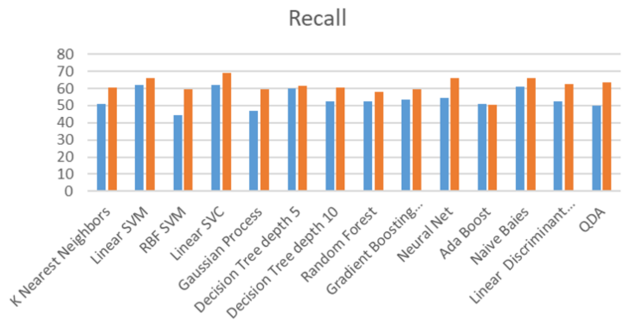

Fig. 5. Recall results are summarized, taking in account the considered classifiers. In orange are shown the results obtained by adopting the bottom-up approach whereas the blue ones shows the result of the "normal" aproach.

IV. Conclusion

In this work we present OINOS, a suite for the evaluation of classifier performances, composed of a set of modules for the comparison of several ML algorithms with respect to different datasets. We faced a classification problem based on neurophysiology data (i.e., EEG time series), to distinguish alcoholic to non/alcoholic subjects during the execution of a task. Through the performance evaluation of a set of classifiers we found the better configuration among the proposed classifiers and dataset formatting strategies. Although the big cardinality of the dataset, a need of alternative approaches for

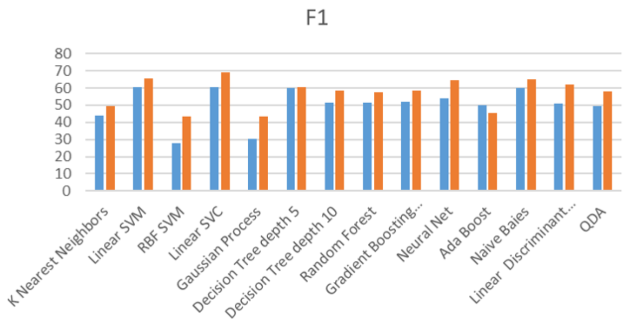

Fig. 6. F1 results are summarized, taking in account the considered classifiers. In orange are shown the results obtained by adopting the bottom-up approach whereas the blue ones shows the result of the "normal" aproach.

dataset formatting to facilitate the learning of the classifiers has emerged. The use of a custom procedure allowed us to find a way to improve the classification.

To show how to use OINOS, here we have performed the evaluation of classifiers for an application related to the biomedical field. Nevertheless, such kind of tools are of great help in many other fields [13], as financial [14], face recognition [15], and communications systems [16] where they could result useful for evaluating the performance of recent communication algorithms (e.g., [17], [18])). Finally, since some ML strategies are based on neural networks, a future development could be that of expanding classical artificial neural networks (ANNs) with the bio-inspired spiking neural networks (SNNs) [19]–[22], since recently such approaches are proving to be appropriate for classification/prediction of spatio-temporal stream data [23]–[25], and compare their performances on classical problems which approaches are classically faced with ANN [26], [27]

The work is an opensource project, available at https://gitlab.com/knizontes/oinos.

References

[1] S. Angra and S. Ahuja, "Machine learning and its applications: A review," in *2017 International Conference on Big Data Analytics and Computational Intelligence (ICBDAC)*, 2017, pp. 57–60.

[2] R. Malhotra, "A systematic review of machine learning techniques for software fault prediction," *Applied Soft Computing*, vol. 27, pp. 504 – 518, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1568494614005857

[3] M. Matta, G. C. Cardarilli, L. Di Nunzio, R. Fazzolari, D. Giardino, M. Re, F. Silvestri, and S. Span, "Q-rts: a real-time swarm intelligence based on multi-agent q-learning," *Electronics Letters*, vol. 55, no. 10, pp. 589–591, 2019.

[4] G. C. Cardarilli, L. Di Nunzio, R. Fazzolari, M. Re, and S. Span, "Awsom, an algorithm for high-speed learning in hardware self-organizing maps," *IEEE Transactions on Circuits and Systems II: Express Briefs*, pp. 1–1, 2019.

[5] S. Coco, A. Laudani, F. Riganti Fulginei, and A. Salvini, "Team problem 22 approached by a hybrid artificial life method," *COMPEL-The international journal for computation and mathematics in electrical and electronic engineering*, vol. 31, no. 3, pp. 816–826, 2012.

[6] S. Coco, A. Laudani, F. R. Fulginei, and A. Salvini, "Bacterial chemotaxis shape optimization of electromagnetic devices," *Inverse Problems in Science and Engineering*, vol. 22, no. 6, pp. 910–923, 2014.

[7] E. Muller, J. Bednar, M. Diesmann, M. Gewaltig, M. Hines, and A. Davison, "Python in neuroscience," *Frontiers in Neuroinformatics*, vol. 9, no. 11, pp. 62–76, 2015.

[8] INRIA, "Scikit learn." [Online]. Available: https://scikit-learn.org

[9] ——, "Scikit eeg." [Online]. Available: http://kdd.ics.uci.edu/databases/eeg/eeg.data.html

[10] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, and V. Dubourg, "Scikit-learn: Machine learning in python," *Journal of machine learning research*, vol. 12, pp. 2825–2830, 2011.

[11] S. D. Bay, D. F. Kibler, M. J. Pazzani, and P. Smyth, "The uci kdd archive of large data sets for data mining research and experimentation," *SIGKDD explorations*, vol. 2, no. 2, pp. 81–85, 2000.

[12] G. Susi, S. Ye-Chen, J. de Frutos Lucas, G. Niso, and F. Maestú, "Neurocognitive aging and functional connectivity using magnetoencephalography," in *Oxford research encyclopedia of psychology and aging*. Oxford: Oxford University press, 2018.

[13] A. Soofi and A. Awan, "Classification techniques in machine learning: Applications and issues," *Journal of Basic & Applied Sciences*, vol. 13, 2017.

[14] B. Henrique, V. Amorim Sobreiro, and H. K. S., "Literature review: Machine learning techniques applied to financial market prediction," *Expert Systems with Applications*, vol. 124, pp. 226 – 251, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S095741741930017X

[15] H. Filali, J. Riffi, A. M. Mahraz, and H. Tairi, "Multiple face detection based on machine learning," in *2018 International Conference on Intelligent Systems and Computer Vision (ISCV)*, April 2018, pp. 1–8.

[16] O. Simeone, "A very brief introduction to machine learning with applications to communication systems," *IEEE Transactions on Cognitive Communications and Networking*, vol. 4, no. 4, pp. 648–664, Dec 2018.

[17] A. Detti, M. Orru, R. Paolillo, G. Rossi, P. Loreti, L. Bracciale, and N. Blefari Melazzi, "Application to information centric networking to nosql database," in *2017 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*, 2017.

[18] A. Detti, L. Bracciale, P. Loreti, G. Rossi, and N. Blefari Melazzi, "A cluster-based scalable router for information centric networks," *Computer networks*, vol. 142, 2018.

[19] G. Susi, L. Antn Toro, L. Canuet, M. E. Lpez, F. Maest, C. R. Mirasso, and E. Pereda, "A neuro-inspired system for online learning and recognition of parallel spike trains, based on spike latency, and heterosynaptic stdp," *Frontiers in Neuroscience*, vol. 12, p. 780, 2018.

[20] N. K. Kasabov, "Neucube: A spiking neural network architecture for mapping, learning and understanding of spatio-temporal brain data," *Neural Networks*, vol. 52, pp. 62–76, 2014.

[21] G. Susi, A. Cristini, and S. Mario, "Path multimodality in a feedforward snn module, using lif with latency model," *Neural Network World*, vol. 4, no. 26, 2016.

[22] S. Acciarito, G. C. Cardarilli, A. Cristini, L. D. Nunzio, R. Fazzolari, G. M. Khanal, M. Re, and G. Susi, "Hardware design of lif with latency neuron model with memristive stdp synapses," *Integr. VLSI J.*, vol. 59, no. C, pp. 81–89, Sep. 2017. [Online]. Available: https://doi.org/10.1016/j.vlsi.2017.05.006

[23] N. Kasabov, N. M. Scott, E. Tu, S. Marks, N. Sengupta, E. Capecci, M. Othman, M. G. Doborjeh, N. Murli, R. Hartono, J. I. Espinosa-Ramos, L. Zhou, F. B. Alvi, G. Wang, D. Taylor, V. Feigin, S. Gulyaev, M. Mahmoud, Z.-G. Hou, and J. Yang, "Evolving spatio-temporal data machines based on the neucube neuromorphic framework: Design methodology and selected applications," *Neural Networks*, vol. 78, pp. 1 – 14, 2016.

[24] G. Lo Sciuto, G. Susi, G. Cammarata, and G. Capizzi, "A spiking neural network-based model for anaerobic digestion process," in *2016 International Symposium on Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM)*, 2016, pp. 996–1003.

[25] S. Brusca, G. Capizzi, G. Lo Sciuto, and G. Susi, "A new design methodology to predict wind farm energy production by means of a spiking neural networkbased system," *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields*, vol. 32, no. 4, p. e2267, 2019.

[26] A. Tealab, "Time series forecasting using artificial neural networks methodologies: A systematic review," *Future Computing and Informatics Journal*, vol. 3, no. 2, pp. 334 – 340, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2314728817300715

[27] G. Capizzi, G. Lo Sciuto, P. Monforte, and C. Napoli, "Cascade feed forward neural network-based model for air pollutants evaluation of single monitoring stations in urban areas," *INTL Journal of Electronics and Communications*, vol. 61, no. 4, pp. 327–332, 2015.