Low-Cost Hardware Platform for Testing ML-Based Edge Power Grid Oscillation Detectors

Sergio A. Dorado-Rojas, Shunyao Xu, Luigi Vanfretti, Galilea Olvera
Electrical, Computer, and Systems Engineering
Rensselaer Polytechnic Institute
Troy, United States
{dorads, xus, vanfrl, olverg}@rpi.edu

M. Ilies I. Ayachi, Shehab Ahmed
Electrical and Computer Engineering
King Abdullah University of Science and Technology
Thuwal, Saudi Arabia
{mohammed.ayachi, shehab.ahmed}@kaust.edu.sa

***Abstract*—This paper introduces a low-cost hardware testing platform designed to investigate the performance of a Machine Learning (ML)-based edge application developed to detect forced oscillations in power grids. The core of the ML application lies in a Convolutional Neural Network (CNN) model deployed on an ML edge device (NVIDIA Jetson TX2). The proposed platform consists of a method for real-time signal emulation using the WaveForms Software Development Kit (SDK) that defines low-voltage signals generated by Digilent's Analog Discovery Board. The output of the signal generator is read by the Jetson board using an Analog-to-Digital Converter (ADC). Our experiments compare the performance of different ADCs when performing inference with the same CNN model. Additionally, we give an overview of the communication scheme that allows experiment automation, which is particularly useful when experiment design is time-consuming and laborious.**

***Index Terms*—Convolutional Neural Network, Data Acquisition, Experiment Automation, Forced Oscillations, Real-Time Signal Emulation**

I. Introduction

Power system oscillations can be roughly categorized into free and forced [1]. Free oscillations occur permanently in the system around a stable equilibrium point and are naturally damped out by the system. On the other hand, forced oscillations, such as interarea modes, emerge when a power system is perturbed by external disturbances that excite its modes' natural frequencies [2]. A forced oscillation may cause incipient instabilities or severe equipment damage by inducing negative impacts on the power system. In extreme cases, it may even result in system breakup, power outages, and equipment damage if not detected at the right time. Consequently, it is vital to develop a method for detecting and locating forced oscillations *on time* to reduce their negative impact [3].

A myriad of methods for detection and mitigation has been proposed to address forced oscillations in electrical grids. In [4], two non-parametric methods are presented to estimate an oscillating mode. The authors emphasize the importance of monitoring power system modes in real-time and propose a technique to determine the existence and persistence of forced oscillations. Likewise, in [5] a Phasor Measurement Unit (PMU)-based real-time sub-synchronous oscillation detection pipeline is discussed. The experiment results are quantified via simple metrics to assess the performance of the involved hardware and software systems [6].

Such techniques rely mostly upon signal processing stages to ensure proper and timely event distinction. However, the computational complexity of such methods is usually significant. The online performance of such techniques has been recently assessed. The detection method in [7], for example, takes 1.7 s to process and label a forced oscillation. Similarly, the detection time in [8] is around 350 ms. These two samples of algorithmic time execution unveil the question of whether signal processing-based solutions are also feasible for real-time deployment. In particular, it is unclear if edge devices, such as those in future information exchange schemes such as the Internet of Things (IoT) [9], would have the capabilities to execute efficiently such heavy signal processing workflows.

Machine Learning (ML) emerges as an alternative data-driven paradigm to develop solutions because ML models are computationally efficient and nowadays simple to create. Beyond forced oscillation detection, ML has proven successful for other problems such as stability assessment [10] and dynamic contingency management [11]. However, in power systems, a caveat is that measurement data describe mostly normal operating conditions. Several authors have started generating data via computer-based offline (e.g., [12]), and real-time simulations (e.g., [13]).

Either by conventional or by ML methods, the development of detection algorithms capable of real-time inference requires algorithmic efficiency *and* a suitable testing platform to generate or stream the oscillation data in real-time. A typical hardware platform for real-time experimentation is Hardware-in-the-Loop (HIL) simulation, in which the user can replicate the conditions of a particular engineering system with high accuracy. HIL has gained popularity not only in power systems but also in other domains (e.g., see [14] for an example of autonomous vehicles). In the context of oscillation detection, a HIL testbed has been used to validate the accuracy and feasibility of a fast PMU-based proposal [15].

While real-time simulation represents the most accurate way to produce training data and stream measurements during validation of real-time algorithms for oscillation detection, such simulators are not easily accessible because of their high price tag. Then, there is a need for a low-cost experimental platform to synthesize training datasets and validate detection algorithms in real-time. This paper aims to bridge this gap.

We introduce a low-cost platform (see Fig. 1) for end-to-end validation of an ML solution using a low-voltage signal generator, namely an Analog Discovery 2. Such boards are commonly used in a first electronics class in most universities. Synthetic signals are generated programmatically thanks to the WaveForms Software Development Kit (SDK). The waveform is streamed using an Analog-to-Digital Converter (ADC) to an NVIDIA Jetson TX2 device. Such specialized hardware counts with a Graphics Processing Unit (GPU), and it is capable of executing ML models in real-time. A trained Convolutional Neural Network (CNN) model is downloaded in the Jetson board and used for real-time inference: given a user-defined waveform, the CNN predicts whether a forced oscillation is occurring or not based on the available information window. A discussion of the training process of the CNN is beyond the scope of the paper. The reader is referred to [16] for more insight in this regard.

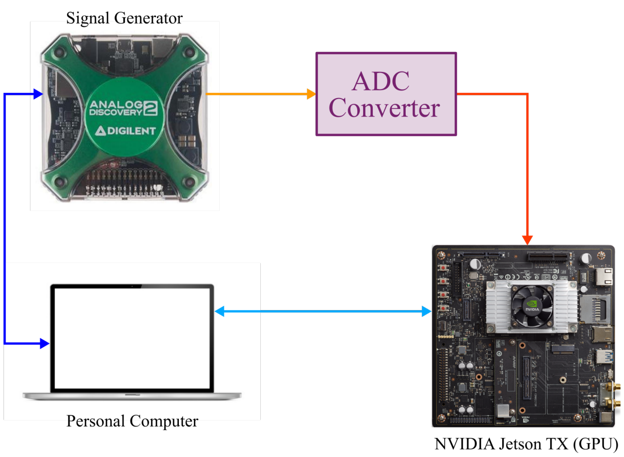

Fig. 1. Proposed Low-cost Test Platform.

In summary, the main contributions of this paper are as follows:

(i) we introduce a low-cost test framework that can be applied in the design phase for an ML-based oscillation detection algorithm before a full real-time simulator-based HIL test;
(ii) we develop a methodology for real-time signal emulation controlling an Analog Discovery 2 board using the WaveForms SDK;
(iii) we compare the inference performance when two different ADC are deployed;
(iv) finally, we present a way to automate the experiments using socket Application Programming Interface (API) in Python when the experiment process is time-consuming and laborious.

The reminder of this paper is structured as follows: Section II introduces the mechanism to generate signals using the Analog Discover 2 board. Section III presents the two different ADC methods for benchmark. Experiment automation is described in Section IV. Results are discussed in Section V. Lastly, Section VI concludes the work.

II. Real-Time Signal Emulation

It is necessary to generate a synthetic signal to emulate a sub-synchronous oscillation, similar to those observed in PMU measurements [5], so that we can assess the performance of a CNN in real-time. Such a task is possible with the Analog Discovery 2 board thanks to the WaveForms SDK. The WaveForms SDK is a public API available in programming languages such as Python and C++. It allows users to interact with the Analog Discovery 2 board and automate testing via simple applications. The scope of this section is to describe at a high level how a signal with different patterns was generated using the SDK. Then, we briefly mentioned how noise is added to the generated signal to emulate measurement and process randomness.

A. *Characteristics of Forced Oscillation Waveforms*

In a steady-state, a power system dynamical state $\mathbf{x}$ is said to operate at a stable equilibrium condition $\mathbf{x}_{\text{eq}}$ when $\dot{\mathbf{x}}|_{\mathbf{x}=\mathbf{x}_{\text{eq}}} = f(\mathbf{x}_{\text{eq}}) = \mathbf{0}$. The continuous stochastic nature of loads makes the state $\mathbf{x}$ oscillate around $\mathbf{x}_{\text{eq}}$. In practice, observables of power grids (e.g. measurements such as current and voltage phasors) show "small-amplitude oscillations" around an equilibrium condition. A forced oscillation is characterized by an abrupt change in the amplitude of a signal in a power grid lasting a specific time. If the oscillation is stable, the power system should return to equilibrium after the event fades out. However, stable forced oscillations could lead to cascading events that can lead to a massive event such as a blackout. Detecting forced oscillations is critical to take remedial actions that guarantee a reliable operation of any electrical system [17].

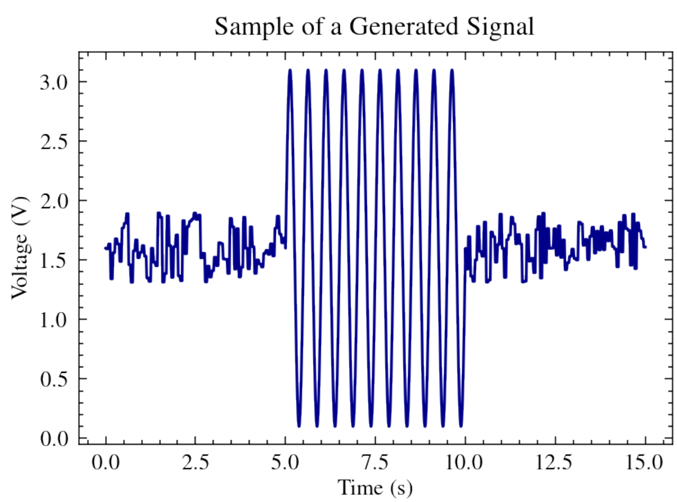

Fig. 2. Example waveform describing a forced oscillation. This signal was produced with the Analog Discovery 2 board.

Assume an event leading to a forced oscillation has occurred in the grid. Before the contingency, measurements will show a stationary behavior characterized by the excursions of the system state around $\mathbf{x}_{eq}$. During the event, the amplitude of the signal will change significantly. The system will return to equilibrium if the proper remedial actions are taken, e.g. ramp-down of a wind farm's power ouput [5]. An example waveform characterizing a conceptual forced oscillation event is shown in Fig. 2.

B. Generation of Signals using the WaveForms SDK

Based on the previous discussion, the most simple signal that characterizes a forced oscillation consists of three parts: an oscillating behavior around a steady-state value; a high-amplitude oscillating waveform during the event; a final noisy-like segment that describes the return to the equilibrium condition. Such signal is generated straightforwardly using the WaveForms Python API. The API allows the user to modify each part of the signal by varying frequency, amplitude, and offset parameters for different shapes. The particular form in Fig. 2 was generated by a random signal (sampled from a uniform distribution) with a pre-specified 1.5 V offset. After 5 s, the output of the signal generator is changed to a sinusoidal signal to mimic a forced oscillation event lasting for 5 s. Lastly, the output is switched again to a noisy signal. Several test signals with different characteristics can be easily generated by amplitude and/or frequency sweeps by following this simple workflow. To increase the "complexity" of the generated waveform, noise is superimposed also on the oscillation part as shown in Fig. 3.

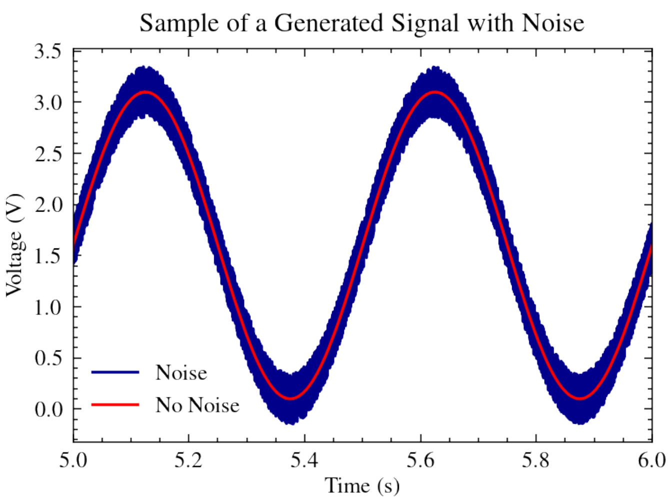

Fig. 3. Example waveform with noise superimposed on the forced oscillation sinusoid.

III. Data Acquisition Methods

After generating a signal waveform for testing real-time inference in an ML-based oscillation detector, the next step is to send the signal to the NVIDIA Jetson TX2. This requires a data acquisition or conversion stage. The core of this step is the ADC. ADCs are a mature technology whose main advantages are fast conversion and low cost, so their use in the context of the proposed platform is justified.

Following the ADC conversion, the data is transmitted to the Jetson via I^2C. Therefore, in this section we explain how the ADC and the I^2C communication stages are implemented. For benchmarking purposes, two different ADCs were tested, namely an 8-bit ADC (PCF8591) and a 12-bit ADC (ADS7823). A comparison between the inference performance of both ADCs is presented at the end of this section.

A. General Aspects of ADC Conversion

Roughly speaking, an ADC takes the analog signal at its input and produces a value by determining how far the input voltage is between the low and high reference voltages. The more discrete levels an ADC has, the more accurate the digital representation of the analog signal is. An ADC with a larger number of "levels" (i.e., bits) will provide better accuracy and larger resolution than an ADC with fewer bits.

The first ADC employed in the platform is 8-bit ADC (PCF8591). The corresponding circuit schematic is shown in Fig. 4. The analog signal is output from the Analog Discovery 2 board using channel 1, connected to the analog input AIN0 of the PCF8591. The SDA and the SCL pins of PCF8591 are connected to the Jetson board's corresponding pins (I^2C bus 0). By doing so, data can be received in the NVIDIA device from the ADC via I^2C. Note that the grounds of all devices (i.e., Jetson TX2, Analog Discovery 2, and PCF8591) are connected to guarantee the same low reference voltage.

An example of inference using the PCF8591 is presented in Fig. 5. The CNN can detect whether an oscillation is occurring or not with relatively good accuracy[1] (namely, 99.86% in average).

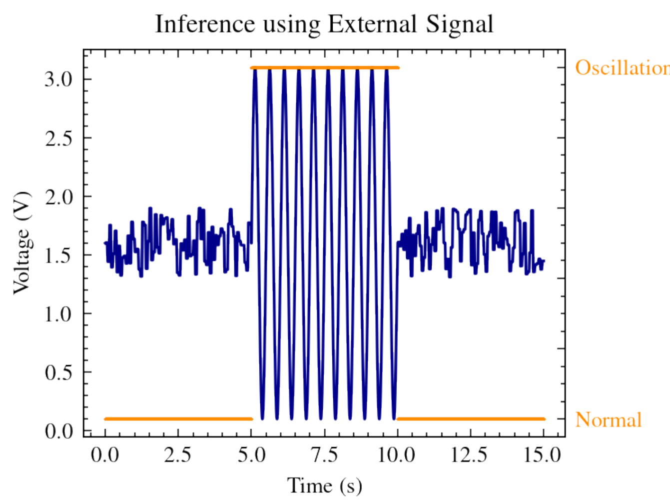

Fig. 5. Inference Result for Implementing PCF8591 in Data Acquisition.

The second ADC variant used in the platform project is a 12-bit ADC: an ADS7823. Logically, the ADS7823 has a better resolution and a higher accuracy than the PCF8591.

[1]It is worth mentioning that physical forced oscillation signals do not have a clear pattern as the sinusoid signal in Fig. 5. The accuracy of the CNN downgrades when facing real measurements. However, the method maintains a remarkable inference performance using short data windows and reduced computational time. For further discussion, the reader is referred to [16].

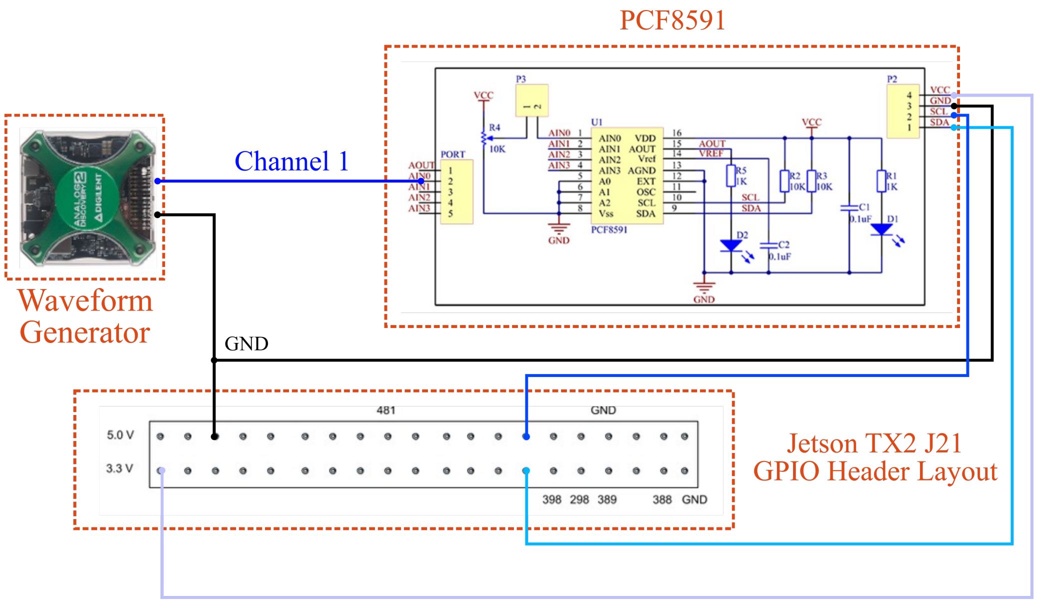

Fig. 4. Circuit Schematic using the PCF8591 ADC.

The circuit schematic for the ADS7823 is similar to the one in Fig. 4 and, therefore, will be omitted. The ADS7823 also uses I^2C communication to transmit the signal to the NVIDIA Jetson TX2 board.

B. A Simple ADC Comparison

We assessed the effect of the ADC resolution by performing 10,000 inferences with the trained CNN model. Results are presented in Table I, where average accuracy is obtained after averaging the accuracy of all inferences over 10,000 experiments. The ADS7823 achieves better performance since the converted signal has a better resolution, and it is easier for the CNN to identify the oscillation patterns and classify the oscillation condition correctly. A more detailed comparison between both ADCs can be seen in Section V.

TABLE I
INFERENCE ACCURACY USING DIFFERENT ADCS

ADC	Average Accuracy
ADS7823	0.9986
PCF8591	0.9513

IV. EXPERIMENT AUTOMATION

The previous experimental conditions are insufficient to draw significant conclusions about the effect of the ADC selection on inference. By varying other parameters of the test signal, such as frequency and noise level variation, different scenarios can be crafted where we can extract more insight concerning the actual performance differences.

Performing a relevant parameter sweep *manually* is a time-consuming effort. However, one of the advantages of the WaveForms SDK is the possibility of automating several tests using the Python API. This section discusses setting up a simple automation approach by establishing a communications network between the signal generator, the host computer, and the NVIDIA device.

The setup is shown in Fig. 6. The communication between the host computer and the Jetson TX2 takes place through TCP/IP socket API. The laptop is configured as the client and the Jetson TX2 as the server. The WaveForms SDK, running on the computer, controls the Analog Discovery 2 board, connected via USB. The output of the signal generator passes through the ADC converter and is read by the TX2 board using I^2C.

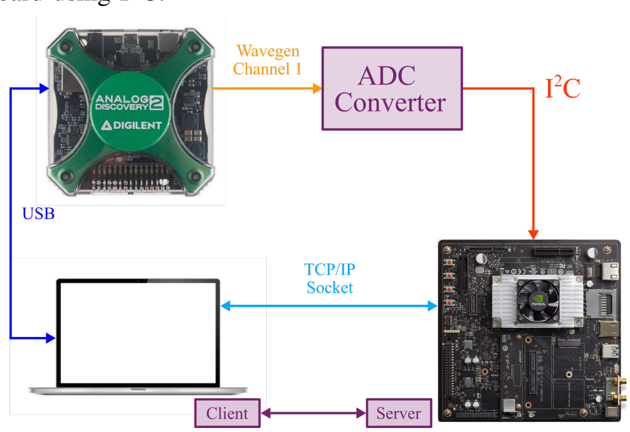

Fig. 6. Communication Map in the Proposed Experimental Setup.

The importance of automation for both testing and communication is illustrated with a simple example. Consider an experiment that consists of a frequency sweep over 100 values. Such variation can be easily created using a loop in the WaveForms SDK. Then, the Analog board outputs the signal to the ADC. However, the host computer must trigger the Jetson TX2 to start acquiring data. Moreover, the following experiment should begin only once the Jetson TX2 completes

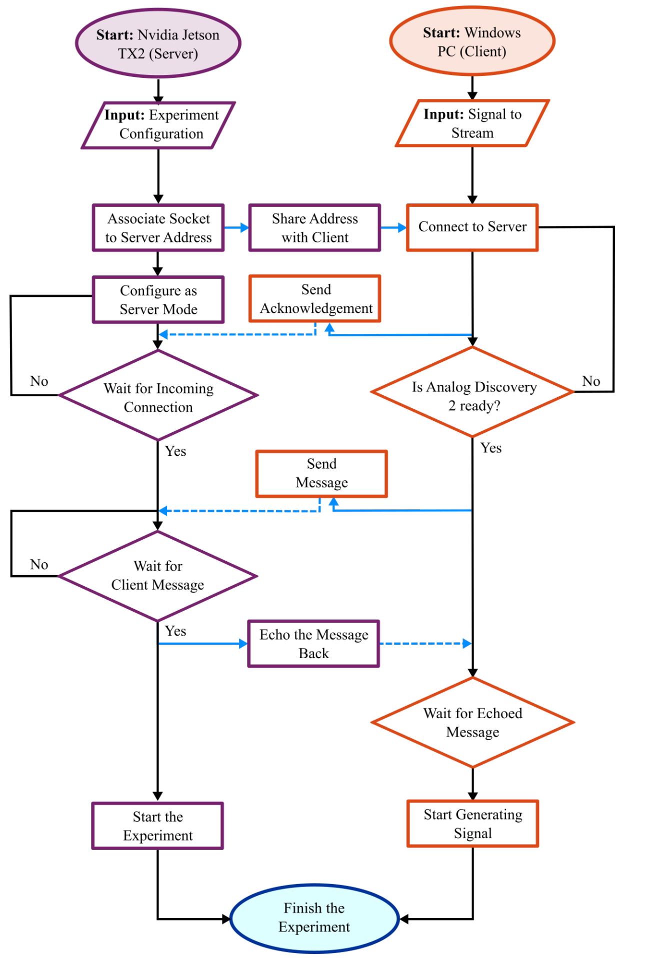

Fig. 7. Experiment Automation Workflow. The Solid Blue Lines Indicate Information Sent Between Server and Client, & Dashed Lines Indicate (Uncertain) Receipt of Message.

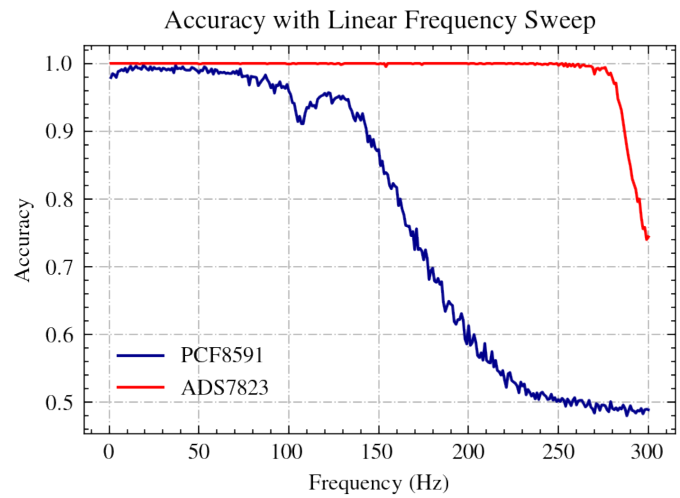

Fig. 8. Frequency Variation Experiment Result.

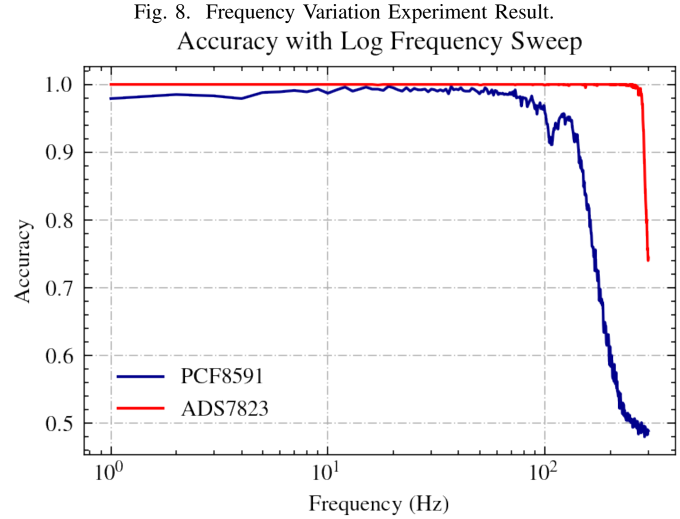

Fig. 9. Frequency Variation Experiment Results (Semilog).

inference with the current signal. To achieve this, the laptop and the Jetson are set in a two-way communication link over TCP/IP. The host computer is the client, and the Jetson is the server. The complete workflow is presented in Fig. 7.

V. Results

A frequency variation experiment is now described after portraying the communication map and the hardware connection. We perform 300 experiments varying the frequency from 1 to 300 Hz, keeping constant the sinusoidal amplitude during the oscillation event. CNN inference is performed on 1000 windows of the produced signal. Half of the windows correspond to normal conditions, and the other half is a sustained oscillation. After the TX2 computes 1000 inferences, a message is sent to the client (host computer) to start the next experiment by varying the signal frequency.

The results of the frequency variation experiment are shown in Fig. 8. The x-axis indicates the frequency value, and the y-axis represents the average accuracy over the 1000 inferences during an experiment. We observe that for low frequencies, the performance of both ADCs is close to each other. Nevertheless, as the frequency increases, the effectiveness of the PCF8591 ADC downgrades rapidly. Despite this, both ADCs are found to be effective in the context of subsynchronous oscillations, where oscillations of concern are below 60 Hz.

The previous graph is replotted in a logarithmic scale in Fig. 9, which is equivalent to having carried out a logarithmic sweep. We aim to identify the critical frequency in which the performance of the PCF8591 downgrades. We observe that the 8-bit ADC is not effective after ≈ 100 Hz. However, even the ADS7823 exhibits a significant reduction in accuracy when the frequency exceeds ≈ 200 Hz. Recall that both experiments are using the same CNN model. Therefore, our experiments emphasize the importance of appropriate hardware selection in *every* stage when an ML solution is deployed on a practical application.

Furthermore, we plot the inference results for all the time windows in a histogram (Fig. 10). The x-axis corresponds to the accuracy levels and the y-axis to how many times the CNN achieved the corresponding accuracy. Note that both histograms are skewed towards the 1.0 accuracy, meaning that both ADCs are effective at detecting the oscillation in the particular experiment. However, the PCF8591 bins are distributed horizontally, indicating the accuracy is more sensitive to the resolution of the ADC. In general, we can conclude that the higher the resolution of the ADC, the easier it is for the CNN to detect the oscillation patterns and classify the oscillation condition correctly. The results in Table II quantify these observations. The reader is referred to [16] for a performance analysis of the CNN when subjected to real-world data.

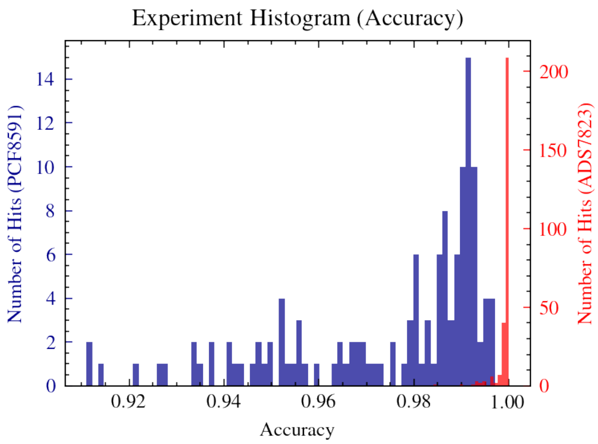

Fig. 10. Accuracy Histogram.

TABLE II
STATISTICAL RESULTS OF THE FREQUENCY VARIATION EXPERIMENTS

ADC	Average Accuracy	
	Mean	**Standard Deviation**
PCF8591	0.97498	0.0213
ADS7823	0.99940	0.00170

VI. CONCLUSIONS

This paper has presented a low-cost platform for evaluating the real-time performance of CNN models in the detection of sub-synchronous forced oscillation in power grids. Potential applications of such a platform beyond teaching and demonstration are also related to the cheap and fast prototyping of ML-based edge embedded solutions with real-time constraints, such as protective relays.

In our setup, a synthetic signal was generated using a generator (Analog Discovery 2), employed throughout several introductory electronics courses. The signal is converted to a digital representation thanks to an ADC and then transmitted to an NVIDIA Jetson TX2 device. This ML-edge device is capable of executing in real-time a pre-trained CNN model. Thus, automation on the signal generation and the communication between the Jetson device and the host computer allows simple testing and validation of the accuracy of the proposed solution for oscillation detection.

We also validated the performance impact of ADC selection through simple experiments. Better accuracy is achieved with a high-resolution ADC for the same CNN model. This aspect is not regularly considered in ML training. Still, it is crucial when the CNN is deployed on hardware as part of an IoT solution for monitoring and diagnostics in cyber-physical systems such as the power grid.

REFERENCES

[1] M. Ghorbaniparvar, "Survey on forced oscillations in power system," *Journal of Modern Power Systems and Clean Energy*, vol. 5, no. 5, pp. 671–682, Sep. 2017.

[2] M. A. Magdy and F. Coowar, "Frequency domain analysis of power system forced oscillations," *IEE Proceedings C (Generation, Transmission and Distribution)*, vol. 137, no. 4, pp. 261–268, Jul. 1990.

[3] G. Rogers, *Power System Oscillations*. Springer Science & Business Media, Dec. 2012.

[4] L. Vanfretti, S. Bengtsson, V. S. Perić, and J. O. Gjerde, "Spectral estimation of low-frequency oscillations in the nordic grid using ambient synchrophasor data under the presence of forced oscillations," in *2013 IEEE Grenoble Conference*, Jun. 2013, pp. 1–6.

[5] L. Vanfretti, M. Baudette, J. L. Domínguez-Garcíaz, A. White, M. S. Almas, and J. O. Gjerdeóy, "A PMU-based fast real-time sub-synchronous oscillation detection application," in *2015 IEEE 15th International Conference on Environment and Electrical Engineering (EEEIC)*. ieeexplore.ieee.org, Jun. 2015, pp. 1892–1897.

[6] J. L. Domínguez-García, M. Baudette, G. Del-Rosario, A. Ruíz-Alvárez, M. S. Almas, I. Cairó, and L. Vanfretti, "Validation experiment design of a PMU-based application for detection of sub-synchronous oscillations," in *2015 IEEE 15th International Conference on Environment and Electrical Engineering (EEEIC)*, Jun. 2015, pp. 1898–1903.

[7] H. Cho, S. Oh, S. Nam, and B. Lee, "Non-linear dynamics based sub-synchronous resonance index by using power system measurement data," *IET Gener. Transm. Distrib.*, vol. 12, no. 17, pp. 4026–4033, 2018.

[8] H. Khalilinia and V. Venkatasubramanian, "Subsynchronous resonance monitoring using ambient high speed sensor data," *IEEE Trans. Power Syst.*, vol. 31, no. 2, pp. 1073–1083, Mar. 2016.

[9] A. Ghasempour, "Internet of things in smart grid: Architecture, applications, services, key technologies, and challenges," *Inventions*, vol. 4, no. 1, p. 22, Mar. 2019.

[10] S. A. Dorado-Rojas, T. Bogodorova, and L. Vanfretti, "Time Series-Based Small-Signal Stability Assessment using Deep Learning," in *2021 North American Power Symposium (NAPS)*, Nov. 2021, pp. 1–6.

[11] A. Ghasemkhani, A. Darvishi, I. Niazazari, A. Darvishi, H. Livani, and L. Yang, "DeepGrid: Robust deep reinforcement learning-based contingency management," in *2020 IEEE Power Energy Society Innovative Smart Grid Technologies Conference (ISGT)*, Feb. 2020, pp. 1–5.

[12] S. A. Dorado-Rojas, M. de Castro Fernandes, and L. Vanfretti, "Synthetic Training Data Generation for ML-based Small-Signal Stability Assessment," in *2020 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, Nov. 2020, pp. 1–7.

[13] M. S. Almas, L. Vanfretti, and L. Vanfretti, "Experimental performance assessment of a generator's excitation control system using real-time hardware-in-the-loop simulation," 2014.

[14] Ş. Y. Gelbal, S. Tamilarasan, M. R. Cantaş, L. Güvenç, and B. Aksun-Güvenç, "A connected and autonomous vehicle hardware-in-the-loop simulator for developing automated driving algorithms," in *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Oct. 2017, pp. 3397–3402.

[15] L. Vanfretti, M. Baudette, I. Al-Khatib, M. S. Almas, and J. O. Gjerde, "Testing and validation of a fast real-time oscillation detection PMU-based application for wind-farm monitoring," in *2013 First International Black Sea Conference on Communications and Networking (BlackSeaCom)*, Jul. 2013, pp. 216–221.

[16] S. A. Dorado-Rojas, S. Xu, L. Vanfretti, M. I. I. Ayachi, and S. Ahmed, "ML-Based Edge Application for Detection of Forced Oscillations in Power Grids," in *IEEE Power & Energy Society General Meeting*, 2022.

[17] A. Ghasempour and J. Lou, "Advanced metering infrastructure in smart grid: Requirements, challenges, architectures, technologies, and optimizations," in *Smart Grids: Emerging Technologies, Challenges and Future Directions*. Nova Science Publishers Hauppauge, NY, USA, 2017, pp. 1–8.