2019

Initial Implementation of a Machine Learning System for SRF Cavity Fault Classification at CEBAF

A. Carpenter
Jefferson Lab

K. M. Iftekharuddin
Old Dominion University, kiftekha@odu.edu

T. Powers
Jefferson Lab

Y. Roblin
Jefferson Lab

A. Solopava Shabalina
Jefferson Lab

Follow this and additional works at: https://digitalcommons.odu.edu/ece_fac_pubs

Original Publication Citation

Carpenter, A., Iftekharuddin, K. M., Powers, T., Roblin, Y., Solopova Shabalina, A. D., Tennant, C., & Vidyaratne, L. (2019). Initial implementation of a machine learning system for SRF cavity fault classification at CEBAF. In K. White, K. Brown, P. Dyer, & V. R. W. Schaa (Eds.), *Proceedings of the 17th International Conference on Accelerator and Large Experimental Physics Control Systems*. JACoW Publishing. https://doi.org/10.18429/JACoW-ICALEPCS2019-WEPHA025

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA025

INITIAL IMPLEMENTATION OF A MACHINE LEARNING SYSTEM FOR SRF CAVITY FAULT CLASSIFICATION AT CEBAF

A. Carpenter†, T. Powers, Y. Roblin, A. Solopova, C. Tennant
Jefferson Lab, Newport News, USA
L.Vidyaratne, K. Iftekharuddin, Old Dominion University, Norfolk, USA

Abstract

The Continuous Electron Beam Accelerator Facility (CEBAF) at Jefferson Laboratory is a high power Continuous Wave (CW) electron accelerator. It uses a mixture of of SRF cryomodules: older, lower energy C20/C50 modules and newer, higher energy C100 modules. The cryomodules are arrayed in two anti-parallel linear accelerators. Accurately classifying the type of cavity faults is essential to maintaining and improving accelerator performance. Each C100 cryomodule contains eight 7-cell cavities. When a cavity fault occurs within a cryomodule, all eight cavities generate 17 waveforms each containing 8192 points. This data is exported from the control system and saved for review. Analysis of these waveforms is time intensive and requires a subject matter expert (SME). SMEs examine the data from each event and label it according to one of several known cavity fault types. Multiple machine learning models have been developed on this labeled dataset with sufficient performance to warrant the creation of a limited machine learning software system for use by accelerator operations staff. This paper discusses the transition from model development to implementation of a prototype system.

INTRODUCTION

Jefferson Lab's Continuous Electron Beam Accelerator Facility (CEBAF) is a high power Continuous Wave (CW electron accelerator. It utilizes two styles of SRF modules, older, lower gradient C20/C50 modules and newer higher, gradient C100 modules. In 2013, the upgrade from 6 to 12 GeV was completed. This upgrade included the installation of 11 C100-style 100 MV cryomodules and associated RF systems (see Fig. 1) [1]. RF faults are routinely the largest contributor to lost beam time. Since the primary mechanism for reducing the occurrence of RF faults is to reduce cavity gradient, RF faults also adversely impact CEBAF's energy reach. Accurately identifying which cavity faulted allows operators to lower the gradient only on targeted cavities rather than the entire module. However, this process is complicated due to strong mechanical coupling between cavities within a C100 cryomodule whereby a fault in one cavity may precipitate faults in other nearby cavities [2]. Additionally, identifying the cause of the fault may allow engineering staff to determine remediation methods in place of gradient reduction. An effort has been underway to produce an automated machine learning system capable of identifying the location and cause of RF faults within the C100 cryomodules.

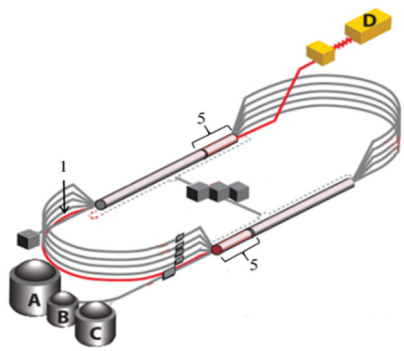

Figure 1: Schematic of the CEBAF accelerator showing the locations of the 11 C100 cryomodules for which RF fault data is recorded and analyzed.

MACHINE LEARNING SYSTEM OVERVIEW

Our machine learning system is comprised of four components: data generation, data storage, the machine learning model and its analysis, and the presentation of results. Creation of this system has required considerable effort related to the development of each component. The low-level RF hardware and EPICS control system was modified to generate synchronized diagnostic waveform data for each cryomodule. Waveform harvester software was written to collect and store the waveform data for each RF fault. A system expert then manually analyzed the stored waveform data to identify and label the cause and cavity location of the fault. This labeled dataset allows for the use of supervised machine learning techniques in developing a set of models for identification of fault location and cause. We turned the models into a software product capable of analyzing waveform data from a fault in real-time. Finally, creating additional software for displaying and aggregating the results of this analysis for CEBAF operations and engineering staff is an outstanding task we plan to tackle in the future.

Each component of the system acts as the foundation for the next, making it difficult to simply "write some software" that perform the desired analysis. Fortunately, each component provides immediate benefit making each incremental development worthwhile in its own right. For example, capturing and storing diagnostic information allows RF system experts to gain insight into the cause and location of individual faults. Accumulating and analyzing a large body of data allows operations and maintenance staff

* Work supported by Jefferson Science Associates, LLC under U.S. DOE Contract № DE-AC05-06OR23177
† adamc@jlab.org

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA025

to be trained to identify common failure modes, reducing the reliance on system experts for routine operations. Producing a machine learning model and related software reduces the training requirements and workload on operations and maintenance staff and allows for more accurate and timely corrective actions in response to faults. The following sections will discuss each component in more detail.

DATA GENERATION AND STORAGE

Each of the eight cavities within a C100 cryomodule has an independent Field Control Chassis (FCC) EPICS Input/Output Controller (IOC), capable of buffering and presenting 17 diagnostic waveforms. These waveforms are stored in buffers containing 8192 points and sampled at a configurable rate typically ranging from 5 kHz to 20 kHz. Upon experiencing an RF fault, these buffers are frozen and made available in an EPICS waveform and neighboring IOCs are rapidly notified of the fault. This allows for a time-synchronized set of waveforms to be produced across all cavities within the cryomodule. In addition to the raw waveform data, the IOCs also present PVs for the timestamp associated with the fault, the sampling interval, and the relative offset of fault from the start of the buffered waveform.

A `harvester` daemon process monitors the status of each FCC IOC waiting for the IOCs to indicate that a fault has occurred and they are in a "capture ready" state (see Fig. 2). The `harvester` generates data files corresponding to the waveforms from each FCC IOC, saving them to long-term storage. FCC IOCs from the same cryomodule that present waveforms within a specified three second time window are grouped together to create a single fault event. Approximately 19500 RF faults have been captured from spring 2018 to summer 2019; however, many of these are not of operational interest. As an optional, final step, the `harvester` daemon may run an executable to perform any desired post-capture tasks.

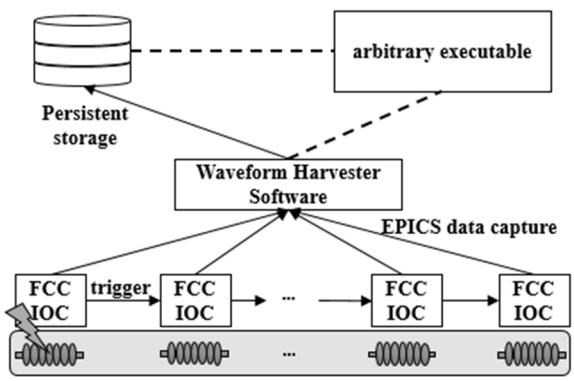

Figure 2: A conceptual diagram for the data generation and storage systems. A fault at a cavity triggers data collection at that cavity's FCC IOC, and the fault trigger is propagated to neighboring FCC IOCs associated with the cryomodule.

While this system for data acquisition is conceptually simple, it is still important to understand the various ways in which data acquisition can go awry. Original implementations gave each IOC a single buffer for each waveform and a toggle that allowed the buffer to be used for fault trigger data acquisition or manual investigation. This meant that external checks were needed to ensure the proper configuration for data collection. An additional source of error is that each FCC IOC enters into a capture ready state independently with no guarantee that all FCC IOCs relating to the cryomodule will follow suit. This process occasionally encounters errors due to failures in the notification process or because the time between the first and later IOCs signaling capture-ready states is longer than the specified event time window. While future work is being discussed to address these issues, downstream components of a machine learning system must be aware of these failure modes in order to validate the generated data.

DATA ANALYSIS AND EVENT LABELING

Once a body of data is generated, the data is analyzed offline by SRF subject matter experts in a time consuming process. This produces hundreds of labeled fault examples for later model development using supervised machine learning techniques. The expert analysis consists of three main functions – filtering out bad or irrelevant data, identifying new fault causes and locations (labels), and categorizing individual examples with previously identified labels.

The filtering process used by the system expert is important to understand as it affects the dataset used to train the downstream machine learning model. While the system expert removes faulty data, they also filter out faults that are essentially irrelevant to operations. Only faults that occurred during "steady state" beam operations are considered relevant to reducing events that trip off beam. Other faults typically occurred during the recovery from a previous fault and are beyond the scope of this effort.

The system expert must identify a set of common fault modes and subsequently label examples according to which cavity experienced the fault and the underlying cause of the fault. Our current labeled dataset includes labels for which individual cavity faulted (or that multiple faulted) and labels for four single cavity fault types [2, 3]. These labeled examples form the foundation of supervised machine learning approaches. Over time, system experts may identify additional, previously unknown, fault causes. It is important that the downstream machine learning software applications allow for this natural evolution.

MODEL DEVELOPMENT

A well-trained model can be used to streamline the time consuming process of labeling individual faults and generating accurate labels in real-time. Data scientists, or analysts, develop the machine learning models in another time consuming process. The initial labeled dataset contained several hundred examples from the end of CEBAF's spring 2018 physics run.

Analysts conducted an initial round of model development using traditional machine learning models [4]. Deep learning techniques were marked for future consideration,

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA025

as we believed the labeled dataset was of insufficient size. Traditional machine learning models are not well suited to raw time series data and require an additional step of feature engineering via statistical techniques. This approach required the use of the Python package `tsfresh` to perform computationally intensive feature engineering [4]. Initial investigations required several hours per event to calculate the full range of statistical information available from `tsfresh`. However, later optimizations and a reduced feature extraction process lowered this time to the order of seconds. We evaluated a range of traditional machine learning models, and a random forest model performed with sufficient accuracy to warrant the creation of analytical software for operations [5].

Analysts pursued a second round of model development using deep learning techniques, specifically a long short-term memory (LSTM) neural network model. This approach operates on the unprocessed waveform data reducing computation time at both the training and prediction steps. The authors found that despite limited data and development effort, we could produce a model with reasonable accuracy. This demonstrates the potential for deep learning techniques for RF fault classification. Deep learning models are currently under development and yield competitive results [6]. See Tables 1 and 2 for a summary of the most recent accuracy results.

Table 1: Accuracy of Several Machine Learning Models Applied to the Fault Location Problem. (*Deep Learning)

Method	Accuracy (%)
Random Forest	95.7
Gradient Boosting Classifier	95.2
Extra Trees	94.1
Bagging Classifier	94.1
Bi-directional LSTM *	94.1
Support Vector Classification	90.9
k-Nearest Neighbor	88.8
Decision Tree	87.7
Gaussian Naïve Bayes	86.1

Table 2: Results of Several Machine Learning Models Applied to the Fault Type Problem (*Deep Learning)

Method	Accuracy (%)
Decision Tree	97.6
Random Forest	88.0
RNN-LSTM *	86.0

SOFTWARE APPLICATIONS FOR ONLINE DATA ANALYSIS

Given machine learning models capable of accurately identifying fault location and cause, the next step in developing a machine learning system is to create a software application capable of leveraging the model to analyze data in real-time. This application must be triggered after a fault and data capture has occurred, be able to use one of the multiple models that have been developed, save the results of analysis for later review, and allow for integration with visualization and reporting tools.

One of the primary challenges of this software was how to implement the ability to load different machine learning models. We expect to generate new models for one of three main reasons: improving model performance, updating the set of labels, and retraining models to match changes in the diagnostic waveform data. One could naively imagine trying to directly load different binary model files exported from a software package like SciKit-Learn's `sklearn` or Google's `tensorflow` [7, 8]. However, this presents a number of potential problems ranging from managing package dependency conflicts to needing to support multiple machine learning frameworks within a single application. The chosen solution is to design the application in a modular fashion with a main Python script and backend pluggable model applications each with their own Python virtual environment. The main Python script provides a full featured command line interface including listing model information and analyzing events. The backend pluggable models supply all of the analytical routines and data validation needed for their particular model implementation.

Developing maintainable software that produces accurate results requires an understanding of the various supporting components. For example, knowledge of the system expert's filtering process and of the failure modes of the data collection process dictates the programmatic requirements for data validation. Proper data validation improves the accuracy of the predictions as the model is presented with data similar to the training set. Recognizing in advance that new models, requiring possibly incompatible software stacks, will continue to be developed and that old models will likely be retrained dictates a modular design that is more easily maintained.

The online classification software, `rf_classifier`, was developed to meet these needs [9]. The application consists of a Python command line application for analyzing a fault event and pluggable models applications that perform the actual analysis. This main application has the capability to list information about available analysis models and to specify which model is used we can use in the analysis. Each model application follows a standard design that defines a common interface with the main application and requires model applications contain their own documentation and test suites. Each pluggable model is required to approximate the data validation process used by the system expert labeling process and utilize the underlying model to analyze supplied data. The results from analyzing a fault event must include identifying information about the fault event, the predicted cavity label, the predicted fault type label, and the confidence associated with each label, where the confidence is a number between [0,1] produced by the underlying model or null if such a value is inappropriate.

Currently only one pluggable model has been developed. For a supplied fault event, this model validates that: all of the cavities within a cryomodule are either bypassed or op-

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA025

erating in a generator driven resonance mode, all of the required waveforms are present in the data, a capture file is present for each FCC IOC for the cryomodule, and all of the waveforms across all cavities were sampled at the same rate. The initial model identifies which individual cavity faulted or that multiple cavities faulted and which of the following fault types occurred - Single Cavity Turn Off, Multiple Cavity Turn Off, Microphonics, Quench or E_Quench [2]. The application's simple command line interface allows us to call it programmatically or interactively. The general intent is for the `harvester` post-capture script to call `rf_classifier` after data has been captured and to store the results in an existing waveform database described below. This software is under review and will soon be released for use in CEBAF's operational controls environment. Source code and documentation for the main application and models are available at **https://github.com/JeffersonLab/rf_classifer.**

OPERATIONS TOOLS FOR VISUALIZATION AND REPORTING

Generating reams of data is only useful if it is made available for consumption. Currently operations staff use a web-based application, `wfbrowser`, for viewing both a timeline of C100 cryomodules fault events and the waveforms associated with individual fault events (see Fig. 3) [10]. The `wfbrowser` application relies on a backend database containing information on all fault events that have been harvested. This database is updated in real-time via a harvester daemon post-capture script. Integrating the results of `rf_classifier` into this existing pipeline is straightforward and currently under development. This update requires that the `wfbrowser` backend database and server-side software be updated to support labeling events, that the `wfbrowser` client UI include updates to display the labeling information from `rf_classifier` and that the `harvester` post-capture script be updated to include a call to `rf_classifier`. Once we have integrated the data into the existing database, generating any reports needed by operations or maintenance staff will become a much simpler process.

FUTURE WORK

While the initial implementation of a system for on-line classification of C100 RF faults is nearing production, opportunities exist for improvement along a number of paths. At a basic level, we have identified a number of failures from the data generation and storage systems that should be corrected or mitigated, and additional fault data has been produced that has yet to be fully analyzed and labeled.

Reporting and visualization tools are currently in an evolving state. Updates to `wfbrowser` are in development to provide operations staff with access to basic visualizations and summary reporting. Future software enhancements include providing remediation guidance to operators. In this case, system experts would define remediation steps that the software would recommend when seeing repeated faults at a given location.

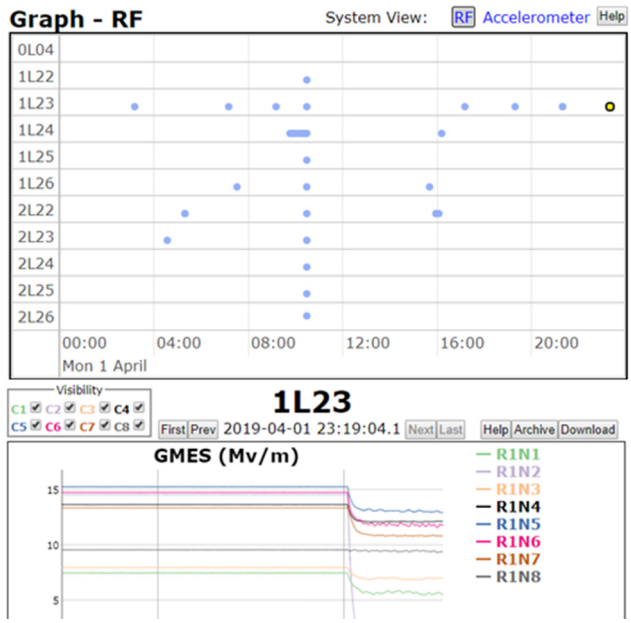

Figure 3: Screenshots from `wfbrowser`'s fault event timeline (top) and waveform (bottom) interfaces.

Work on developing a deep learning model is underway with encouraging preliminary results. Moving to a deep learning model would significantly reduce the amount of computational time spent at prediction time, as we would no longer need to extract features via `tsfresh`. Finally, all operations software tools are to be deployed in preparation for the fall 2019 physics run.

CONCLUSION

The initial deployment of a machine learning system for classifying the location and fault type within CEBAF's new C100 style cryomodules is well underway. The system is able to capture and analyze, in real-time, fault data from all C100 cryomodules. It utilizes models capable of identifying which cavity faulted (95.7% accuracy) and which type of fault occurred (97.6% accuracy) using the data analyzed in this work. This system will aid in improving CEBAF's availability and energy reach by helping operations staff better target gradient reductions to faulting cavities and by better informing maintenance operations of primary fault causes.

ACKNOWLEDGMENTS

Special thanks to Ramachrishna Bachimanchi, George Lahti, and Chris Slominski at Jefferson Lab for their work on the data generation and storage components.

REFERENCES

[1] M. F. Spata, "12 GeV CEBAF Initial Operational Experience and Challenges", in *Proc. 9th Int. Particle Accelerator Conf.* (IPAC'18), Vancouver, Canada, Apr.-May 2018, pp. 1771- 1775. doi:10.18429/JACoW-IPAC2018-WEYGBD1

[2] T. Powers, A. Solopova, "CEBAF C100 Fault Classification Based On Time Domain RF Signals", in *Proc. 19th Int. Conf. on RF Superconductivity*, Dresden, Germany, June-July 2019.

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA025

[3] A. D. Solopova *et al.*, "SRF Cavity Fault Classification Using Machine Learning at CEBAF", in *Proc. 10th Int. Particle Accelerator Conf. (IPAC'19)*, Melbourne, Australia, May 2019, pp. 1167-1170. doi:10.18429/JACoW-IPAC2019-TUXXPLM2
[4] tsfresh, https://tsfresh.readthedocs.io/en/latest/
[5] C. Tennant *et al.*, "Progress on Using Machine Learning for SRF Cavity Fault Classification", JLAB-TN-19-018.
[6] A. Solopova *et al.*, "SRF Cavity Fault Classification Using Machine Learning at CEBAF", *Advanced Control Methods for Particle Accelerators Workshop*, Santa Fe, NM, 2019.
[7] scikit-learn, https://scikit-learn.org/stable/
[8] TensorFlow, https://www.tensorflow.org
[9] rf_classifier, https://github.com/JeffersonLab/rf_classifier
[10] wfbrowser, https://github.com/JeffersonLab/wfBrowser