


Time series forecasting methods and their applications to particle accelerators

Sichen Li^{*} and Andreas Adelman[†]

Paul Scherrer Institute, 5232 Villigen Switzerland

 (Received 24 September 2022; accepted 5 December 2022; published 15 February 2023)

Particle accelerators are complex facilities that produce large amounts of structured data and have clear optimization goals as well as precisely defined control requirements. As such they are naturally amenable to data-driven research methodologies. The data from sensors and monitors inside the accelerator form multivariate time series. With fast preemptive approaches being highly preferred in accelerator control and diagnostics, the application of data-driven time series forecasting methods is particularly promising. This review formulates the time series forecasting problem and summarizes existing models with applications in various scientific areas. Several current and future attempts in the field of particle accelerators are introduced. The application of time series forecasting to particle accelerators has shown encouraging results and promise for broader use, and existing problems such as data consistency and compatibility have started to be addressed.

DOI: 10.1103/PhysRevAccelBeams.26.024801

I. INTRODUCTION

Particle accelerators have a significant role in various areas of science, from searches for new physics and nuclear-waste transmutation to cancer treatment. They are also facilities that lend themselves to data-driven research methodologies, such as event forecasting based on machine learning (ML), given that they produce substantial volumes of structured data and that their operation is defined by clear optimization goals and precise control requirements. To achieve optimal operational conditions while keeping the accelerator under control and within safety limits, a multitude of different sensors and monitors are placed at specific positions inside the accelerator complex. The data are recorded as multivariate time series, sampled at specified frequencies. The future values of some quantities of interest, or a potential failure of the machine, might then be inferred by time series forecasting methods.

A *time series* $\vec{x}_t \in \mathbb{R}^n$, where t stands for time and n is the dimension of the desired variables, is “a collection of observations made sequentially through time” [1]. The *forecasting problem* is to infer the future value \vec{x}_{t+h} based

on the current and past values of \vec{x} , where h is called the *lead time*. Typical examples of \vec{x}_t in a particle accelerator scenario include the measurement of beam current, magnet strength and temperature, and the output of loss monitors.

According to the dimension n of the input signals \vec{x} , forecasting problems can be categorized into *univariate* and *multivariate* problems. Based on the value of h , they can also be divided into *one-step-ahead* or *multistep-ahead* problems. Section II introduces existing methods in two main categories: *linear* and *nonlinear* models. The illustration of each method is accompanied by practical applications in fields such as energy and finance. This review paper chooses to focus on several typical, commonly used prospective methods, especially those of interest for applications in particle accelerator diagnostics.

In practical terms, the quantity of interest in particle accelerator operation is sometimes not only the future values of the input signals \vec{x}_{t+h} , but rather another value depending on \vec{x}_{t+h} , i.e., $y_{t+h} = f(\vec{x}_{t+h})$. An example is the probability of machine failure in h seconds following the latest measurements, where $y \in [0, 1]$. In this example, the form of the function f needs also to be inferred. Such a problem setup of learning f fits into the scope of *anomaly detection*, where an anomaly score y_t (usually $y_t \in [0, 1]$ or a non-negative value) is inferred from input \vec{x}_t at every timestamp t . However, the combined problem—extrapolating from the input time series \vec{x}_t to the future anomaly score y_{t+h} —is formulated rather ambiguously in current studies [2,3]. Section III lists several existing attempts in the particle accelerator field aiming to tackle such composite “anomaly forecasting” problems. In this context,

^{*}sichen.li@psi.ch

Also at the Department of Physics, ETH Zurich.

[†]andreas.adelmann@psi.ch

Published by the American Physical Society under the terms of the *Creative Commons Attribution 4.0 International* license. Further distribution of this work must maintain attribution to the author(s) and the published article’s title, journal citation, and DOI.

also the topic of remaining useful life (RUL) predictions in predictive maintenance is discussed, which is introduced in Sec. IV.

II. FORECASTING METHODS

According to the basic assumptions about the underlying generating process, time series models can be categorized into *linear* and *nonlinear* ones. In the former category, we introduce the autoregressive integrated moving average (ARIMA) and state-space models. In the latter category, after introducing the general concept of artificial neural network (ANN), we start from the simple multilayer perceptron (MLP) and then move toward the more complex recurrent neural network (RNN). Recent attempts in hybrid models that integrate the linear and nonlinear models are also introduced and shortly discussed.

A. Linear models

1. ARIMA

The *auto-regressive integrated moving average* (ARIMA) class of models, introduced by Box *et al.* [4], laid the foundation for many variations and further developments in time series forecasting. It assumes that the prediction is a weighted linear sum of past observations and random errors. A univariate autoregressive moving average (ARMA) model of order (p, m) follows the relation

$$x_t = \theta_0 + \epsilon_t + \sum_{i=1}^p \phi_i x_{t-i} + \sum_{j=1}^m \theta_j \epsilon_{t-j}$$

where ϵ_t is the random error, which is assumed to be independently and identically distributed with mean $\mu = 0$ and variance σ^2 ; $\phi_i, i = 1 \dots p$ and $\theta_j, j = 0 \dots m$ are model parameters, with θ_0 indicating a constant contribution [5]. It combines the previously formulated concept of the autoregressive (AR) process from Udney Yule [6] and Walker [7], and moving average (MA) techniques from the pioneering works of Allen [8]. Explicitly, by setting $m = 0$,

$$x_t = \theta_0 + \epsilon_t + \sum_{i=1}^p \phi_i x_{t-i}$$

becomes an AR model of order p . And setting $p = 0$,

$$x_t = \theta_0 + \epsilon_t + \sum_{j=1}^m \theta_j \epsilon_{t-j}$$

returns a MA model of order m [1]. The “I” in ARIMA means “integrated,” which refers to the operation of *differencing*. By calculating the differences between successive observations, namely computing $x'_t = x_t - x_{t-1}$, an originally nonstationary time series could be converted to

stationary. The general form of an ARIMA model is denoted ARIMA (p, d, m) , where d refers to the number of differencing steps needed to reach stationarity.

Other approaches, such as general exponential smoothing (GES), which was originally introduced by Brown and Meyer [9] and Holt [10] and uses multiples of polynomials, sinusoids, and exponentials of time to model the trend, could also be incorporated into ARIMA. Furthermore, according to Gardner [11], “the equivalent ARIMA model is even simpler and more efficient” than GES in standard form. The multidimensional generalization of ARIMA extends the univariate x_t into a set of n interrelated variables $\vec{x}_t = (x_{1,t}, \dots, x_{n,t})$, leading to the vector-ARIMA (VARIMA) method, where each component of \vec{x} is modeled as a linear sum of present and past values of all n components and a multivariate white noise. Such a problem is often referred to as *multiple time-series modeling* [1].

In addition to the model formulation, Box and Jenkins have also established a practical approach to build ARIMA models—now known as the *Box-Jenkins approach*—that strings together model identification, estimation, and verification into a full iterative cycle [12]. After removing potential nonstationarity and seasonality through differencing [13], a plausible model of the orders p and m is identified by checking autocorrelation patterns or other model-selection criteria of the time series. Then the model parameters θ and ϕ are fitted by minimizing the overall errors. Finally, various diagnostic checks are performed on the residual of the real series and the fitted model. The three-step cycle is typically run several times before reaching a satisfactory model. The versatility of ARIMA usually enables it to imitate time series of diverse types, without having to introduce many parameters.

Ever since its proposal, the ARIMA model has had a key role in a wide range of forecasting-related areas, including the recent application in predictions of the Covid-19 epidemic evolution [14]. De Gooijer and Hyndman [15] provide a comprehensive list of earlier empirical applications of ARIMA and its variants in the scope of the *International Journal of Forecasting* papers. More recently, the ARIMA model is serving increasingly as one of the comparison baselines for newly developed nonlinear models. As an example, Siami-Namini *et al.* [16] show the superiority of the long short-term memory (LSTM) model over ARIMA on several standard time series datasets. However, such superior performance does not challenge ARIMA’s position as the foundation of forecasting models. Considering its robustness, high interpretability as well as the black-box nature of many models based on deep learning [17], ARIMA appears increasingly as a fundamental component in hybrid models, which take advantage of its statistical properties while avoiding its linear rigidity. Zhang [5] proposes to combine the forecasts from a linear ARIMA model and a

nonlinear artificial neural network (ANN) model. On this basis, Wang *et al.* [18] test the combination of ARIMA and ANN in both additive and multiplicative ways, and the latter shows consistent improvement in forecasting accuracy compared to ARIMA and ANN individually as well as to the additive hybrid model. One notable application is that by Liu *et al.* [19] on wind speed prediction, where they employ an empirical mode-decomposition approach that uses ARIMA for low-frequency and LSTM for high-frequency subsequences prediction.

2. State-space model

A second important and practical formulation is based on *linear dynamical systems*. It develops a recursive algorithm for computing forecasts. Originating from control engineering, the model views any observation at time t as a signal part plus a noise part, and the signal is then decomposed into a linear combination of q -state variables, to form the state vector $\vec{h}_t \in \mathbb{R}^q$. The *observation* (or *measurement*) equation for univariate x_t reads

$$x_t = f(\vec{h}_t) + \epsilon_t \quad (1)$$

where f is a function and ϵ_t denotes the zero-mean noise part of time series x_t . The future values of the state vector \vec{h}_t only depend on its current value, and not on its past—in other words, the state vector has *Markovian* properties. In linear state-space models, \vec{h}_t is assumed to evolve according to the *transition* equation

$$\vec{h}_t = \mathbf{G}_t \vec{h}_{t-1} + \vec{\xi}_t \quad (2)$$

where \mathbf{G}_t is a $q \times q$ transition matrix, and $\vec{\xi}_t \in \mathbb{R}^q$ is the disturbance term of the state vector \vec{h}_t , assumed to have zero mean. State-space models make fewer assumptions about the form of the trend, yet they still can produce adaptive and robust forecasts [1]. For instance, Bae and Harris [20] show better performance of the state-space model in both cycle tracking and error reduction relative to plain multiple regression in short-term multivariate forecasts of U.S. fuel consumption.

The corresponding updating procedure of the state-space model is the so-called *Kalman filter* [21], which recursively updates the estimate of the state vector \vec{h}_t and thereby calculates the latest forecast x_t whenever a new observation becomes available. For instance, in the case of a one-dimensional state space with f as the identity function and constant $G_t = G$, we have

$$\begin{aligned} x_t &= h_t + \epsilon_t \\ h_t &= Gh_{t-1} + \xi_t \end{aligned} \quad (3)$$

where we assume $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$ and $\xi_t \sim \mathcal{N}(0, \tau^2)$. Peng [22] gives an introductory example of the Kalman-filter

algorithm with detailed derivation. It starts from an initial guess of the mean and variance of the initial state h_0 , denoted by h_0^0 and P_0^0 . The upper index refers to the number of observations that have been used to update the state, and the lower index refers to the time t . Following Eq. (3), we can calculate our guess for the mean and variance of the next state h_1

$$\begin{aligned} h_1^0 &= Gh_0^0 \quad (\text{mean}) \\ P_1^0 &= G^2 P_0^0 + \tau^2 \quad (\text{variance}) \end{aligned}$$

With the observation x_1 as new information, we can update the previous guess h_1^0 and P_1^0 to h_1^1 and P_1^1 :

$$\begin{aligned} h_1^1 &= h_1^0 + K_1(x_1 - h_1^0) \\ P_1^1 &= (1 - K_1)P_1^0 \end{aligned}$$

where $K_1 = \frac{P_1^0}{P_1^0 + \sigma^2}$ is the *Kalman gain coefficient* at $t = 1$. In general, given the current estimate h_t^t and P_t^t at the time t after updating with t observations, we can update with the new observation x_{t+1} according to

$$\begin{aligned} h_{t+1}^t &= Gh_t^t \\ P_{t+1}^t &= G^2 P_t^t + \tau^2 \\ K_{t+1} &= \frac{P_{t+1}^t}{P_{t+1}^t + \sigma^2} \\ h_{t+1}^{t+1} &= h_{t+1}^t + K_{t+1}(x_{t+1} - h_{t+1}^t) \\ P_{t+1}^{t+1} &= (1 - K_{t+1})P_{t+1}^t. \end{aligned}$$

A general ARIMA model can also be recasted in the state-space formulation to apply Kalman filtering and ease the estimation procedure [23]. With details given by Meinhold and Singpurwalla [24], Kalman filtering ensures that the minimum mean squared estimator of the state vector is obtained in case of normal noise. The flexibility of the procedure has been established in various forecasting problems. Harvey [25] has written a chapter with a detailed theoretical presentation of the method, supplemented by various applications in econometrics. Visser and Molenaar [26] have proposed a trend regression model that incorporates both deterministic and stochastic trends in a general ARIMA format for climatological data. They then transcribe the model into the state-space format and use Kalman filtering to estimate and evaluate the model parameters. The hybrid approach with neural networks is also highly fruitful here. For instance, Peel [27] builds a Kalman filter on top of an ensemble of neural network models, where he exploits its advantage of handling multiple input sources simultaneously and its intrinsic ability to filter predictions over time.

It is crucial to quantify the forecast uncertainty, both for unwrapping the underlying models and for better guiding further operations. Taking this insight into account led to the incorporation of Bayesian inference in forecasting models. One inspiring attempt to extend the capacity of linear models is that of Ghosh *et al.* [28] on traffic flow forecast, which uses Bayesian inference to fit the model parameters in place of the typical point estimation by residue minimization.

B. Nonlinear models

Although the linear models discussed above have the advantages of simple implementation, straightforward interpretability, and also good performances in some proven cases [29], the true model underlying time series may in reality be nonlinear and therefore difficult to unwrap [30]. Earlier attempts of adapting the linear models to nonlinear behavior include the bilinear model [31], which contains nonlinear cross terms of past values and white noise, but is based on structural theories analogous to linear models; and the threshold autoregressive (TAR) model [32], which combines piecewise linear models systematically and reaches global nonlinearity with local linearity. These models are mostly confined to only some specific patterns of nonlinearity, however, which in turn results in their weak performance for general problems [33].

While the methods introduced above, from linear models like ARIMA to nonlinear attempts like TAR, are rather generally considered *model-driven*, the following broad category of ML methods is more believed to be *data-driven*, in the sense that they do not necessarily require an explicit form of an underlying model. The term “machine learning” has broad and rather ambiguous meanings, with techniques ranging from ordinary least square methods to deep neural networks with millions of parameters (for instance, the well-known image recognition network *ResNet-50* has more than 23 million trainable parameters [34]). According to Bandara *et al.* [35], traditional *model-driven* methods work better when the data volume is minimal. But nowadays, complex ML models, which used to be outperformed traditional statistical models in the forecasting of simple short-time series [36,37], have gradually become dominant in the era of ever-increasing data quantity and quality. The revolutionary changes brought by big-data technology enable us to manage longer and uninterrupted time series. In addition, the access to many interrelated series has opened up novel learning possibilities. While traditional models require us to explicitly write out a specific form of correlations, an ML-based model can naturally come up with combined features from various inputs and exploit cross-series information [38]. Below, we introduce the basic concept of artificial neural network (ANN) and in particular recurrent neural network (RNN) architectures, together with some extensions and example applications.

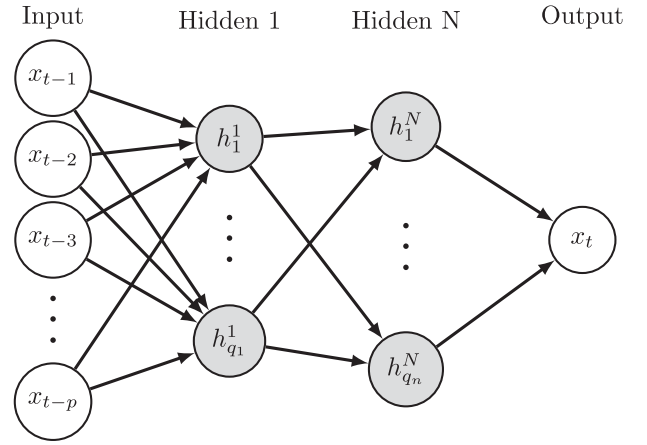


FIG. 1. Illustration of an MLP.

1. Multilayer perceptrons

An ANN is composed of connected units called neurons or nodes, where linear or nonlinear calculations are performed with numbers transmitted and received through the connections. The concept is to mimic how the neural system of humans works, even though this is a much oversimplified analogy. As the simplest form of ANN, multilayer perceptrons (MLP), also known as feed-forward neural networks, contain only forward connections between nodes, without loops as in RNNs. Here, we describe an MLP with N hidden layers as

$$\begin{aligned} \vec{x}^i &= (x_{t-1}, \dots, x_{t-p}) \in \mathbb{R}^p \\ \vec{h}^1 &= G^1(\vec{b}^1 + \mathbf{W}^{1,i} \cdot \vec{x}^i), \quad \vec{h}^1 = (h_1^1, \dots, h_{q_1}^1) \in \mathbb{R}^{q_1} \\ &\vdots \\ \vec{h}^N &= G^N(\vec{b}^N + \mathbf{W}^{N,N-1} \cdot \vec{h}^{N-1}), \quad \vec{h}^N \in \mathbb{R}^{q_N} \\ x_t &= x^o = G^o(b_o + \vec{w}^{o,N} \cdot \vec{h}^N) \end{aligned}$$

where i and o denote *input* and *output*, respectively. The input layer \vec{x}^i has p nodes that take p past values of the series. Note that \vec{x}^i is for past values of the same variable, instead of making it multivariate. The j th hidden layer \vec{h}^j has q_j nodes, and the output layer x^o has only one output, which is the current value x_t . $\mathbf{W}^{1,i}$ is a (q_1, p) matrix of connection weights from the input layer (of p nodes) to the first hidden layer (of q_1 nodes), $\mathbf{W}^{N,N-1}$ are weights from the $(N-1)$ th to the N th hidden layer, and $\vec{w}^{o,N} \in \mathbb{R}^{q_N}$ are weights from the N th hidden layer to the output. The vectors \vec{b} are the bias of each layer, and the functions G are the activation functions applied on each layer, with typical choices being sigmoid, hyperbolic tangent, or rectified linear unit [39] functions. Figure 1 visualizes the structure of an MLP.

The advantages of ANNs go beyond the aforementioned lenient requirement for *a priori* assumptions; they can

universally approximate various forms of functions [40] and perform especially well when modeling nonlinear relations. They also possess high generalization power toward out-of-sample data.

Zhang *et al.* [41] give a detailed overview of ANNs for forecasting problems, with a focus on MLP and some hybrid approaches. Starting from the earliest success by Lapedes and Farber [42] on a generated dataset following deterministic nonlinear dynamics, ANNs have been widely applied to various fields. In finance, the inherent noise, fat-tail distributions, and nonlinear patterns of the financial data make it difficult for conventional methods such as ARIMA to capture the dynamics, which lead to the popularity of ANNs. Li and Ma [43] comprehensively introduce the application of ANNs on problems ranging from microscopic ones such as exchange-rate or stock-price forecasting to macroscopic scenarios such as financial-crisis forecasting, while Krollner *et al.* [44] made a more specialized survey on ANN-related applications in stock-index forecasting.

2. Recurrent neural networks

One drawback of the MLP approach is that it isolates the inputs at every timestamp and treats them as independent variables. The temporal order and the dependencies of the input series on time contain crucial information about the evolution of the series but are not taken into account [45]. This issue is targeted and overcome by RNNs, which by design can carry forward—or in a more neurological term, “remember”—the states from previous inputs.

The unit component that constitutes an RNN is called an *RNN cell*. A generic RNN cell consists of the *input* time series x_t (in the univariate case), a *hidden state* $\vec{h}_t \in \mathbb{R}^q$ with q , the cell dimension or RNN size, and the *cell output* $\vec{o}_t \in \mathbb{R}^q$. In the case of time series forecasting, the cell output \vec{o}_t needs to be transformed again into the final network output x_{t+h} , which is the h -step-ahead prediction of x_t , normally by fully connected layers. At each time step, the hidden state \vec{h}_t is updated according to Eq. (4)

$$\left. \begin{aligned} \vec{h}_t &= G(\vec{h}_{t-1}, \mathbf{W}^i, x_t, \vec{V}^i, \vec{b}^i) \\ \vec{o}_t &= G^o(\vec{h}_t, \mathbf{W}^o, \vec{b}^o) \end{aligned} \right\} \text{inside RNN cell}$$

$$x_{t+h} = f(\vec{o}_t) \quad \text{outside RNN cell} \quad (4)$$

where $\mathbf{W}^i \in \mathbb{R}^{q \times q}$ denotes the hidden-to-hidden weight matrix, $\mathbf{W}^o \in \mathbb{R}^{q \times q}$ is the hidden-to-output weight matrix, $\vec{V}^i \in \mathbb{R}^q$ is the input-to-hidden weights (in multivariate case, i.e., $\vec{x}_t = (x_{1,t}, \dots, x_{n,t}) \in \mathbb{R}^n$, \mathbf{V}^i would become a matrix of size $q \times n$), and b^i, b^o are the bias vectors associated with the input and output, respectively. Here i and o again refer to *input* and *output*. G and G^o are nonlinear activation functions for the hidden state and output state, and x_{t+h} is the final prediction. It is shown

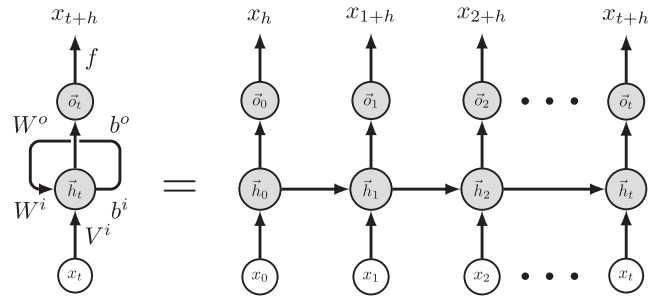


FIG. 2. Illustration of an RNN cell in folded and unfolded versions.

from the updating scheme that in each RNN cell, the current hidden state h_t is updated from the previous hidden state h_{t-1} together with the current input x_t , which enables the network to retain information from the recent past. Their similar forms with Eqs. (1) and (2) reveal that RNNs could be interpreted as a kind of nonlinear state-space model from a time-series perspective [46].

Figure 2 shows the propagating of a generic RNN cell in looped and unfolded view. It demonstrates the feedback structure that enables the network to propagate past states into future time steps and highlights how the learning process naturally follows the evolution of the series.

Three types of RNN cells are most widely used, especially in forecasting: the basic Elman RNN (ERNN) cell [47], the Gated Recurrent Unit (GRU) cell [48], and the Long Short Term Memory (LSTM) cell [49]. Each type of cell has its own formulation and updates the hidden state differently. An example ERNN cell with the sigmoid function σ and the hyperbolic tangent function \tanh as activation functions have the following updating scheme:

$$\begin{aligned} \vec{h}_t &= \sigma(\mathbf{W}^i \vec{h}_{t-1} + \vec{V}^i x_t + \vec{b}^i) \\ \vec{o}_t &= \tanh(\mathbf{W}^o \vec{h}_t + \vec{b}^o) \end{aligned}$$

and it is visualized in Fig. 3. The sigmoid activation function normalizes the values into $[0, 1]$ range while keeping differentiable; the hyperbolic tangent activation function brings the output into $[-1, 1]$ range and further polarizes the positive/negative values.

To overcome the issue of the short memory of the simple ERNN cell due to gradient vanishing in long sequences, the LSTM cell is developed to incorporate long-term dependencies from the earlier past into the model [50]. The LSTM cell is composed of three gates: the *forget gate* shown in orange, the *input gate* shown in green, and the *output gate* shown in blue. Unlike the ERNN cell which has only h_t as its state, the state of the LSTM cell contains two components: the hidden state \vec{h}_t corresponding to short-term memory and the internal cell state \vec{c}_t responsible for long-term memory. To start with, the *forget gate* takes the input and the previous hidden state, and outputs with

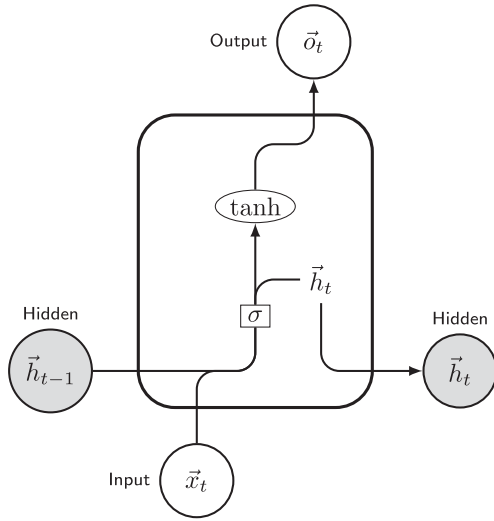


FIG. 3. The structure and updating scheme of an ERNN cell.

the sigmoid function the portions of old information that should be forgotten

$$\vec{f}_t = \sigma(\mathbf{W}^f \vec{h}_{t-1} + \vec{V}^f x_t + \vec{b}^f),$$

where \mathbf{W}^f , \vec{V}^f , and \vec{b}^f denote the weights and bias of the *forget gate*. Then the cell state \vec{c}_t is updated with a candidate cell state \vec{c}_t and the portions \vec{i}_t of the current information that should be kept from the *input gate*.

$$\begin{aligned} \vec{i}_t &= \sigma(\mathbf{W}^i \vec{h}_{t-1} + \vec{V}^i x_t + \vec{b}^i) \\ \vec{c}_t &= \tanh(\mathbf{W}^c \vec{h}_{t-1} + \vec{V}^c x_t + \vec{b}^c) \\ \vec{c}_t &= \vec{f}_t \times \vec{c}_{t-1} + \vec{i}_t \times \vec{c}_t, \end{aligned}$$

where the superscripts i and c refer to *input* and *cell*, respectively. Finally in the *output gate*, the hidden state \vec{h}_t is updated for the next iteration.

$$\begin{aligned} \vec{o}_t &= \sigma(\mathbf{W}^o \vec{h}_{t-1} + \vec{V}^o x_t + \vec{b}^o) \\ \vec{h}_t &= \vec{o}_t \times \tanh(\vec{c}_t) \end{aligned}$$

The whole operational process of an LSTM cell is depicted in Fig. 4.

According to Hewamalage *et al.* [38], RNNs are suitable for modeling time series with homogeneous seasonal patterns; otherwise, the seasonality of the series needs to be properly handled. Suilin [51] addressed this by using attention weights, with the idea that if a series possesses daily periodicity, it is beneficial to assign more weight to the value 1 day ago. This might be a useful technique in particle accelerator scenario, where periodicity is either observed or even required.

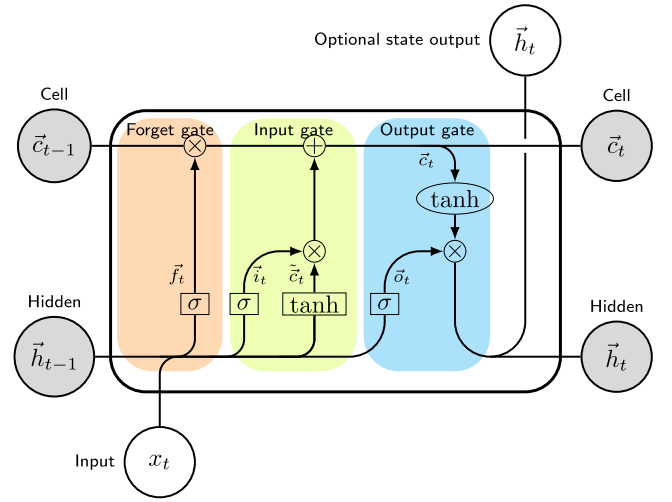


FIG. 4. The structure and updating scheme of an LSTM cell.

New developments have brought an innovative trend that further integrates the deep neural network methodology—which possesses strong learning capability—with the traditional models, which are more stable and interpretable. Compared to the above-mentioned hybrid models where the final output is simply composed of outputs from multiple separate submodels, recent novel architectures aim to merge from a more fundamental level. The DeepAR algorithm is capable of producing accurate probabilistic forecasts by training an autoregressive recurrent neural network on multivariate-related time series [52]. Furthermore, the deep state-space model [53] parametrizes a linear state-space model per time series with a jointly learned recurrent neural network. As a consequence, desired properties from both sides are satisfied: data efficiency and interpretability from state-space models and the ability to learn complex patterns from deep learning approaches.

III. APPLICATION IN PARTICLE ACCELERATOR DIAGNOSTICS

Recent years have seen a boost in applications of data-driven methods in theoretical and engineering research around particle accelerators. Among all promising areas of application, such as beam dynamics modeling [54] and beam energy optimization [55,56], diagnostics and control have always been an indispensable and crucial part in ensuring a stable and more productive operation of the accelerator [57,58]. With appropriate forecasting methods including but not limited to the above-mentioned ones, short-term anomalous events such as equipment failures may be mitigated promptly by fast recovery operations, and long-term parameter drifts could also be compensated according to their forecasted future changes. These methods have indeed aroused some interest in the field, for instance, a recent application of the Kalman filter technique

in a particle accelerator use case was presented by Syed *et al.* [59] from the European XFEL, where they applied it for anomaly detection of superconducting rf cavities. By introducing the Koopman operator, they achieved a speed-up of 3 orders of magnitudes with a linear approximation of the previous nonlinear state-space model. However, attempts in this area that frame such applications as forecasting problems are not yet emerging widely. Due to the complex nature of the data and large diversities across different accelerators, the problem of predicting the future behavior of an accelerator facility is not as straightforward to formulate as that of a degrading engine. Here we first present four studies from the Spallation Neutron Source (SNS) at Oakridge, the High Intensity Proton Accelerators (HIPA) at the Paul Scherrer Institute, the Advanced Photon Source (APS) at Argonne National Laboratory and the Continuous Electron Beam Accelerator Facility (CEBAF) at Jefferson Lab. They all focus on the *short-term* failure prediction from an anomaly detection perspective, explore existing models or attempt to establish new models, thereby opening up possibilities for future research. The goal of all four studies is to identify potential anomalous events in advance, which embeds a different concept of *forecasting* than the previously introduced models. Then, we introduce a newly published study from CERN that implements *autoregressive* modeling for beam loss prediction to cope with machine drift on the timescale of years. Following a similar direction, a diagnostic and feedback system that utilizes time evolution to reduce energy deviation is also planned at the HiRES beamline at the Lawrence Berkeley National Laboratory.

A. Preemptive detection of machine trips at the Spallation Neutron Source

The accelerator system of SNS delivers proton pulses of μs timescale to a liquid mercury target in a stainless steel container, for the production of neutrons through the spallation process [60]. Each of the beam loss trips in SNS costs around 40 s downtime, which amounts to about 33,000 lost pulses daily. If such failure could be predicted in advance, the machine protection system (MPS) could react to it by suspending the beam production and resetting the machine, which would in turn reduce the downtime to 1 s each. In addition, the reduction of beam loss trips could also lower the damage to the superconducting cavities and reduce the radioactivation of the accelerator. Compared to existing methods that identify the machine trips, Reščič *et al.* [61] aim to provide an approach that is not only generalizable and system-agnostic across all subsystems and machines but also preemptive, in that it should predict failures in advance instead of reporting them after they have already occurred.

The data used in this work are univariate pulses taken from the SNS differential beam current monitor (DCM) in

March 2021. Each pulse is a waveform of 120,000 data points at a frequency of 100 MHz. The length between two consequent pulses is about 16 ms during normal operation, which is referred to as the *time budget* allowed to make predictions. The problem is formulated as binary classification, where the pulses before the machine trips are labeled as *bad* pulses. When the model outputs a *bad* label that indicates a potential failure, the actual trip would then come after the current pulse, and in this way, forecasting is realized.

In a previous study, Reščič *et al.* [62] have gone through a holistic research of ML classification methods, including logistic regression, k nearest neighbors, tree-based methods, support vector machines, and MLPs and achieved almost 92% accuracy in identifying bad pulses from an MLP model combined with parameter tuning and data refining. However, only the pulses right before and right after the trips are taken and labeled, and nearly 8% of daily good pulses are incorrectly predicted as bad. Their following-up work [61] improves the result with a more complete dataset and introduces fast Fourier transform (FFT) for feature extraction and principal component analysis (PCA) for dimensionality reduction. About 26 pulses before the trips—instead of only one previously—are taken as bad pulses (labeled as *Before* pulses) that lead to failure, 2 pulses after the trips (labeled as *After* pulses) are taken as good pulses, and pulses in normal operation without trips are also taken and labeled as *Notrip* pulses. Both the *After* pulses and the *Notrip* pulses are classified, respectively, against the *Before* ones. By analyzing their distance to the next trips and comparing the classification results, the newly taken *Notrip* pulses are considered to be more representative of normal operation. Therefore the authors decide to focus on the *Before–Notrip* classifier. The best-performing Random Forest model together with PCA achieves 96% accuracy and 61% recall, i.e., 61% out of all real trips are successfully predicted as trips. By further leveraging classification threshold and improvement techniques, the classifier could in principle reach a strict 0% false-positive rate, at the cost of a true-positive rate of less than 58%. Figure 5 shows the receiver operating characteristics (ROC) curve and the precision-recall curve of three models on the beam loss dataset, which is taken around machine trips where beam loss occurred. Both types of curves are generated by leveraging the classification threshold from 0 to 1. The ROC curve shows the true-positive rate against the false-positive rate; the uppermost left curve is optimal and has the greatest area under the curve (AUC) value. The average precision is calculated over the full threshold range.

All classifiers successfully predict the accelerator failures inside 4 ms, far less than the available time budget of 16 ms. This enables SNS to implement the model in real-time operation and invoke mitigation techniques in field-programmable gate arrays (FPGAs) to realize the inhibition of pulses and resetting of the machine.

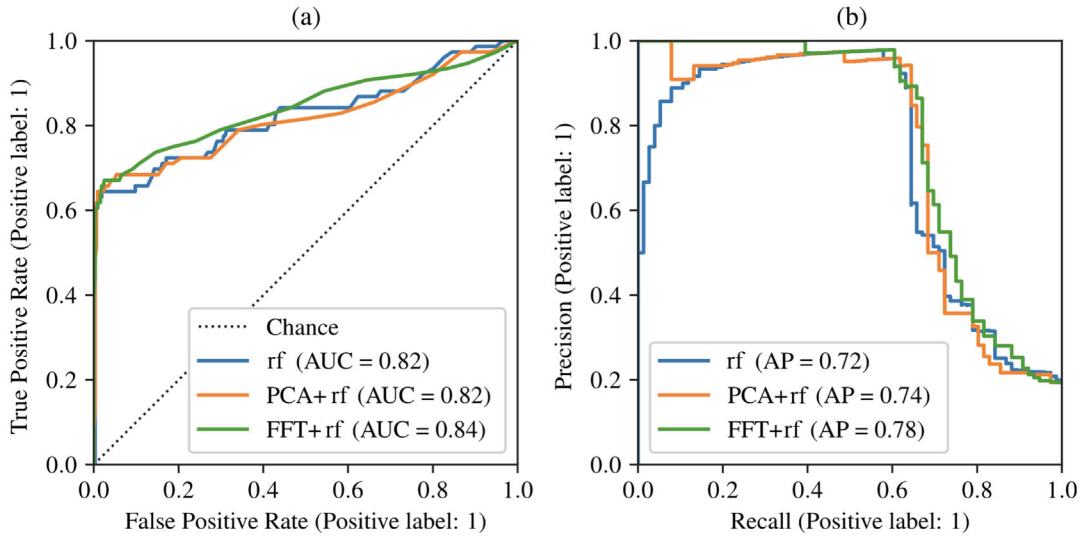


FIG. 5. The ROC curve (left) and precision-recall curve (right) of the rf classifier baseline, rf with PCA and rf with FFT models applied on the beam loss dataset (from [61], Fig. 7).

Another inspiring study using the same DCM data from SNS realizes uncertainty-aware anomaly detection of the pulses. Blokland *et al.* [63] build a Siamese neural network [64] to distinguish two types of errant pulses from normal pulses by ranking their similarity. While keeping the false-positive rate below the established 0.05% limit, the true-positive rate increases to more than 64% when training and testing with the same errant type and also reaches 45% in cross-type testing, as shown in Fig. 6.

Starting from the current offline model validation result, the authors have been working toward online prediction together with real-time implementation, where they compare the incoming pulse with a series of past pulses using the Siamese network, make decisions based on their similarity level, and abort the predicted errant beam to reduce system downtime.

B. Interlock forecasting of the High Intensity Proton Accelerators

HIPA produces a proton beam of nearly 1.4 MW power, which makes it one of the most powerful proton cyclotron facilities in the world [65]. The interlock system is part of the Machine Protection System that immediately shuts off the beam whenever some monitor signal exceeds the safety limit. However, such shutdowns may lead to abrupt operational changes and lose equivalently 25 s of beam time each. Li *et al.* [66] propose to build a forecasting model of the interlocks. Once the model reports an incoming interlock, the suggested recovery operation to reduce the beam current by 10% would be applied, which could potentially circumvent the interlocks from happening, thus saving beam time for the users.

The dataset is composed of 376 process variables from the Experimental Physics and Industrial Control System

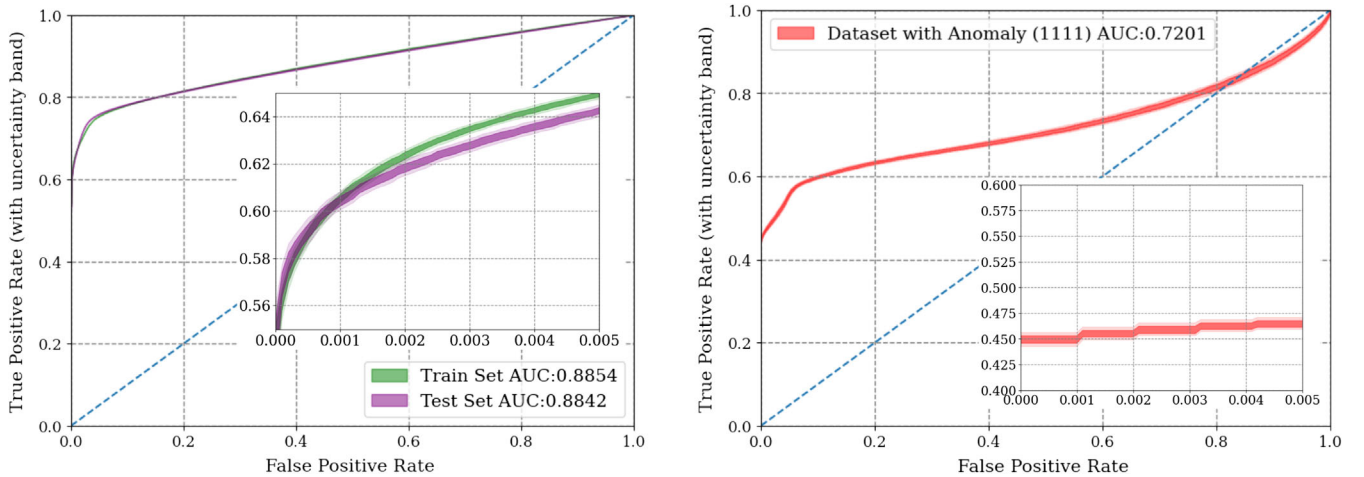


FIG. 6. The ROC curves for training and testing with the same (left) and different (right) errant types (from [63], Figs. 14 and 15).

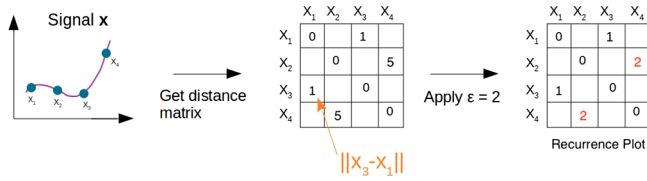


FIG. 7. Generation of recurrence plot from a signal with fixed $\epsilon = 2$ (from [66], Fig. 7).

(EPICS), which are recorded at a 5-Hz frequency. The problem is formulated as a binary classification of two classes of samples. The positive class consists of *interlock samples* that are taken at 1 to 12 s before the interlocks. This is how the concept of forecasting is embedded here, just like taking the pulses before the trips in the SNS case. The negative class consists of *stable samples* that are taken in the middle between two adjacent interlocks with a buffer region of 10 m on both sides, to represent stable operating states. The window length is a trainable parameter to be decided in the model.

The authors develop a recurrence plot-convolution neural network (RPCNN) model for the classification task. Each one-dimensional time series of the input is transformed into a two-dimensional *recurrence plot* (RP) to extract finer dynamical patterns. Then the plots are trained with a *convolutional neural network* (CNN), which is an established and powerful method in image processing. Recurrence plots (RPs) were developed to detect hidden dynamical patterns and nonlinearities in dynamical systems [67]. The structures in a recurrence plot reveal detailed information about the system's time progression. The authors use the so-called global recurrence plot [68] with a fixed threshold distance ϵ , as defined in Eq. (5)

$$R_{i,j} = \begin{cases} \|\vec{x}_i - \vec{x}_j\|, & \|\vec{x}_i - \vec{x}_j\| \leq \epsilon \\ \epsilon, & \|\vec{x}_i - \vec{x}_j\| > \epsilon \end{cases} \quad (5)$$

where i, j are the indices of time steps inside the time window taken from the signals, and the resulting R is symmetric. Figure 7 diagrammatically depicts the process of transforming an initial signal into a recurrence plot with fixed $\epsilon = 2$. Figure 8 lists several examples of actual recurrence plots generated from the RPCNN model, whose patterns convey a wealth of information not immediately available from the time series they are built from, such as the band structures indicating abrupt changes. The process of sample taking, recurrence plot generation and model construction is described in Fig. 9.

In practice, the recurrence plots are produced internally by a custom *recurrence plot layer* before the convolutional and max-pooling layers. This procedure prevents the recurrence plots from being generated and stored explicitly beforehand and also allows ϵ to be a trainable parameter and the optimization of the plots on the fly. The output is a score $y \in [0, 1]$, indicating the probability that the incoming sample belongs to the positive class, thereby forecasting an interlock.

The authors choose the best-performing model based on a custom metric they call *beam time saved*, which computes potential time saved by invoking the recovery operation back on the machine. As the recovery operation would cost an equivalent of 6 s of beam-time loss per interlock, false positives need to be strictly controlled in order to reach a bonus in *beam time saved*. Therefore, the resulting classifier has a true-positive rate of 4.9%, together with an extremely low false-positive rate of 0.17%, and it can potentially save 0.5 s of beam time per interlock. Figure 10 shows the best and mean ROC curves of RPCNN classifiers with the random initialization, as well as their uncertainties.

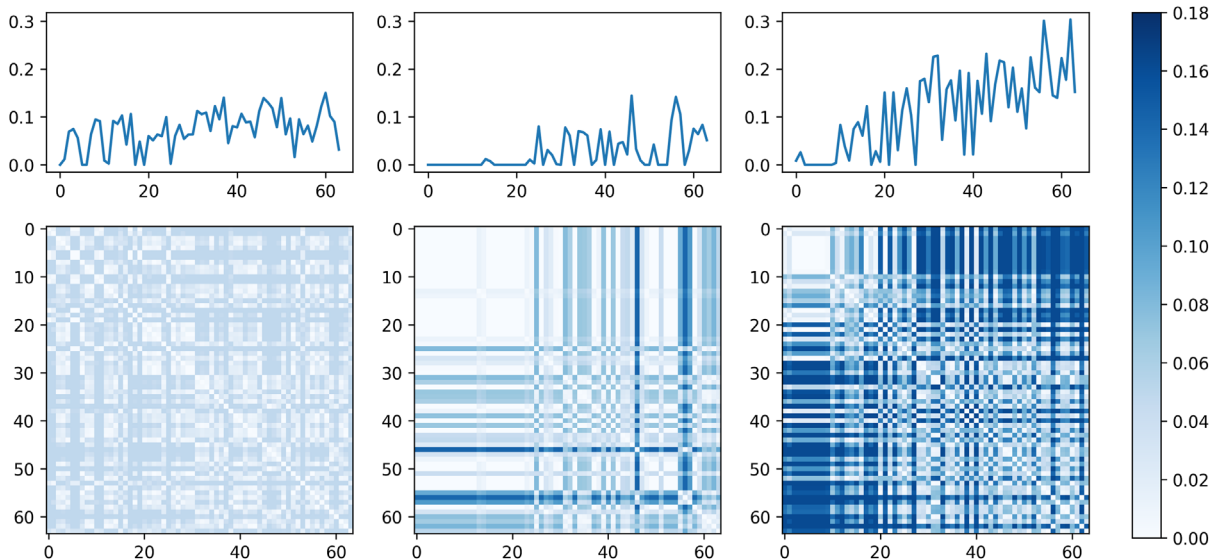


FIG. 8. Examples of recurrence plots (bottom row) generated from the original signals (top row) by the RPCNN model. From left to right: uncorrelated stochastic data, data starting to grow, and stochastic data with a linear trend (from [66], Fig. 8).

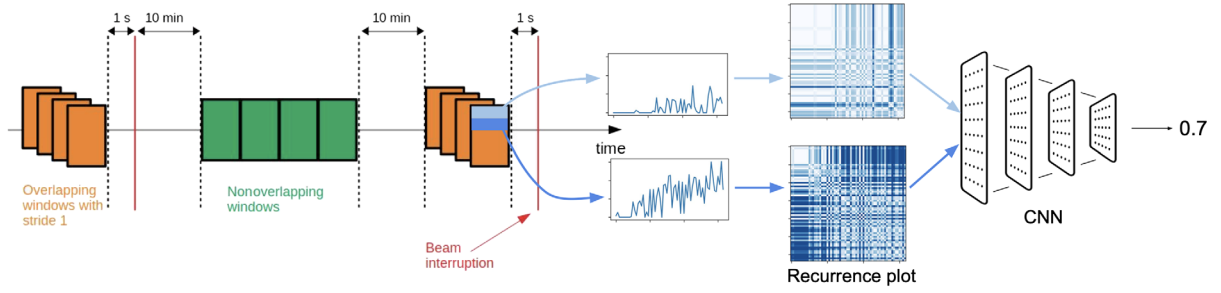


FIG. 9. The RPCNN model structure. The positive and negative classes of samples are taken either close to (orange) or far from (green) the interlocks. Each of the 376-time series is transformed into recurrence plots and fed into the CNN. The model output is a probability value (adapted from [66], Figs. 5 and 8).

To alleviate the limitations of false positives, the authors further study the input channels and discover from statistical tests that a significant difference is only present inside 0.4 s before the interlocks. A preliminary study with a linear least absolute shrinkage and selection operator (LASSO) model in which only single timestamps are used and the positive samples are pushed closer to the interlocks, showed improvements in both classification power and stability. The *beam time saved* metric is also modified to fit the continuous real-time context, and the new model is shown to potentially save around 6 min beam time in a day.

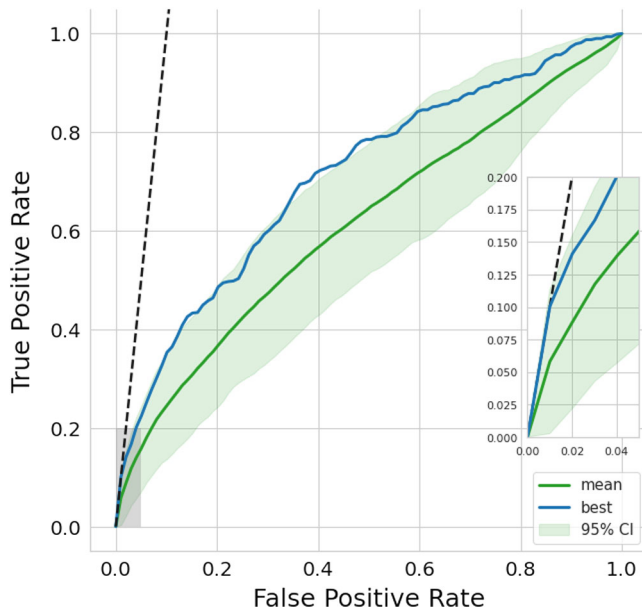


FIG. 10. The ROC curves of an ensemble of RPCNN classifiers. The blue line shows the best classifier with $AUC = 0.71$, the green line is the mean curve, and the shaded area is the 95% confidence interval of different model initialization profiles. The dashed line is the separatrix, left of which there is a positive *beam time saved*. The inset is enlarged on the gray-shaded region (from [66], Fig. 11).

C. Power supply trips prediction in the Advanced Photon Source storage ring

The Advanced Photon Source (APS) at Argonne National Lab provides ultrabright x-ray beams of 7 GeV for advanced research. Trips in the magnet power supplies of the storage ring are highly undesirable, as they would cause complete electron beam loss and interrupt user experiments instantly. Because the trips are rare events with diverse triggering mechanisms, overfitting would become inevitable in supervised learning, and labeling them as one class cannot reflect reality either. Therefore, Lobach *et al.* [69] focus on unsupervised anomaly detection methods that train on normal operation data and identify trip precursors by measuring their level of deviation.

For the temperature anomalies caused by valve faults in the water-cooling system, the authors apply the *spectral residual saliency detection* method on the temperatures of 680 power supplies in a time window of 3 h. The anomalies clearly stand out on the saliency maps, and the model successfully gives warnings up to 30 m in advance.

The authors achieve an even earlier advanced warning of 1 h by training an *autoencoder* on normal operation data of the temperatures from 40 averaged power supplies and tracking the reconstruction error at each time step. If only one power supply temperature is considered as input, an autoencoder trained and tested on sliding windows of 20 m would even give the warning 6 h before the trip happens.

Following the above success, the authors employ the autoencoder approach again on power supply current anomalies and obtain a preliminary result of a 20% true-positive rate. While there is still much room for improvement in the current approach, they already show great potential in early-enough warning and possibilities for preventive action.

D. Real-time cavity fault prediction at the Continuous Electron Beam Accelerator Facility

The Continuous Electron Beam Accelerator Facility (CEBAF) at Jefferson Lab is a high power, continuous wave recirculating Linac whose peak energy reaches

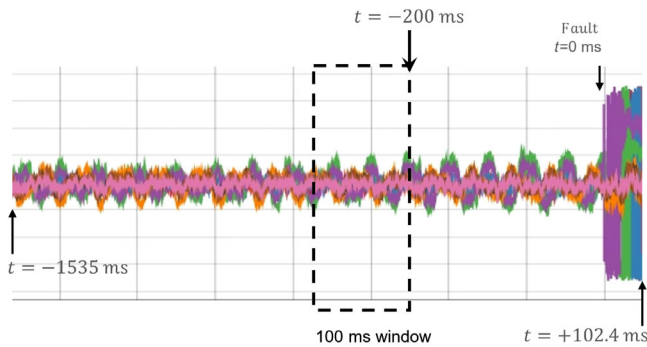


FIG. 11. Recorded waveforms and window taking of the CEBAF SRF fault forecasting model (from [71], Fig. 2).

12 GeV. The cryomodules that provide the energy gain are composed of superconducting radio-frequency (SRF) cavities, and the machine experiences frequent downtimes caused by various types of SRF faults—on average 4.1 times in an hour, which amounts to about 1 h beam time loss per day. Tennant *et al.* [70] have proposed successful machine learning models that realize fast classification of SRF faults. In a recent follow-up work [71], the authors go on to build deep learning based forecasting models for such faults. The data acquired from CEBAF contain waveforms of 17 rf signals, 4 of which are used in the study. The sample interval is 0.2 ms, and each waveform lasts for 1637.4 ms including 1535 ms before and 102.4 ms after the fault onset. The forecasting of impending faults is also formulated as binary classification between two classes of 100 ms windows—the *stable class* taking from normal running conditions and the *prefault class* taking at *lead time* $h \in [200, 100, 50, 20, 10, 5, 0]$ ms before the fault onset. Figure 11 shows an example of the recorded waveforms of the four signals and illustrates the $h = 200$ ms window taking of the *prefault class*.

Adopting the U-Net architecture [72], the binary classification model is trained only with *normal class* samples and aims to output similar normal samples by minimizing the reconstruction loss. During testing, a *prefault* sample would lead to a larger reconstruction error thus indicating its abnormality. Figure 12 lists the ROC curves for a different time before the fault onset. The closest prediction at $h = 5$ ms reaches $AUC = 0.83$, whereas the earliest prediction at $h = 200$ ms has $AUC = 0.71$. Though it is evident that the performance declines with a longer prediction time, the results have specified the timescales—about a few hundred milliseconds—for possible mitigating operations.

Furthermore, the authors build a subsequent multiclass classification network for fault-type identification from the outputs of the previous model that are classified as upcoming faults. Results show that fast-developing faults are harder to identify than slow-developing ones if samples are taken far from the faults. This echoes the discovery of HIPA interlocks mentioned in III B that the forecasting of abrupt anomalies is challenging.

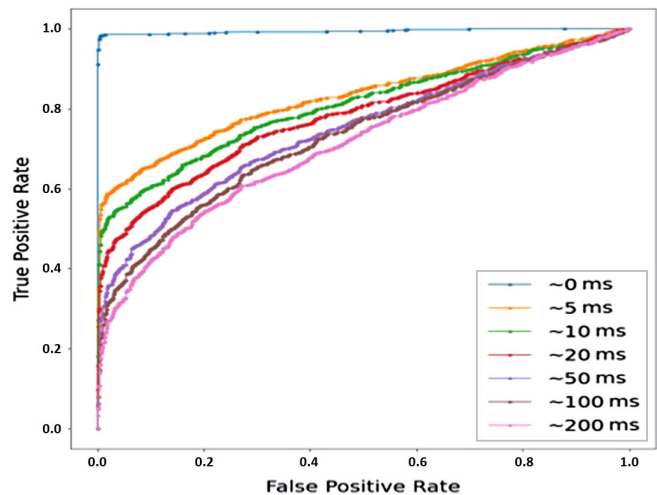


FIG. 12. ROC curves of classifying samples taken at different times before the SRF faults against normal samples (from [71], Fig. 5).

E. Beam loss prediction at the Large Hadron Collider

Beam losses in the Large Hadron Collider (LHC) at CERN are mostly occurring in the collimation system to remove particles with excessively high oscillation or momentum. The loss level is manually optimized by multiple control variables, including vertical and horizontal tunes, and currents in the focusing and defocusing magnets along the collider. It is therefore crucial to model the beam loss from those control variables for better machine operation. However, there is the problem of generalization shown by a previous study [73], when the model is trained on previous LHC fills and then applied to fills of another year. Krymova *et al.* [74] propose an *autoregressive* approach that factors in the past value of losses to alleviate the problem. To take advantage of the efficient inference procedure of *state-space models*, the authors transcribe the autoregressive formulation into a Kalman filter formulation and established several model variants with different inputs and outputs. In addition to the control variables, measurements of emittance and heat load sum are also taken into account as possible inputs. By making the parameter matrices dependent on the control variables, the authors manage to build a nonlinear model including cross terms between different inputs. For the estimation of model parameters, a customized expectation-maximization algorithm is implemented.

The authors use R^2 -score as the metric to evaluate the prediction performance, which is defined as

$$R^2 := 1 - \frac{\sum_t (y_t - \hat{y}_t)^2}{\sum_t (y_t - \bar{y}_t)^2}$$

where y_t is the ground truth, \hat{y}_t is the corresponding prediction, and \bar{y}_t is the mean of all observations in 1 year's

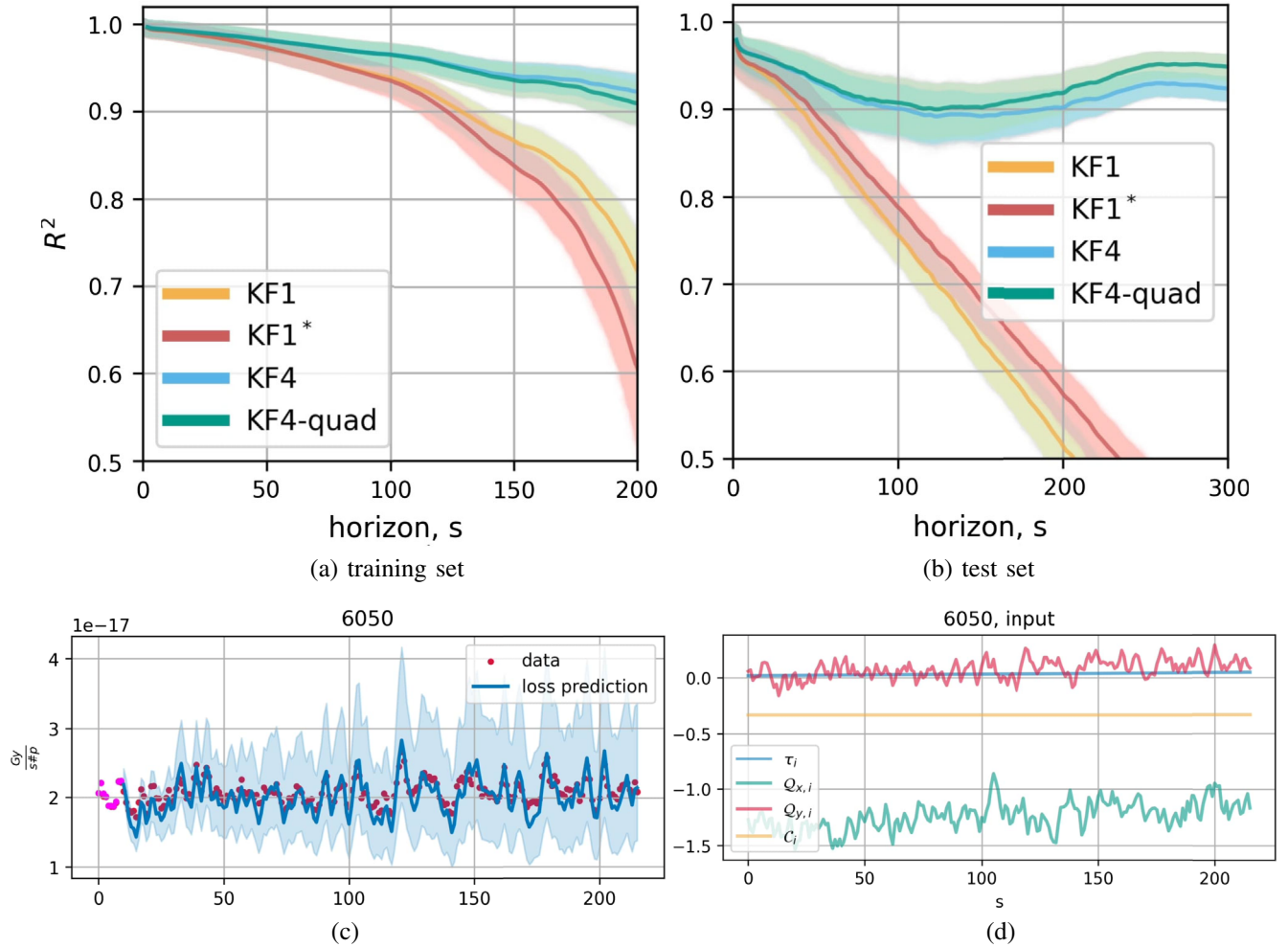


FIG. 13. (a) and (b) show the R^2 -score against different forecast horizons, where (a) is the result of the 2017 training set and (b) is the 2018 testing set. More specifically, the central lines and shaded areas are the mean and envelope of the R^2 -score of 1000 bootstrapped predictions, respectively. (c) shows the prediction result of Fill 6050 from the KF4 model with a 2σ confidence band, and (d) is the values of the corresponding four input variables for (c). The horizons are 200 s in 2017 and 300 s in 2018 (from [74], Figs. 8 and 10).

dataset. Figure 13 shows the R^2 -score of predictions against forecast horizons from four models, together with the prediction result and corresponding input series of an example fill from the model KF4, which involves noncontrolled measurements and additional output components. The model parameters are estimated from a training set with data taken in 2017 and evaluated on a 2018 testing set. The excellent performance on long horizons has provided encouraging evidence that a carefully designed model can capture the global trend and simultaneously establish a relation with inputs.

F. Proposed energy stability prediction at the High Repetition rate Electron Scattering beamline

The High Repetition rate Electron Scattering (HiRES) beamline at Lawrence Berkeley National Laboratory is a state-of-the-art compact machine for MHz ultrafast electron

diffraction (UED) pulses. Scheinker *et al.* [75,76] have previously proposed extremum seeking (ES) as an optimization technique that realizes automatic and model-independent tuning for accelerator parameters. It is also proved to be robust against drift that brings the system outside the training range [77].

With the help of the high-resolution FPGA-enabled feedback system, HiRES is shown to be stable against jitters and it is established that the machine could reach an energy stability of $\Delta E/E = 5 \times 10^{-5}$ on short timescales. However, unknown parameter drifts on longer timescales could magnify the energy deviation by more than 10 times. A nonstatic diagnostic model that involves time evolution is therefore proposed, which would be integrated with the previous optimization technique to establish a novel suite of ML-based adaptive control systems for intelligent feedback.

To infer the energy stability $\Delta E/E$, Cropp *et al.* [78] have set out to predict the beam x -position from the

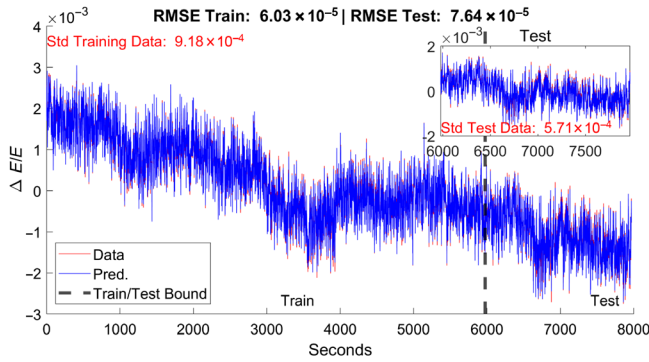


FIG. 14. Linear regression result of energy stability. The dashed line splits the training (left) and test (right) data (from [78], Fig. 6).

amplitude and phase of cavity probes, rf power, and laser properties on a virtual cathode camera. Currently, results such as the one shown in Fig. 14 are still achieved by simple linear regression at each time step, without considering past information. According to the authors, this preliminary regression result has already helped to ensure much less fluctuation than in the original hardware

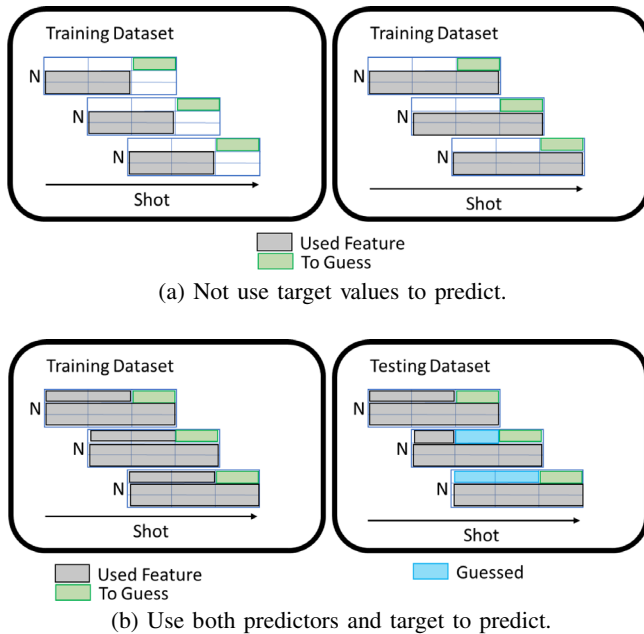


FIG. 15. Possible setup for time series prediction. Each grid denotes a time step; the time window is taken within one shot of the electron pulse. The gray areas are the input of the model, used as a predictors to predict the target, while N denotes the number of input features. The green areas are the target value expected to be predicted at each time step. (a) Predictors (gray) and target (green) are different variables, while the left and right plots show whether to include the present predictors to predict the present target. (b) Target is included as predictor. In the right plot, past predictions of the target (blue) are used in the testing dataset (from [78], Figs. 4 and 5).

feedback system. Starting from this, the authors consider time series forecasting techniques and conceive various possible ways to incorporate time evolution into the model, as shown in Fig. 15. Following the result presented in Sec. III E, such a model involving time would be a natural oppessor for long-term drifts once properly defined.

IV. DISCUSSION AND CONCLUSION

Several current and planned applications of forecasting methods on accelerators have been presented in Sec. III. For the *short-term* anomaly prediction problems, researchers mainly formulate it as binary classification and embed the concept of forecasting into the collection of positive class samples before the anomalies. The variety of failures range from beam loss trips, machine interlocks, and magnetic power supply faults to cavity faults, and the models applied are likewise extensive, from classical Random Forest and Lasso models to CNNs and Autoencoders. These models have generally performed well on archived data, and some have promised the feasibility to be implemented in real time. Leveraging the 16-ms time interval between pulses, the SNS model is able to give predictions before the next pulse comes, and fast mitigation techniques are already under development. The APS autoencoder model for power supply trips could reach early warnings in the timescale of hours, while interlocks at HIPA and cavity faults at CEBAF are both predicted in the timescale of hundred ms. For the *long-term* forecasting, a Kalman filter based model trained on the beam loss data at CERN in 2017 is capable of predicting the loss in the next 300 ms in 2018 with R^2 over 0.9. These established models could already work alongside the existing diagnostic or protection systems as references to monitor the failures or drifts.

In spite of what has been accomplished, there is still much room left for improvement. The suppression of false positives usually comes at the expense of the true positives, as seen in the example of the Random Forest model of SNS and the RPCNN model of HIPA, which could prompt us to more dynamic classification thresholds or advanced model structure, such as the Siamese network explored by SNS. The performance decline with a longer horizon is another common issue that is present in CEBAF and CERN examples. Recurrent models, especially the LSTM model introduced in Sec. II B 2 are designed to deal with past dependencies and could be expected to tackle such problems. Furthermore, the interpretability of the models, namely localization or attribution of the result to particular inputs, could be examined by methods such as sensitivity or causal analysis.

A. Challenges of ML application in accelerators

The main problem reported in ML applications for particle accelerator scenarios is the data—although a large

quantity of data is available, there is still a long process to go through until an instructive or even deployable model can be built. First, decisions need to be made about which data source is to be taken, how to extract and record the data, and about the merging of possibly different logging systems. Then the storage format needs to be decided and unified, which has to meet also the requirement for extensibility and easy query. Once the model is built, further issues arise such as accommodating the real-time input, online learning, model updating, storage of temporary results, and finally, feeding the operations invoked by the model output back to the machine. Such issues have already raised considerable interest and effort in the community. For instance, Kafkes and St. John [79] have published the Booster Operation Optimization Sequential Time-series for Regression (BOOSTR) dataset, which is composed of 15 Hz cycle-by-cycle multivariate time series of readings and settings from devices of the Rapid-Cycling Synchrotron at Fermilab. Such an attempt is very encouraging, as it addresses the problem from the root, while also aiming to create a more open and inclusive system to promote the data-driven research in the particle accelerator community.

Another ever-present issue for ML application is how to achieve better inclusion of expert knowledge. The studies at SNS and CEBAF, for instance, have started to take various known types of anomalies into account and examine cross-type performance or build multiclass models, and such kind of integration or comparison could be made even more in depth. For example, the manual record of failure types may be compared with ML multiclass classification models, and the log book may be utilized by natural language processing techniques.

B. Insight from related fields

The remaining useful life (RUL) prediction has been an important research topic in the predictive maintenance field, aiming to detect possible defects early and thus to identify and apply the required maintenance activities such that possible breakdowns are avoided. Instead of predicting an anomaly score for the input signals at every time step, the output for RUL predictions is the duration from the current time until the nearest failure. According to Kang *et al.* [80], ML techniques have also spawned many new attempts in this area, and the problem settings are also migratable to particle accelerator control, despite the time-scale difference. However, RUL prediction is mainly applied in device degradation, which possesses a clear gradual change curve. There are even physical models established to explain such effects. Equivalent models lack accelerator control, and preliminary trials have shown that such methods do not yield satisfactory performance on abrupt failures.

In conclusion, there are bright prospects for the application of data-driven time series forecasting techniques in

problems related to particle accelerators, especially in control and diagnostics. The field would benefit from an extension of current research, increasing attention to data quality, innovative insights from similar fields, and more intense exchange. In this way, time series forecasting models will emerge that are more tightly tailored to particle accelerator scenarios.

-
- [1] C. Chatfield, *Time-series Forecasting* (Chapman and Hall/CRC, London, 2000).
 - [2] M. Fahim and A. Sillitti, *IEEE Access* **7**, 81664 (2019).
 - [3] X.-X. Lin, P. Lin, and E.-H. Yeh, *IEEE Netw.* **35**, 212 (2020).
 - [4] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time Series Analysis: Forecasting and Control* (John Wiley & Sons, New York, 2015).
 - [5] G. P. Zhang, *Neurocomputing* **50**, 159 (2003).
 - [6] G. Udny Yule, *Philos. Trans. R. Soc. A* **226**, 267 (1927).
 - [7] G. T. Walker, *Proc. R. Soc. A* **131**, 518 (1931).
 - [8] R. G. Allen, *Econometrica* **18**, 209 (1950).
 - [9] R. G. Brown and R. F. Meyer, *Oper. Res.* **9**, 673 (1961).
 - [10] C. C. Holt, *Planning Production, Inventories, and Work Force* (Prentice-Hall, NJ, 1960).
 - [11] E. S. Gardner Jr., *J. Forecast.* **4**, 1 (1985).
 - [12] K. W. Hipel, A. I. McLeod, and W. C. Lennox, *Water Resour. Res.* **13**, 567 (1977).
 - [13] D. A. Dickey and S. G. Pantula, *J. Bus. Econ. Stat.* **5**, 455 (1987).
 - [14] D. Benvenuto, M. Giovanetti, L. Vassallo, S. Angeletti, and M. Ciccozzi, *Data Brief* **29**, 105340 (2020).
 - [15] J. G. De Gooijer and R. J. Hyndman, *Int. J. Forecast.* **22**, 443 (2006).
 - [16] S. Siami-Namini, N. Tavakoli, and A. S. Namin, A comparison of arima and lstm in forecasting time series, in *Proceedings of 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)* (IEEE, 2018), pp. 1394–1401.
 - [17] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, *PLoS One* **13**, e0194889 (2018).
 - [18] L. Wang, H. Zou, J. Su, L. Li, and S. Chaudhry, *Syst. Res. Behav. Sci.* **30**, 244 (2013).
 - [19] M.-D. Liu, L. Ding, and Y.-L. Bai, *Energy Conversion and Management* **233**, 113917 (2021).
 - [20] K. Bae and D. Harris, *Nonrenewable Resour.* **4**, 325 (1995).
 - [21] R. E. Kalman, *J. Basic Eng.* **82**, 35 (1960).
 - [22] R. D. Peng, *A Very Short Course on Time Series Analysis*, <https://bookdown.org/rdpeng/timeseriesbook/> (2020).
 - [23] A. C. Harvey, *Forecasting, Structural Time Series Models and the Kalman Filter* (Cambridge University Press, Cambridge, England, 1990).
 - [24] R. J. Meinhold and N. D. Singpurwalla, *Am. Stat.* **37**, 123 (1983).
 - [25] A. C. Harvey, in *Advances in Econometrics. Fifth World Congress* (Cambridge University Press, Cambridge, England, 1987), Vol. 1, pp. 285–313.
 - [26] H. Visser and J. Molenaar, *J. Clim.* **8**, 969 (1995).
 - [27] L. Peel, Data driven prognostics using a kalman filter ensemble of neural network models, in *Proceedings of*

- 2008 *International Conference on Prognostics and Health Management, Denver, CO* (IEEE, New York, 2008), pp. 1–6.
- [28] B. Ghosh, B. Basu, and M. O'Mahony, *J. Transp. Eng.* **133**, 180 (2007).
- [29] P. Han, P. Wang, M. Tian, S. Zhang, J. Liu, and D. Zhu, Application of the arima models in drought forecasting using the standardized precipitation index, in *Proceedings of International Conference on Computer and Computing Technologies in Agriculture, Jilin, China* (Springer, New York, 2012), pp. 352–358.
- [30] C. Cheng, A. Sa-Ngasoongsong, O. Beyca, T. Le, H. Yang, Z. Kong, and S. T. Bukkapatnam, *IIE Trans.* **47**, 1053 (2015).
- [31] T. S. Rao, *J. R. Stat. Soc. Ser. B* **43**, 244 (1981).
- [32] H. Tong, *Threshold Models in Non-Linear Time Series Analysis*, Lecture Notes in Statistics (Springer, New York, 2012), pp. 59–121, [10.1007/978-1-4684-7888-4_3](https://doi.org/10.1007/978-1-4684-7888-4_3).
- [33] J. G. De Gooijer and K. Kumar, *Int. J. Forecast.* **8**, 135 (1992).
- [34] K. He, X. Zhang, S. Ren, and J. Sun, Deep residual learning for image recognition, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Juan, PR* (IEEE, New York, 2016), pp. 770–778.
- [35] K. Bandara, C. Bergmeir, and S. Smyl, *Expert Syst. Appl.* **140**, 112896 (2020).
- [36] S. F. Crone, M. Hibon, and K. Nikolopoulos, *Int. J. Forecast.* **27**, 635 (2011).
- [37] Rob J. Hyndman, A brief history of time series forecasting competitions, <https://robjhyndman.com/hyndsight/forecasting-competitions/> (2018).
- [38] H. Hewamalage, C. Bergmeir, and K. Bandara, *Int. J. Forecast.* **37**, 388 (2021).
- [39] V. Nair and G. E. Hinton, Rectified linear units improve restricted Boltzmann machines, in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)* (Omni Press, Haifa, Israel, 2010), pp. 807–814, <http://www.icml2010.org/papers/432.pdf>.
- [40] K. Hornik, M. Stinchcombe, and H. White, *Neural Netw.* **2**, 359 (1989).
- [41] G. Zhang, B. E. Patuwo, and M. Y. Hu, *Int. J. Forecast.* **14**, 35 (1998).
- [42] A. Lapedes and R. Farber, Nonlinear signal processing using neural networks: Prediction and system modelling, *Proceedings of the first IEEE International Conference on Neural Networks* (IEEE, San Diego, CA, 1987), <https://www.osti.gov/biblio/5470451> [Los Alamos National Laboratory Reports No. LA-UR-87-2662, No. CONF-8706130-4].
- [43] Y. Li and W. Ma, Applications of artificial neural networks in financial economics: A survey, in *Proceedings of 2010 International Symposium on Computational Intelligence and Design* (IEEE, New York, 2010), Vol. 1, pp. 211–214.
- [44] B. Krollner, B. J. Vanstone, G. R. Finnie *et al.*, Financial time series forecasting with machine learning techniques: A survey, in *Proceedings of the 18th European Symposium on Artificial Neural Networks, ESANN 2010, Bruges, Belgium* (2010), <https://www.esann.org/sites/default/files/proceedings/legacy/es2010-50.pdf>.
- [45] F. M. Bianchi, E. Maiorino, M. C. Kampffmeyer, A. Rizzi, and R. Jenssen, [arXiv:1705.04378](https://arxiv.org/abs/1705.04378).
- [46] R. P. Masini, M. C. Medeiros, and E. F. Mendes, *J. Econ. Surv.* **37**, 76 (2021).
- [47] J. L. Elman, *Cogn. Sci.* **14**, 179 (1990).
- [48] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, [arXiv:1406.1078](https://arxiv.org/abs/1406.1078).
- [49] S. Hochreiter and J. Schmidhuber, *Neural Comput.* **9**, 1735 (1997).
- [50] H. Nguyen, K. P. Tran, S. Thomassey, and M. Hamad, *Int. J. Inf. Manage.* **57**, 102282 (2021).
- [51] A. Suilin, Kaggle-web-traffic (2017), <https://github.com/Arturus/kaggle-web-traffic>.
- [52] D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski, *Int. J. Forecast.* **36**, 1181 (2020).
- [53] S. S. Rangapuram, M. W. Seeger, J. Gasthaus, L. Stella, Y. Wang, and T. Januschowski, *Advances in Neural Information Processing Systems* (2018), Vol. 31, pp. 7785–7794, <https://proceedings.neurips.cc/paper/2018/file/5cf68969fb67aa6082363a6d4e6468e2-Paper.pdf>.
- [54] W. Zhao, I. Patil, B. Han, Y. Yang, L. Xing, and E. Schüler, *Radiother. Oncol.* **153**, 122 (2020).
- [55] J. Kirschner, M. Nonnenmacher, M. Mutný, A. Krause, N. Hiller, R. Ischebeck, and A. Adelman, Bayesian optimisation for fast and safe parameter tuning of swissfel, in *Proceedings of the 39th International Free-Electron Laser Conference, FEL2019, Hamburg, Germany, 2019* (JACoW, Geneva, Switzerland, 2019), pp. 707–710.
- [56] J. Kirschner, M. Mutný, A. Krause, J. Coello de Portugal, N. Hiller, and J. Snuverink, *Phys. Rev. Accel. Beams* **25**, 062802 (2022).
- [57] A. L. Edelen, S. G. Biedron, B. E. Chase, D. Edstrom, S. V. Milton, and P. Stabile, *IEEE Trans. Nucl. Sci.* **63**, 878 (2016).
- [58] A. L. Edelen, S. Biedron, S. V. Miltonpresenter, D. L. Bowring, B. E. Chase, J. P. Edelen, and J. Steimel, Neural network model of the PXIE RFQ cooling system and resonant frequency response, in *Proceedings of the 7th International Particle Accelerator Conference (IPAC2016), Busan, Korea* (JACoW, Geneva, Switzerland, 2016), [10.18429/JACoW-IPAC2016-THPOY020](https://doi.org/10.18429/JACoW-IPAC2016-THPOY020).
- [59] W. H. Syed, A. Eichler, A. Nawaz, B. Sharan, and H. Werner, Koopman-based Kalman filter for fault detection for the superconducting radio frequency cavities of the european XFEL, in *Proceedings of 2021 60th IEEE Conference on Decision and Control (CDC), Austin, TX* (IEEE, New York, 2021), pp. 6855–6860.
- [60] J. R. Haines, B. Riemer, D. K. Felde, J. D. Hunn, S. J. Pawel, and C.-C. Tsai, *J. Nucl. Mater.* **343**, 58 (2005).
- [61] M. Reščič, R. Seviour, and W. Blokland, *Nucl. Instrum. Methods Phys. Res., Sect. A* **1025**, 166064 (2022).
- [62] M. Reščič, R. Seviour, and W. Blokland, *Nucl. Instrum. Methods Phys. Res., Sect. A* **955**, 163240 (2020).
- [63] W. Blokland, P. Ramuhalli, C. Peters, Y. Yucesan, A. Zhukov, M. Schram, K. Rajput, and T. Jeske, [arXiv:2110.12006](https://arxiv.org/abs/2110.12006).
- [64] G. Koch, R. Zemel, R. Salakhutdinov *et al.*, Siamese neural networks for one-shot image recognition, in *Proceedings of the 32nd International Conference on Machine Learning, Lille, France* (2015), Vol. 37.
- [65] D. Reggiani, B. Blau, R. Dölling, P. A. Duperrex, D. Kiselev, V. Talanov, J. Welte, and M. Wohlmuther, *J. Neutron Res.* **22**, 325 (2020).

- [66] S. Li, M. Zacharias, J. Snuverink, J. Coello de Portugal, F. Perez-Cruz, D. Reggiani, and A. Adelman, *Information* **12**, 121 (2021).
- [67] J.-P. Eckmann, S. O. Kamphorst, and D. Ruelle, *Europhys. Lett.* **4**, 973 (1987).
- [68] C. L. Webber, Jr. and J. P. Zbilut, *Tutorials in Contemporary Nonlinear Methods for the Behavioral Sciences* (2005), pp. 26–94, <http://www.nsf.gov/sbe/bcs/pac/nmbs/nmbs.jsp>.
- [69] I. Lobach, M. Borland, G. Fystro, A. Sannibale, and Y. Sun, Machine learning for predicting power supply trips in storage rings, in *Proceedings of the 2022 North American Particle Accelerator Conference (NAPAC), Albuquerque, New Mexico* (unpublished).
- [70] C. Tennant, A. Carpenter, T. Powers, A. S. Solopova, L. Vidyaratne, and K. Iftekharuddin, *Phys. Rev. Accel. Beams* **23**, 114601 (2020).
- [71] M. M. Rahman, K. Iftekharuddin, A. Carpenter, T. McGuckin, C. Tennant, and L. Vidyaratne, Real-time cavity fault prediction in cebaf using deep learning, in *Proceedings of NAPAC 2022, Albuquerque, NM* (2022).
- [72] O. Ronneberger, P. Fischer, and T. Brox, U-net: Convolutional networks for biomedical image segmentation, in *Proceedings of International Conference on Medical Image Computing and Computer-Assisted Intervention* (Springer, New York, 2015), pp. 234–241.
- [73] L. T. D. Coyle, Machine learning applications for hadron colliders: LHC lifetime optimization, Ph.D. thesis, Grenoble INP, 2018.
- [74] E. Krymova, G. Obozinski, M. Schenk, L. Coyle, and T. Pieloni, Data-driven modeling of beam loss in the LHC, [arXiv:2208.08935](https://arxiv.org/abs/2208.08935).
- [75] A. Scheinker *et al.*, Model independent beam tuning, in *Proceedings of the 4th International Particle Accelerator Conference, IPAC-2013, Shanghai, China, 2013* (JACoW, Shanghai, China, 2013), pp. 12–17.
- [76] A. Scheinker and D. Scheinker, *Int. J. Robust Nonlinear Control* **28**, 568 (2018).
- [77] A. Scheinker, *Information* **12**, 161 (2021).
- [78] F. Cropp, P. Musumeci, A. Scheinker, D. Filippetto, A. Gilardi, S. Paiagua, and D. Wang, Toward machine learning-based adaptive control and global feedback for compact accelerators, in *Proceedings of 13th International Particle Accelerator Conference (IPAC'22), Bangkok, Thailand* (JACoW, Geneva, Switzerland, 2022).
- [79] D. Kafkes and J. St John, *Data* **6**, 42 (2021).
- [80] Z. Kang, C. Catal, and B. Tekinerdogan, *Sensors* **21**, 932 (2021).