Online Neural Dynamics Forecasting for power system security

Karacelebi, Mert; Cremer, Jochen L.

DOI
10.1016/j.ijepes.2025.110566

Publication date
2025

Document Version
Final published version

Published in
International Journal of Electrical Power and Energy Systems

Citation (APA)
Karacelebi, M., & Cremer, J. L. (2025). Online Neural Dynamics Forecasting for power system security. *International Journal of Electrical Power and Energy Systems*, *167*, Article 110566. https://doi.org/10.1016/j.ijepes.2025.110566

Electrical Power and Energy Systems 167 (2025) 110566

Contents lists available at ScienceDirect

International Journal of Electrical Power and Energy Systems

journal homepage: www.elsevier.com/locate/ijepes

Online Neural Dynamics Forecasting for power system security

Mert Karacelebi *, Jochen L. Cremer

Electrical Sustainable Energy, Delft University of Technology, Delft, 2628CD, South Holland, Netherlands

ARTICLE INFO

Keywords:
Power system dynamics
Neural Ordinary Differential Equations
Power system monitoring
Power system dynamic security
Power system faults

ABSTRACT

The increase in variable renewable energy sources has brought about significant changes in power system dynamics, mainly due to the widespread adoption of power electronics and nonlinear controllers. The resulting complex system dynamics and the unpredictable nature of disturbances pose substantial challenges for real-time dynamic security assessment (DSA). Machine learning (ML) methods offer advantages in terms of computational speed compared to numerical methods and simulators. Offline-trained ML models, however, are limited by their training domain; e.g., they cannot easily consider various grid topologies and data changes. Neural Ordinary Differential Equations (NODEs) leverage the integration of neural networks and ODE solvers to enable continuous-time dynamic trajectory predictions from time series data, resolving the limitation on topological and data changes. This paper introduces the Online Neural Dynamics Forecaster (ONDF) workflow, designed to monitor and assess system security in real-time using multiple NODEs trained solely with local post-fault measurements. Through several case studies, we compare the regression and DSA classification capabilities of ONDF with various ML models. Our findings demonstrate that ONDF provides a novel and scalable approach for system operators to make informed decisions and apply corrective control actions based on predicted dynamics.

1. Introduction

Ensuring the electricity supply is a vital and challenging task for system operators. Although power systems are equipped with hierarchical protection and control systems, rare severe disturbances can disrupt the electricity supply. Any disruption has drastic consequences for society, the economy, and industry. To avoid such scenarios, system operators simulate the system's response to severe disturbances using dynamic simulation models. Most of these simulations consider the security of the system losing only one component of the system (N-1); however, scenarios may cause changes in the network topology, and cascading failures are similarly dangerous. Further, the massive integration of inverter-based generation resources (IBRs) increases the complexity of the dynamics [1], and conventional methods require new dynamic models, stability theory, and controllers [2,3]. Large-scale realistic simulations take long and are not feasible to conduct in near real-time, as the number of dynamic states is enormous. However, decentralized equivalents of the main contributors to dynamics can also inform about dynamics [4].

Machine learning (ML) approaches can learn from data such as equivalent models (also called surrogates or proxies) relating the pre-fault state of the power system to the post-fault security. The training data are often synthetically generated using dynamic simulations that mathematically model the system response. Although the generated training data often considers a wide variety of operating conditions, the data are often limited in the variety of network topology and disturbances and do not consider distance- or system-level protection schemes and cascading failures. The curse of dimensionality ultimately limits the realism of the studies and generated data and, therefore, limits the generalization of these trained ML models. As demonstrated over the last decades [5], several models such as decision trees [6], artificial neural networks (ANN), and support vector machines (SVM) [7] can predict the dynamic post-fault security of a system using the two, pre-fault operating conditions and system measurements. However, ML models can only generalize a security boundary for the disturbances and conditions that are 'similar' to the ones in the training data, and, in response, recently, the challenge was mitigated to generate better and better training databases [8–10]. However, the severity, type, and location of a disturbance are highly unpredictable, and simultaneously sampling a large number of disturbances and operating conditions is not practical for large-scale systems.

Post-disturbance time series data contains valuable information for system operators to assess system security in near real time. System operators can assess transient stability by comparing the violations of predefined security limits using post-disturbance voltage signals [11].

* Corresponding author.
E-mail addresses: m.karacelebi@tudelft.nl (M. Karacelebi), J.L.Cremer@tudelft.nl (J.L. Cremer).

https://doi.org/10.1016/j.ijepes.2025.110566
Received 5 July 2024; Received in revised form 11 January 2025; Accepted 24 February 2025
Available online 24 March 2025

Alternative to security indices, models like extreme learning machines can predict the security label using the centralized post-disturbance data sampled from different time instances as separate features [12]. More advanced centralized machine learning models process the time sequence data collected from all dynamic elements to capture temporal relationships between features. For example, Gated Recurrent Units (GRUs) construct an information stream from past to future hidden states. GRUs use centralized measurements from fault clarification instances to asses first-swing transient stability [13]. Long short term memory (LSTM) model combined with the convolutional neural network (CNN) model can accurately classify transient security using centralized sequential voltage, phase angle, power flow measurements, and static fault information as location, duration, and type [14]. Besides security labels, LSTM models can also predict possible cascading failures in the system [15]. Although the temporal relationship is captured by preserving the sequence data, topological changes in the system alter the system's dynamics, hence the change in the security boundary constructed by ML algorithms. Combining graph attention networks and GRUs provides high accuracy for voltage and transient stability [16] under changing system topology. The mentioned centralized solutions could face generalization problems following changes in the dynamic system model, such as unseen operating conditions, topologies, disturbances, and future system expansions. These complex ML algorithms often require full system observability and large training datasets that are computationally expensive to generate using simulation tools. However, the computational requirements for generating such an extensive dataset and training large ML models remain challenging for system operators. The evolving nature of the system dynamics requires further methods like transfer learning [17] or an update (retraining) of the model [18] to preserve the quality of the prediction. Furthermore, system operators often receive security labels or indices from these ML algorithms where a change in security criteria requires the recomputation of labels and retraining of the model. Still, the information regarding the post-disturbance system states would be limited without dynamics forecasting.

An alternative approach takes the first part of a system's post-disturbance dynamics as input to the model to forecast the future dynamic system states as output, similar to time-series forecasting (e.g., autoregressive integrated moving averages). However, several challenges exist in the centralized forecast approach. This approach requires very fast communication of the measured actual dynamics to the centralized control center, where the model can predict the dynamics further ahead. And, even if communications are fast, predicting the dynamics may be inaccurate as discrete system events may occur, which is notoriously difficult for ML models to learn. These discrete system events may involve multiple failures occurring in cascades and changes in the topology from the active protection systems. Nonetheless, this type of approach shows promising success. For example, the Koopman mode analysis forecasts the rotor angles and speed in the discrete form to detect transient instability [19]. Nonlinear autoregressive and LSTM models use sequential, discretized, previous measurements as input to predict post-fault states, such as the system frequency of the next time step [20–22]. However, these discretizations in time may miss fast-changing dynamics that occur in shorter time steps. Simulations consider these fast-changing dynamics through adaptive time steps. Considering these fast-changing dynamics becomes particularly important with inverter-based grids. Beyond learning in the time domain, the multi-dimensional Fourier transformation trains in the frequency domain [23]. Graph Neural Networks (GNNs) consider sequential states and process topological and time-domain information [24]. However, as topological changes alter the system's dynamic characteristics, they need to be sampled, ultimately addressing but not solving the curse of dimensionality. For example, the training dataset for the IEEE 39 bus system contains 20,000 samples [24]. This curse of dimensionality, requiring explosive numbers of samples and more dimensions the model needs to generalize, challenges these centralized ML workflows, as all measurements from all generation units in the system are needed to represent the dynamics accurately.

Recently, a new time domain model was proposed, the neural ordinary differential equation (NODE) model [25]. NODE can identify the unknown ODE system equations [26] and be used for the continuous-time reversible generative models [27]. NODEs construct an ODE system where a deep residual neural network (RNN) parameterizes the dynamics of the hidden state instead of actual states. Learning dynamics instead of states, the NODEs can forecast longer time horizons using a short duration of data [28]. Capability of learning dynamics for NODEs further expanded with combining linear and nonlinear neural network layers to accurately learn long-time trajectories from chaotic data [29]. The ordinary differential equation (ODE) solver with reverse mode automatic differentiation efficiently computes the scalar gradients of the loss relative to the model parameters. NODE model is a continuous function that represents dynamic systems while considering the underlying physical flows, unlike the discretized form in RNN [30]. The NODE formulation allows approximation of the dynamic simulation results of an ODE system purely from the simulation data [31]. The continuous function representation is also suitable for control tasks using the forecasted trajectories [32]. Having an ODE solver for learning and prediction provides additional flexibility against noisy or missing data. The noisy data could lead to false predictions in sequential models, especially when noise varies between sample points or features. However, NODE models can still identify underlying dynamics even if data is irregularly collected and contains a certain amount of noise [33]. That is why the application scope of the NODE has been expanded in recent years.

NODEs were investigated for power systems (and power system dynamics). For example, continuous output can eliminate data inconsistencies between digital twins and measurements for forecasting in power systems [34]. NODE estimates the unknown parameters dynamic model of the generation units from simulations to improve mathematical models [35] and replace the analytical models of various system controllers [36] for power system identification. NODE can predict rotor angles of synchronous machines even under noisy training data and makes NODE more suitable for grid applications where measurements could contain certain noise [37]. NODE's learning capability of system dynamics is not limited to the transmission systems. The Bayesian inference model can provide microgrid dynamic model parameters' probability distribution for NODE that predicts microgrid dynamics accurately [38]. The Latent NODE models can integrate and impute multi-time scale smart distribution grid measurements, which are irregularly sampled and contain measurement noise [39]. Inspired by these successes, this paper aims to address the dimensionality challenges in real-time, ML-based security assessments by proposing, for the first time, NODEs for post-fault prediction of dynamics.

This paper proposes the *Online Neural Dynamics Forecaster* (ONDF), a near-real-time decentralized workflow that forecasts the dynamics of the measured location to identify the system's security. ONDF directly predicts the future dynamic time trajectory (model output) from the post-fault local power dynamics (model input). Meanwhile, the ONDF workflow adapts to 'any' system conditions and disturbances. Before the operation, the ONDF model considers multiple, pre-trained basis NODE models that maximize the diversity of the pre-trained conditions using approximate entropy. This approach is fault-agnostic and generalizes to many faults and operating conditions, marking a significant step forward in addressing a long-existing limitation of ML-based approaches as the curse of dimensionality. The proposed real-time security evaluation combines decentralized NODE forecasts of available measurements at the control center. As this evaluation is decentralized, it scales to large

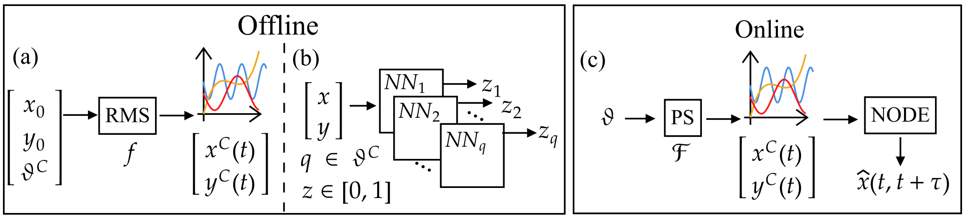

Fig. 1. Approximating power system (PS) dynamics in offline and online times using simulations (a), artificial neural networks (b), proposed NODEs (c).

power networks using only a few measurements at large generators. This paper contributes with

1. ONDF to forecast real-time local dynamics with NODE models. We propose using NODE models trained in near real-time with the moving window approach to approximate changing dynamics from cascading events. Our approach is a continuous-time-domain formulation and efficiently trains with minimal measurements.
2. Decentralized approach predicts the dynamics for a centralized security assessment. Multiple NODEs learn the local dynamics of the selected measurements without requiring full system observability; hence, ONDF is scalable to larger networks. The approach is significantly faster than any centralized approach, as the number of dynamic states is significantly lower.

Our case studies consider the 400 kV Transmission example with 46 buses in DigSilent PowerFactory to provide evidence of the benefits and limitations. Further case studies on the IEEE 9-bus and Texas 2000-bus systems provide related evidence in terms of scalability. The case study model considered cascading failures, distance, and system protection systems such as under/over frequency/voltage protection. Studies demonstrate the proposed model outperforms conventional ML models and approximate ODE systems. Studies investigate the NODE model for estimating the post-fault dynamic security of the power system. Studies illustrate the functionality of the NODES, e.g., the approximation of multiple stages during training, the moving-window training workflow, and the benefits of decentralization. Finally, the computational studies verify the suitability of our proposed modifications of NODEs, making them suitable for near real-time training.

2. Online power system dynamics forecasting

$x^*(t)$ are the complex, nonlinear dynamic states of the system dynamics $\mathcal{F}$ following a disturbance $\vartheta \in \Omega^{\vartheta}$. Fig. 1 shows three approaches for estimating dynamic security based on different models. The first approach simulates f with the Root Mean Square (RMS) method to compute $x(t)$ as an initial value (x_0, y_0) problem involving many differential–algebraic equations (DAE)s. The second approach relies on ML models (e.g., ANNs) where each ANN_z model predicts the binary label z for the contingency considered $q \in \vartheta^C$. Despite the progress in learning better models, there are several open challenges for using ANN online due to the unpredictable nature of the disturbance and the error between the simulation and measurements that can lead to inaccurate predictions. Updating the DAE system alters the $x(t)$ for the same (x_0, y_0) and q. Therefore, corresponding z changes and ANN requires another expensive offline data generation process.

This paper uses the measured system dynamics $\mathcal{F}$ to train NODE models sequentially, considering the evolving post-fault dynamics. The learned NODE has a continuous output $\hat{x}(t_k, t_k + \tau)$ from initial time t_k to $t_k + \tau$ where k corresponds to the investigated time window. Although NODE models are pre-trained with simulated dynamics f, retraining with actual dynamics X_{train} collected from the system $\mathcal{F}$ reduces the impact of the model error. The proposed NODE moves the time windows for training; hence, for the first time, this model can consider the dynamics caused by cascading failures or control actions. Cascading and post-fault actions are typically not considered as they would require extremely many training samples from simulations.

2.1. Neural networks to predict security

An ANN model transforms features $[x, y]$ from the input to output layer that contains estimated label z, for example, dynamic security $z = 0$ or insecurity $z = 1$. Transformations are 'neuron' functions aggregated in layers. Each neuron function with trainable parameters θ multiplies the output from the previous layer's neurons. The gradient descent method fits the model parameters θ to extensive correct combinations $([x, y], v)$ as shown in Fig. 1(b). Using training data, the backward propagation computes the gradients of each neuron concerning the scalar-valued loss function $\frac{\partial L}{\partial \theta}$ following the chain rule. Then, gradient descent iteratively updates θ in the direction of the negative gradient.

2.2. Neural ordinary differential equations (NODEs)

Residual neural network models apply nonlinear sequential transformations $h(x(t), \theta)$ to the d dimensional hidden state $x(t) \in \mathfrak{R}^d$ where $t \in [0, 1, 2, \dots, \tau]$.

$$x(t+1) = x(t) + h(x(t), \theta_t) \tag{1}$$

is equivalent to Euler's method when t becomes small with an increasing number of layers in the network.

The neural network $h(x(t), \theta, t)$ approximates the hidden state dynamics $\frac{dx}{dt}$ instead of the state itself x_t. The ODE solver computes

$$\begin{aligned} x(t_0 + \tau) &= x_0 + \int_{t_0}^{t_0+\tau} h\Big(x(t), \theta, t\Big) dt \\ &= \text{ODESolve}\Big(x_0, h, t, (t_0, t_0 + \tau), \theta\Big) \end{aligned} \tag{2}$$

from the ANN instead of the known DAE system. The NODE model learns the hidden dynamics and constructs a continuous function from the discrete data, shown in Fig. 1(c). However, applying the chain rule to an ODE solver is computationally resource-intensive and would not approach the speed for near real-time DSA; therefore, the NODE training uses the adjoint sensitivity method [25]. The adjoint states $a(t)$ are equivalent to the derivative of loss to state $\frac{\partial L}{\partial x(t)}$. Another ODE can define the adjoint dynamics

$$\frac{\mathrm{d}a(t)}{\mathrm{d}t} = -a(t)^T \frac{\partial h(x(t), t, \theta)}{\partial x} \tag{3}$$

and the reverse mode automatic differentiation computes the associated vector Jacobian pairs. The loss gradient

$$\frac{\mathrm{d}L}{\mathrm{d}\theta} = -\int_{t_0+\tau}^{t_0} a(t)^T \frac{\partial h(x(t), t, \theta)}{\partial \theta} \mathrm{d}t \tag{4}$$

uses an integral reverse in time, and the ODE solver computes the model parameter updates from the adjoint system Eq. (3).

3. Online neural dynamics forecaster (ONDF)

Our proposed ONDF workflows (Fig. 2) perform the dynamics forecasting in real-time against the experienced disturbance. The online workflow takes PMU measurements and pre-trained NODE models from the offline workflow as inputs and outputs of the forecasted dynamics.

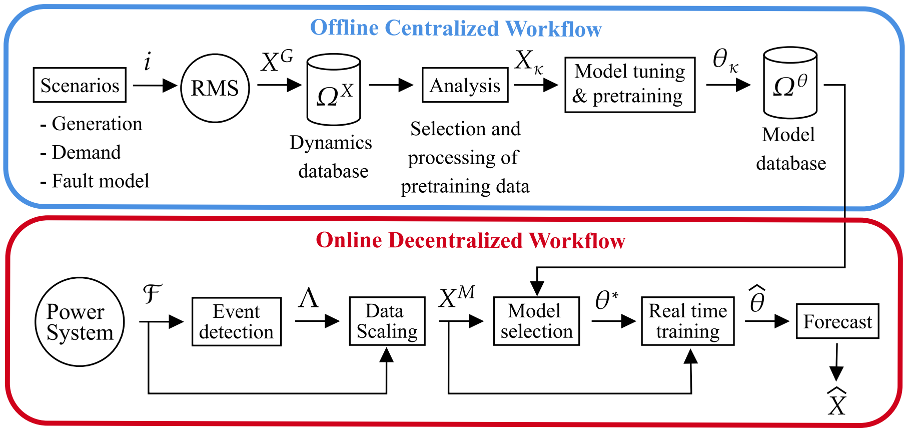

Fig. 2. ONDF workflows with RMS and the actual power system.

3.1. Offline centralized workflow

The offline workflow (details in Section 3.3) performs dynamic simulations to create the database Ω^X for the NODE pretraining. Creating this database is a common computational challenge requiring representative sampling of many feasible operating conditions [40]. However, the proposed approach does not suffer from this challenge as the purpose of the database is to select a few selected samples for pre-training to accelerate the online training. The database Ω^X has sets of dynamic states $X_j^G \subseteq X$ with X^{syn} from selected generators j. The workflow selects generators with large power ratings in their region, as their dynamics represent the system's security. This selection is similar to the limited available dynamic measurements in real-life power systems. Centralized data-driven approaches often rely on time-series measurement from all (or many) buses in the system, which the simulation environment can satisfy, but not the online measurements. The approximate entropy (ApEn) analyzes the complexity $x(t)$ by quantifying the 'unpredictability of fluctuations' [41]. Subsequently, the affinity propagation (AP) algorithm clusters the data based on ApEn. The workflow fits NODE models on the cluster centroids κ with X_κ. Then, the workflow adds the model parameters $\theta_\kappa \rightarrow \Omega^\theta$ to the pre-trained models Ω^θ.

The dynamics database in Fig. 2 requires RMS simulations' dynamic trajectories $X_{j,v}^G$ of scenario v. Each scenario v combines different initial conditions $[x_0, y_0] \in \mathcal{N}_{oc}$ and fault models $\vartheta \in \mathcal{N}_f$. The set of operating conditions $\mathcal{N}_{oc}$ contains samples from historical generation and demand profiles to alter the initial conditions (x_0, y_0). The fault model set $\mathcal{N}_f$ includes faults with varying impedances and locations $\mathcal{Z}_i^F$ to diversify the dynamics in the database Ω^X. Historical recordings can supplement simulations Ω^X in power systems that do not have accurate dynamic models. Inaccurate dynamic models can lead to centralized data-driven approaches becoming unreliable in real-life situations, but incorporating an online workflow helps bridge the gap between the mathematical model and reality.

3.2. Online decentralized workflow

The decentralized workflow, shown in Fig. 2, starts by detecting an event related to the change in the states of the monitored power system X^M. Event detection $\mathcal{E}(X^M[t], X^M[t-1])$ activates the signal $\Lambda \in \{0,1\}$ after sudden changes in system dynamics,

$$\Lambda = \begin{cases} 1, & \text{if} \quad \|X^M[t] - X^M[t-1]\| \geq \gamma \\ 0, & \quad \text{otherwise.} \end{cases} \tag{5}$$

Λ might activate responding to simple dynamic events such as line switching; hence, the trigger threshold value γ needs attention.

Subsequently, the workflow selects the pre-trained model M_0 with the lowest loss to post-fault system measurements X^M during the predefined protection delay t_f. X^M depends on the system loading, generation dispatch, topology, and dynamic event properties. Therefore, any change in system topology from the base topology will be captured by retraining the M_0. The protection system clears the disturbance during the delay, and the workflow computes the loss

$$\min_{\theta_\kappa \in \Omega^\theta} \quad L(\hat{X}, X^M[t^M]) \tag{6}$$

$$\text{s.t.} \quad \hat{X} = \text{ODESolve}(X^{M_0}, M_0, t^M, \theta_\kappa) \tag{7}$$

$$t^M = [t_0 + t_f, t_1] \tag{8}$$

using the initial condition X^{M_0} for all models Ω^θ in time span t^M. The workflow selects the best model parameters θ^* for the initial model M_0. Although the initial model's accuracy will increase with training, system operators can already analyze this initial prediction $\hat{X}$ to understand the approximated system response.

In (near) real-time, the workflow's Algorithm 1 starts real-time training of the workflow with the selected model M_0 at t_1 with initial parameters θ^* using the monitored states X^M. A moving window replaces X^M sequentially with the latest measurements to efficiently retrain for each time window k until the system converges ($\Lambda = 0$). The Algorithm 1 collects data X^{train} in batches (line 6) and trains consecutively a NODE model M_k (line 7). The size of the training data $|X^{train}|$ expands until the predefined training time span $t_k + \tau^{train}$ is achieved. Training during the data collection increases the efficiency and feasibility of learning in near real-time. The Algorithm 1 continues training (line 11) while collecting the validation data X^{val} (line 13). The final parameters $\hat{\theta}_k$ corresponds to the minimum validation loss $L(\hat{X}_1, X^{val})$ (line 15). The algorithm shifts the time window k (line 20) and starts training with the new data until $\Lambda = 0$ (line 22), indicating the system dynamics converging to a steady state. Fig. 3 illustrates the algorithm by showing how time spans shift with the size of the training data.

In Fig. 4, the decentralized workflow trains three NODEs with corresponding monitored generators.TSO receives the final set of parameters and corresponding initial conditions $[\hat{\theta}, X_0]$ after each time window. Decentralized training of NODEs ensures capturing local dynamics against any topological changes in the system where each forecast approximates the dynamics under changed topology. This property enables the ONDF to generalize to unseen system dynamics as each NODE's final parameters $\hat{\theta}$ are unique to the measured dynamics X^M.TSO computes the forecasted trajectories $\hat{X}$ and then conducts the custom real-time DSA function $\Psi(\hat{X})$ to activate the system-level corrective control actions. The possible control actions could include over/under frequency or voltage shedding schemes, but the properties and impacts of control actions are out of the scope of this paper.

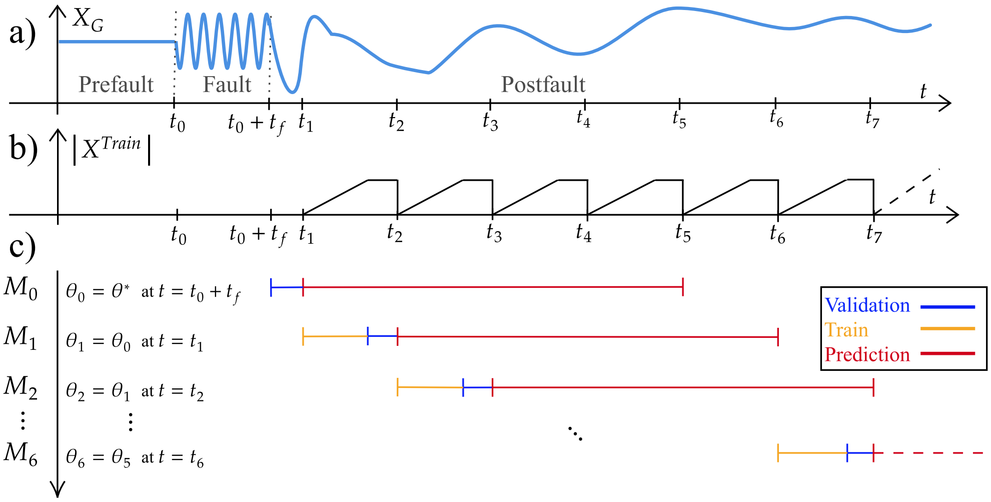

Fig. 3. ONDF online training illustration, the state variable trajectory following a disturbance (a), the size of the training data (b), model time spans in a moving window (c).

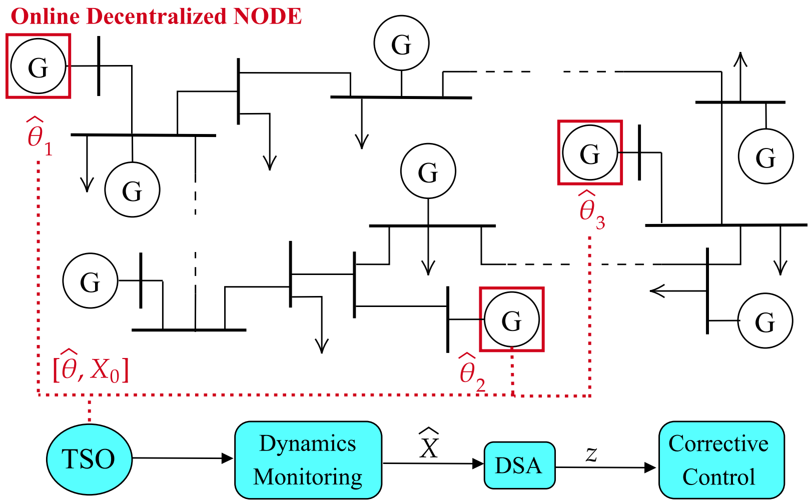

Fig. 4. Decentralized workflow and centralized security assessment.

3.3. Data analysis and selection of pretraining data

The analysis of the complexity of trajectories in Fig. 2 selects representative dynamic trajectories $X_\kappa \in \Omega^X$ for model tuning and pretraining. The analysis scales the dynamic states based on their machine ratings and nominal operating values. The selection of pretraining data aims to find simulation results with various regularity. The ApEn estimates this rate of regularity in time data series without prior knowledge about the source [42]. Each row vector x of $X^G_{j,v}$ contains the equally spaced time sequence data of generator j and scenario v. The subsequence $\zeta[e] = [x[e], x[e+1], \dots, x[e+m-1]]$ starts at time point e with window size $m \in \mathbb{Z}^+$ where $m \le N$.

$$C(m,e,\mu) = \frac{\text{count}(\varrho, d(\zeta[e],\zeta[\varrho]) \le \mu)}{N-m+1}, \tag{9}$$

counts instances where the maximum distance

$$d(\zeta[e],\zeta[\varrho]) = \max_{\rho}(|x[e+\rho-1] - x[\varrho+\rho-1]|), \tag{10}$$

between two subsequences e and ϱ is lower than the threshold μ for each ϱ, ρ where $e, \varrho, \rho = \{1, 2, \dots, N-m+1\}$. Then,

$$\text{ApEn}(m,\mu,N,x^G) = \phi(m,\mu) - \phi(m+1,\mu) \tag{11}$$

uses the average of the logarithmic sum

$$\phi(m,\mu) = \frac{1}{N-m+1} \sum_{e=1}^{N-m+1} \log C(m,e,\mu). \tag{12}$$

The ApEn is a normalized quantity for samples in Ω^X. The elements of the row vector $\mathcal{W}_v$ contain summed ApEn values corresponding to the sample v for the clustering algorithm.

The AP algorithm selects representative samples based on message communications. The algorithm clusters the samples using the availability $\mathcal{A}(\alpha,\beta)$ and responsibility $\eta(\alpha,\beta)$,

$$\mathcal{A}(\alpha,\beta)_{\alpha\neq\beta} \leftarrow \min\{0, \eta(\beta,\beta) + \sum_{\alpha'\neq\beta,\alpha} \max\{0, \eta(\alpha,\beta)\}\} \tag{13}$$

$$\eta(\alpha,\beta) \leftarrow s(\alpha,\beta) - \max_{\beta',\beta'\neq\beta} \left\{\mathcal{A}(\alpha,\beta') + s(\alpha,\beta')\right\} \tag{14}$$

messages between samples where $\alpha, \beta \in \{v\}$ [43]. The similarity, the negative Euclidean distance $s(\alpha,\beta) = -\left\|\mathcal{W}_\alpha - \mathcal{W}_\beta\right\|^2$ computes these two messages. The responsibility $\eta(\alpha,\beta)$ message from α to β shows the accumulated evidence of how well β acts as an exemplar point for α. The availability $a(\alpha,\beta)$ message from β to α shows the accumulated evidence of α to choose sample β as its exemplar. The algorithm

Algorithm 1: Real-time training of ONDF Workflow

```
Input: $X$, $t$, $\Lambda$, $t_1$, $\theta^*$
Output: $\hat{X}_k$, $\hat{\theta}_k$
$k = 1$, $\theta_1 = \theta^*$
while $\Lambda = 1$ do
    $X^M \leftarrow X$
    $t^M \leftarrow t$
    if $t_k < t < t_k + \tau^{train}$ then
        $X^{train} = X^M[t_k : t]$; $t^{train} = t^M[t_k : t]$
        $\theta_k = \text{Train}(M_k, \theta_k, X^{train}, t^{train})$
    end
    else if $t_k + \tau^{train} < t < t_k + \tau^{val}$ then
        $\hat{X}^{prev}_k = \text{ODESolve}(X^M[t_k], M_k, \tau^{val}, \theta_k)$
        $\theta_k = \text{Train}(M_k, \theta_k, X^{train}, t^{train})$
        $\hat{X}_k = \text{ODESolve}(X^M[t_k], M_k, \tau^{val}, \theta_k)$
        $X^{val} = X^M[t_k : t]$; $t^{val} = t^M[t_k : t]$
        if $L(\hat{X}_k, X^{val}) < L(\hat{X}^{prev}_k, X^{val})$ then
            $\hat{\theta}_k = \theta_k$
        end
    end
    else if $t_k + \tau^{val} \leq t$ then
        $\hat{X}_k = \text{ODESolve}(X^M[t_k], M_k, \tau^{pred}, \hat{\theta}_k)$
        $\theta_{k+1} = \theta_k$  $t_{k+1} = t$ , $k = k + 1$
    end
    $\Lambda = \mathcal{E}(X^M_t, X^M_{t-1})$
end
```

initializes from zero

$$\mathcal{A}(\beta, \beta) \leftarrow \sum_{\alpha' \neq \beta} \max\left\{0, \eta(\alpha', \beta)\right\} \tag{15}$$

and iteratively updates the two messages

$$\eta_{I+1}(\alpha, \beta) = \lambda \eta_I(\alpha, \beta) + (1 - \lambda)\eta_{I+1}(\alpha, \beta) \tag{16}$$

$$\mathcal{A}_{I+1}(\alpha, \beta) = \lambda \mathcal{A}_I(\alpha, \beta) + (1 - \lambda)\mathcal{A}_{I+1}(\alpha, \beta) \tag{17}$$

until identified exemplar points do not change for a predefined number of iterations. λ controls the damping of message updates at iteration I.

As the number of clusters might be large, the centroids can be further clustered with AP repeatedly. The set X^G_κ considers dynamic trajectories of centroid samples κ selected by AP to train NODEs.

3.4. NODE tuning and pretraining for online workflow

The NODE model hyper-parameter tuning and pretraining prepare the NODEs for the online workflow (see Fig. 2). Any subset of training data samples X^G_κ can tune hyper-parameters of the NODE models $M_\kappa \in \Omega^\theta$. Various algorithms can tune hyper-parameters [44]. Using a grid search approach, this paper tunes a single hyper-parameter from the initial base model at a time. Examples of hyperparameters are the ANN architecture (number of hidden layers, neurons, and activation functions) and the learning rate. The NODE pre-training considers data $X^{train}_\kappa = X^G_\kappa[t^{train}_\kappa]$ starting from the time point t_1 with τ^{train} time span so that the temporal order is consistent. The function

$$\text{Train}(M_\kappa, \theta_\kappa, X^{train}_\kappa, t^{train}_\kappa) \tag{18}$$

trains one epoch of the NODE model M_κ considering ANN parameters θ_κ, the data X^{train}_κ and the time instances t^{train}_κ. To learn the best hyper-parameters, the training approach repeats N_{epoch} times the Train function. Subsequently,

$$\hat{X}_\kappa = \text{ODESolve}\left(X^{train}_\kappa[t_1], M_\kappa, \tau, \theta_\kappa\right) \tag{19}$$

computes the solution $\hat{X}$ from the initial condition $X^{train}_\kappa[t_1]$ with the time span τ. The ODE solvers' computing time depends on the size, stiffness, and complexity of the ODE problem. Tuning the error tolerance can shorten the solving time but may result in lower accuracy.

The initial model database Ω^θ contains model parameters θ_κ with the lowest loss $L(X^G_\kappa, \hat{X}_\kappa)$ for the prediction horizon τ that covers the entire trajectory. If there are results from a new disturbance or changed system conditions, the approach can train a new NODE model $M_{\kappa+1}$ to update the database. Over time, the approach is adaptive, adding new system dynamics to the database with high variability.

3.5. Power system dynamics

The power system dynamics in DAE form are

$$\dot{x} = f(x, y, p, t), \quad x(t_0) = x_0 \tag{20a}$$

$$0 = g(x, y, p, t), \quad y(t_0) = y_0, \tag{20b}$$

where $f(x, y, p, t)$ are the ODEs and $g(x, y, p, t)$ the algebraic equations. The DAE is an initial value problem with dynamic x and algebraic y states initialized at time t_0. p are system parameters, e.g., inertia or controller gains. The algebraic nonlinear power flow equations compute the net active P_i and reactive power Q_i injections at each bus $i \in \mathcal{N}^b$ using network conductance and susceptance matrices $G, B \in \Re^{\mathcal{N}^b \times \mathcal{N}^b}$.

The differential equations $f(x, y, p, t)$ are ODEs with states x and derivatives $\dot{x}$ considering synchronous and converter-based generators', each having internal and external states. The synchronous machines consider the states $X^{Syn} = [\delta, \omega, P^E, Q^E, |\bar{V}|]$ where the states are rotor angle δ, rotor speed ω, active power injection P^E, reactive power injection Q^E and terminal voltage $\bar{V}$. The equations are

$$\dot{\delta} = \omega^B(\omega - \omega^S) \tag{21a}$$

$$\dot{\omega} = \frac{1}{2H}\left(T^M - T^E - D(\omega - \omega^S)\right) \tag{21b}$$

$$T^E \approx P^E \tag{21c}$$

$$P^E + jQ^E = (e^d i^d + e^q i^q) + j(e^q i^d - e^d i^q) \tag{21d}$$

$$\bar{V} = e^d + je^q, \tag{21e}$$

with the base synchronous frequency ω^B, and the synchronous reference speed ω^S. The deviation from the reference considers mechanical T^M and electrical T^E torques. H is the inertia constant. The stator voltages e^d, e^q, and currents i^d, i^q in the "dq" reference frame calculate the rest of the states. Dynamic models have many more states and equations from machine regulators such as turbine governors, automatic voltage regulators, and power system stabilizers [45]. The ONDF workflow only learns the selected dynamic states X^{Syn} from the full-order machine model instead of learning the large dimensionality of the actual physical model. The dimension of X^{Syn} is constant for each NODE model. Although the reduction in the order is essential to achieve feasible training of NODEs during the post-fault, it introduces approximation errors. However, the produced error would not cause misclassification of the security label as forecasted dynamics exhibit lower complexity while capturing the trend of the observed dynamics.

The converter-based generation units have various dynamic models and controllers where various dynamic states contain security information. Although some of the selected states are not measurable, NODE can learn from the output of the dynamic state estimation. The time scales of inverter-based generators are shorter than those of synchronous machines, so the NODE can train with shorter training data as dynamics will be observable.

The dynamic state estimation is out of the scope of this paper. Alternatively, ONDF could learn PMUs from transmission lines to monitor voltage stability, thermal overloadings, or power oscillations.

3.6. Simulating representative faults

Simulating the post-fault dynamics $\vartheta = [P_i^F, Q_i^F, u(t)]$ requires a fault model as

$$P_i^F + jQ_i^F = -u(t) g_i^F {V_i^F}^2 + j\left(-u(t) b_i^F {V_i^F}^2\right) \tag{22a}$$

$$u(t) = \begin{cases} 1, & t_f \leq t \leq t_c + t_f \\ 0, & t < t_f,\ t > t_c + t_f. \end{cases} \tag{22b}$$

The three-phase fault with impedance $\mathcal{Z}_i^F = g_i^F + jb_i^F$ causes fault current that causes a power flow P_i^F and Q_i^F at voltage magnitude V_i^F, at bus i. The simulation activates the fault $u(t)$ at time $t = t_f$, and the protection system clears the fault at $t = t_f + t_c$. As the probability of simultaneous faults at different buses is extremely low, g^F and b^F are assumed to be zero except at the selected fault bus.

Simulating all possible fault models ϑ with varying parameters (g_i^F, b_i^F) is not practical. Therefore, the approach only simulates critical faults ϑ^C to compute representative dynamic system responses. Numerical methods compute the dynamic trajectory $x(t)$ using the system Eq. (20) and disturbance Eq. (22). Euler method explicitly approximates with a non-infinitesimal time step Δt.

$$\dot{x} = \frac{dx}{dt} \approx \frac{x(t + \Delta t) - x(t)}{\Delta t} \tag{23a}$$

$$x(t + \Delta t) = x(t) + f(x, y, p, t) \tag{23b}$$

$$y(t + \Delta t) = g(x, y, p, t + \Delta t) \tag{23c}$$

3.7. ONDF application: Online security assessment

The TSO can assess the system's dynamic security following a severe disturbance using ONDF forecast results with the selected generator's measurements. The following function

$$\Psi(X) = \Psi(\delta, \omega, P^E, Q^E, |V|) \tag{24}$$

returns the binary security label $z = 0$ (secure) or $z = 1$ (insecure), based on the comparing forecast results with the corresponding security limits. The detailed definition of Ψ depends on the TSO preferences, and a generic function is assumed. For example, one way to assess static security is by computing the thermal limit violations of the machines based on the P^E and Q^E forecasts. Voltage $|V|$ and frequency ω protection systems allow the operation for a limited time γ^t [s] under emergency conditions for synchronous machines. The deviation of angles δ_j from the center of inertia (COI) rotor angle δ^{COI} can detect out-of-synchronism conditions. More specifically, the security criteria is defined as $|\delta_j - \delta^{COI}| \geq \delta^{max}$, where $\delta_j \in \hat{X}$.

If any predicted generator's angle deviates more than the predefined limit δ^{max} from the COI reference, the system becomes transient insecure. There, the COI is $\delta^{COI} = \frac{\sum_j H_j \delta_j}{\sum_j H_j}$ is weighted by machines' inertia constants H_j. Unlike collecting each machine's rotor angle, this paper computes the COI angle using only measured generators. Any deviation from the limited generation subset still reveals out-of-synchronism conditions. Moreover, the NODE provides continuous predictions starting from initial conditions with the desired time points from the ODE solver. Therefore, any communication delays do not create any problems for the security assessments in the control center. Moreover, any function that takes time series data for the security assessment can use the ONDF workflow for predicted states. Ψ includes all security limits described above in this paper to assess online dynamic security.

4. Case study

The studies considered the 400 kV Transmission System (TS) example of [46], the Texas 2000-bus grid, and the IEEE 9- and 39-bus systems. The TS system has 46 buses and nine transmission lines. The system is divided into four zones with 13 static loads (PQ), 18 synchronous, and seven inverter-based generators (onshore wind) with their dynamic models and controllers. TS system includes distance protection for fault clearing, under/over frequency, and voltage protection for cascading events. Three-phase faults with impedances $Z_f = \{0 + j0, 1 + j0.1, 10 + j1, 10 + j10\}$ were applied to four transmission lines and two high-voltage buses separately where $\mathcal{N}_f = 24$. Hourly dispatch profiles of five days were randomly selected as initial conditions $\mathcal{N}_{oc} = 120$ for each fault, resulting in the database size as $|\Omega^X| = 2880$. The dynamics of five synchronous machines (NE-G2, NW-G5, SE-G6, SW-G1, SW-G8) were selected for Ω^X based on their geographical zones and machine ratings. The inverter-based generators can be used as a measurement point, but in this test network, their contribution to system dynamics was smaller than that of the selected generators due to their capacities. Moreover, the simulation environment provides noise-free data, but PMU measurements can contain a certain error. Having noisy measurements is not a challenge for the NODE that searches for the best ODE representation, where additional noise could increase training error but does not affect the mean approximated dynamics.

The AP clustering algorithm selected the representative simulation results for pretraining with the damping $\lambda = 0.5$. The standard deviation S of the sequence x^G was used for ApEn threshold $\mu = 0.2 \times S$, and the window size was set to $m = 2$ as a typical selection [47]. The initial model database Ω^θ stored the pretrained model parameters θ_κ. The training time for each window was $\tau^{train} = 3.45$ s, and the prediction horizon was $\tau^{pred} = 16.15$ s. The sampling period was 50 ms (equivalent to 69 time points for training). The delay time $t_d = 50$ ms avoided learning transient currents. The training was limited to only 40 epochs to test the convergence limit of the model. In a practical setup, the number of epochs could be increased to improve forecast accuracy and avoid convergence problems. The first 20 training epochs were during data collection. The last 20 epochs were considered for the entire training data. The following were regression baseline models: linear regression, support vector machines (SVM) with three kernels (linear, RBF, polynomial), and feedforward ANN. The ANN used 80% of the Ω^X data as training for the classification task. The ANN had three hidden layers (100, 100, 50 neurons) and used the cross-entropy loss with the ADAM solver. The performance metrics were the mean square error (MSE), the R^2 score, the Discrete Fréchet distance (DFD) for regression, and F1 score, accuracy, and confusion matrix rates (TPR, TNR, FPR, FNR) for classification of the security criterion z. The DFD computes the similarity between two curves considering the minimum required separation distance d to preserve geometric similarity [48]. The DFD is an upper bound for the Fréchet distance computed based on the longest discretized coupling links. Computing the DFD is computationally less demanding than Fréchet distance [49], and acknowledging the upper bound is sufficient for the ONDF. DigSilent Power Factory 2022 simulated the dynamic trajectories for TS and Texas 2000-bus systems. The NODEs were implemented using DiffEqFlux.jl in Julia v1.7 [50]. The library BenchmarkTools.jl measured the computational time of the NODE training. PowerSimulationsDynamics.jl [51] simulated the inverter dynamics in the modified 9-bus system. The experiments were conducted on a virtual machine with "Intel(R) Xeon(R) Gold 6148 CPU @ 2.4 GHz" and 16 Gb RAM.

4.1. NODE performance to forecast power system dynamics

This case study investigated the performance of the NODE model compared to other ML baselines using 50 random samples corresponding to five generators from TS with ten scenarios (five different fault locations) in Ω^X.

Fig. 5 showed the predictions of four dynamic states from X by three models, the ANN, NODE, and SVM (RBF) and RMS simulation results for generator "NE-G2". NODE had the best dynamics forecast with minimal error. SVM overfitted as forecast results replicated the observed training data. ANN forecasts quickly reached the wrong steady-state values. Table 1 showed the mean and standard deviation of

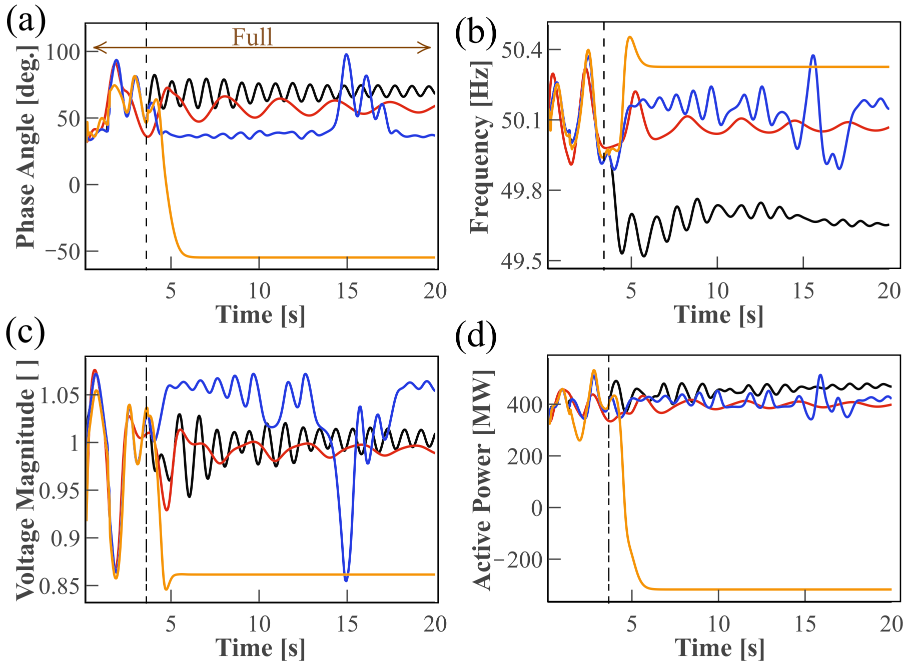

Fig. 5. Four RMS results (——) as the ground truth of TS for rotor angle difference (a), frequency (b), voltage magnitude (c), active power generation (d) and corresponding forecast results of three models NODE (——), ANN (——), and RBF SVM (——). The dashed line (┈┈) shows the end of the training data.

Table 1
Mean performance when predicting the dynamics with NODE and with baseline ML models for training and full trajectories (Fig. 5).

	Training			Full			Conv
Model	MSE [ ]	DFD [ ]	R^2 [ ]	MSE [ ]	DFD [ ]	R^2 [ ]	[%]
NODE	3.8	0.3	0.86	5.6	0.4	0.81	100
Linear	1.2	0.4	0.99	2×10^9	12 000	0.57	76
SVM (Lin)	1.5	0.5	0.98	1×10^9	900	0.68	92
SVM (Pol)	1.5	0.4	0.99	NaN	NaN	NaN	48
SVM (RBF)	1.4	0.4	0.99	12	4.0	0.79	100
ANN	2.1	0.7	0.96	68	9.8	0.41	100
	Considering only samples that have a loss ≤ 1000						
Linear	1.6	0.5	0.98	30	6.2	0.75	100
SVM (Lin)	1.0	0.4	0.98	44	5.8	0.74	100
SVM (Pol)	2.9	0.7	0.98	6.5	2.9	0.80	100

the performance metrics for the training and full trajectories. The predictions with MSE less than 1000 were considered converged. Baseline models could make long forecasts using previous predictions as new inputs that caused error accumulation and convergence issues. Long forecast horizons and changing dynamics caused large MSE differences between training and full trajectories in all baseline models besides the NODE. NODE showed the smallest DFD, indicating that the maximum deviation of the ground truth forecasts was minimal. Large DFD in forecasts could lead to false security assessments. R^2 scores showed baseline models fit well with the training data; however, forecast results did not match the system response.

The box plot of MSE in Fig. 6 for the three best-performing models showed the NODE forecasts had the minimum error with the smallest distribution. The ANN forecasts resulted in major errors and were unreliable. RBF SVM forecasts also had a small mean error; however, the forecast error increased parallel to the unstable system response. Compared to trajectory plots in Fig. 5, the boxplot shows a lower MSE distribution. Simulation with simpler dynamics would cause lower MSE values; that is why the overfitted SVM model also has a smaller mean MSE. However, when system dynamics are more severe, SVM cannot forecast the trend that can be observed from the Tukey whiskers.

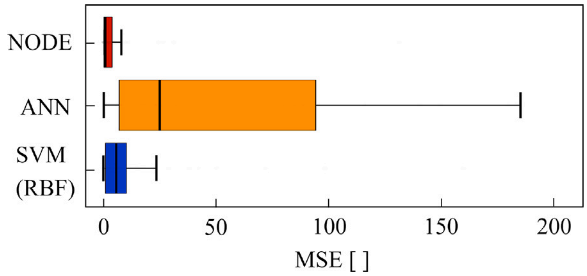

Fig. 6. MSE boxplot of NODE, ANN, and RBF SVM models. The box plots show mean, standard deviations and Tukey whiskers.

Fig. 7 showed the NODE's training performance starting from random parameters θ and also contained the final performance of ANN and RBF-SVM models for the training data in Fig. 5. Random initialization of θ caused high MSE and DFD in the underfitting region. NODE had a good fit between 100 and 400 epochs, preventing possible overfitting during the real-time training. Training the initial model from Ω^{θ} described in Section 4.3 would reduce the number of training epochs to

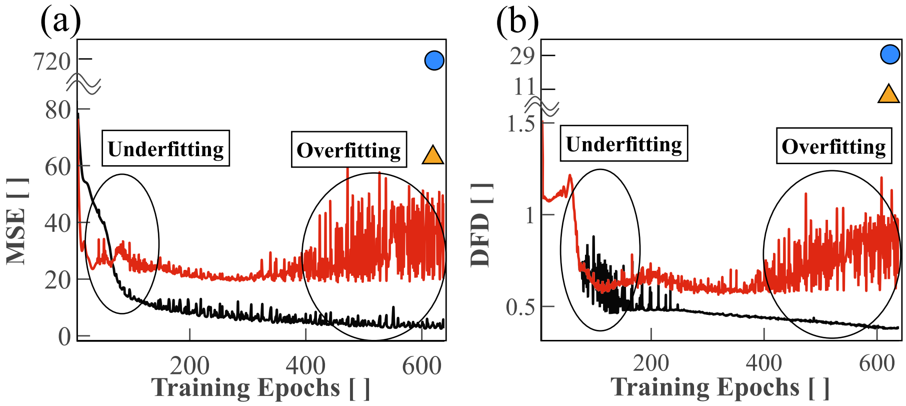

Fig. 7. Offline NODEs training performance with metrics MSE (a) and the DFD (b). The metrics are for training data (—) and for the full data (—). ANN (●), and RBF SVM (▲) results are for full trajectory only.

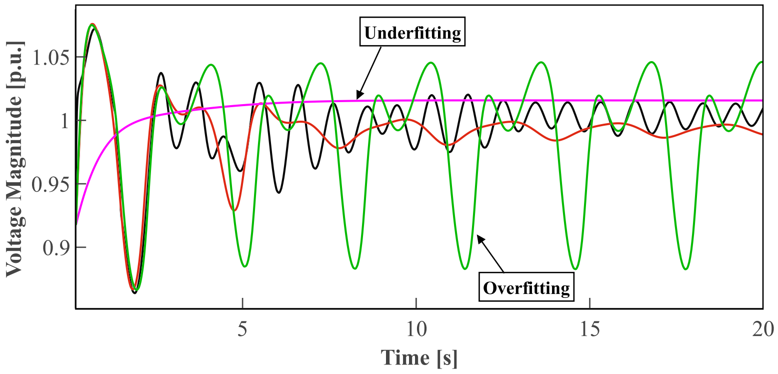

Fig. 8. Voltage magnitude forecasts from underfitting (—), overfitting (—), final (—) NODE models and RMS results (—) as ground truth.

reach a good fit for dynamics approximation. Fig. 8 showed the model forecasts from underfitting and overfitting regions, the final forecast, and the simulation output for the voltage magnitude. Underfitting forecast quickly reached a steady state while omitting voltage dynamics, while overfitting forecast periodically copied the training data similar to SVM (RBF) in Fig. 5.

4.2. Moving window NODE training for cascading events

This study investigated NODE forecasts for three different simulations to observe the impact of the moving window approach on NODE forecasts. Possible cascading events and changing dynamics due to controller actions altered the system response drastically. In order to trigger the cascading failures, we have further degraded the system security by increasing the system load and disconnecting one or two transmission lines (N-1, N-2) before applying the fault.

Fig. 9 showed the NODE forecasts of two generators for three consecutive time snapshots (windows) where all dynamic states were scaled and shifted for illustration purposes. The selected dynamics states had been minimally impacted by the initial event at $t = 0$. Although the dynamic impact of the fault was minimal, the only remaining line between South East to South West became overloaded. The protection system isolated the South West region, where system frequency and voltage rapidly increased due to an imbalance between active and reactive power demand and generation, around $t = 7$. The system split after the first training time window, causing the oscillations in SE-G6 dynamics and increased frequency and voltage in SW-G1. In the second snapshot, NODE forecasted stable steady-state voltage and frequency values as the new training data captured the following system dynamics. Although the frequency is a global variable, unlike the voltage magnitude, the South West region's frequency response is directly correlated to the controllers of SW-G1. In the third snapshot, NODE forecasts detected damped oscillations. The forecasted oscillations had a smaller amplitude than the simulation results, which would have been improved in the next snapshot.

Similarly, the main disturbance cannot be observed following a system disturbance in Fig. 10. Without dynamic characteristics, the NODE forecast cannot anticipate the cascading event. The forecast captured the frequency drop and estimated the frequency nadir correctly after the window shift. The under-frequency protection system disconnected loads from the system, which started the frequency restoration in around 12 seconds. Hence, the training data dynamics did not include the restoration, so the NODE forecast on the active power generation was settled at the nadir.

In Fig. 11, the system operated under N−1 conditions and experienced multiple cascading events during the simulation, leading to highly nonlinear and complex trajectories. From the limited training data, the NODE estimated rough trajectories and final steady-state values of NE-G2 that can be critical for assessing the system's security.

Figs. 9, 10, and 11 present highly nonlinear, complex dynamic trajectories from cascading simulations. Low dimensional decentralized approximation under limited training could not reproduce the same trajectory, but the trend and steady-state values are captured using only local measurements. The fitting towards to trend also shows that the NODE is insensitive against the noise or small oscillations in simulation results. Centralized higher dimensional models aided with physical knowledge of the system could produce more complex dynamic trajectories in the first window. However, any discrete events that alter

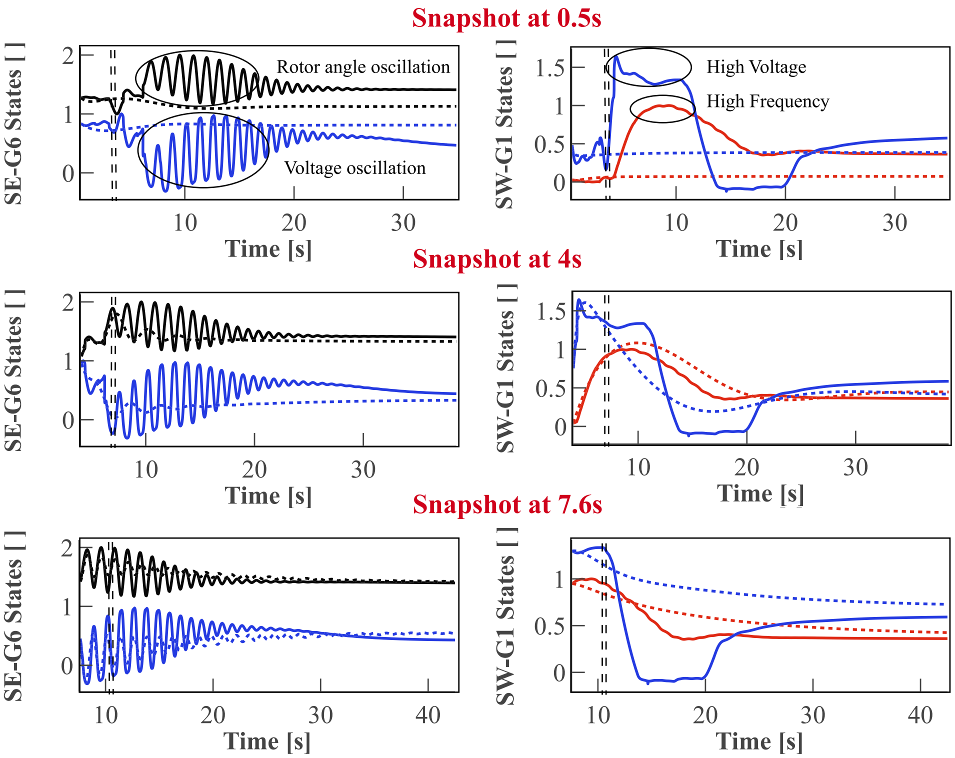

Fig. 9. RMS simulation results for generators SE-G6 (left) and SW-G1 (right) for scaled dynamic states: rotor angle difference (——), voltage magnitude (——), frequency (——) and corresponding NODE predictions (- - -) under three consecutive time windows following a disturbance. The line (= =) shows the end of the training.

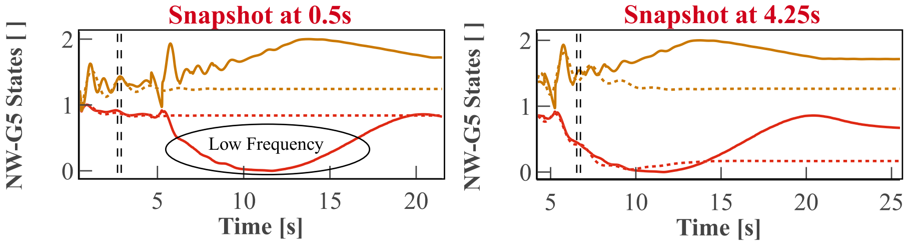

Fig. 10. RMS simulation results and NODE forecasts for generator NW-G5 in two consecutive time snapshots. Frequency (——) and active power generation (——) are illustrated with the corresponding NODE forecasts (- - -). The line (= =) shows the end of the training.

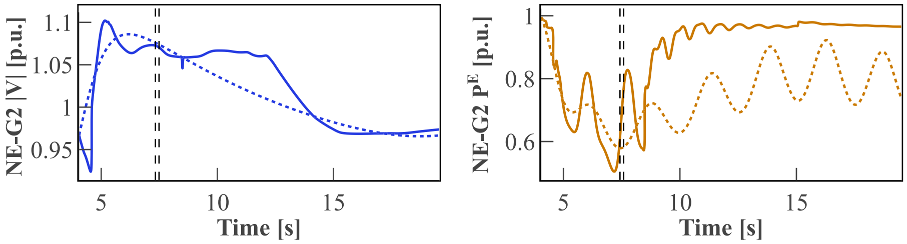

Fig. 11. RMS simulation results of the voltage magnitude (——) (left) and active power generation (——) (right) for NE-G2 for the time window II and corresponding NODE forecast (- - -). The dashed line (= =) shows the end of the training.

the physics of the system would cause the loss of generalization of the forecaster model.

4.3. Offline NODE preparation

This study investigated the selection of pretraining data and NODE tuning for online forecasting. The AP algorithm identified 106 clusters from the dynamics database. A small set of cluster centroids was sufficient as the pre-trained models are warm starts for real-time training. The AP clustered centroids to identify the final 20 centroids. The computed centroids only represent a small portion of the dataset where the main goal is achieving a small initial forecast error rather than generalizing to system dynamics.

The parameter tuning started from the initial model and changed one hyperparameter at a time. Table 2 showed the initial and the tuned model parameters. Table 3 showed their performance after training 250 epochs with random samples from the dynamics database. NN parameters affected the complexity of the constructed ODE system. More layers and neurons in the initial model caused more complex NN, hence longer training times. Wrong ODE solver and tolerance selection

Table 2
Parameter of the initial and final (tuned) NODE models.

Model	NN	ODE Solver	Optimizer
Initial	3 Hidden Layers 100 Neurons/layer Sigmoid activation	RK4[a] Absolute tolerance = 1e−3 Relative tolerance = 1e-5	ADAM Learning rate = 0.01
Tuned	2 Hidden Layers 50 Neurons/layer Tanh activation	Tsit5[b] Absolute tolerance = 1e−5 Relative tolerance = 1e-7	NADAM Learning rate = 0.01

[a] The canonical Runge–Kutta Order 4 method.

[b] Tsitouras 5/4 Runge–Kutta method.

Table 3
Tuning NODE models (mean (standard deviation)).

Model	Time [s]	MSE [ ]	R2 [ ]	DFD [ ]
Initial	84(19)	468(1069)	0.19(0.92)	2.6(2.5)
Tuned	13(1.6)	45(253)	0.83(0.36)	0.7(1.2)

Table 4
ONDF based security assessment classification performance compared to ANN classifier with values in [%].

Model	Insecure share	TPR	TNR	FPR	FNR	F1	Accuracy
Truth	61.3	100	100	0	0	100	100
ANN	68.0	82.9	55.5	44.4	17.1	78.6	72.3
NODE-1	65.6	98.8	87.1	12.9	1.6	95.5	94.3
NODE-2	61.1	99.0	99.0	1.0	1	99.1	99.0
NODE-3	60.9	99.0	99.5	0.5	1	99.3	99.2

Table 5
The ONDF performance in large power grids (mean (standard deviation)).

System	Baseline	NODE		
model	RMS time [s]	Train time [s]	MSE [ ]	DFD [ ]
IEEE (9)	0.84 (0.16)	3.2 (0.8)	8.4 (14)	0.8 (0.2)
TS (46)	11.8 (4.8)	2.4 (0.4)	9.8 (43)	0.8 (1.3)
Texas (2000)	69.1 (19.7)	2.1 (0.8)	18 (117)	0.6 (2.6)

could lead to poor generalization and longer training times. The higher error between timesteps requires the computation of more time-step points during the ODE solution.

The unmentioned design parameters are training and prediction times τ^{train}, τ^{pred}. Values of these parameters are highly dependent on system dynamics. Systems with higher inertia will likely experience slower dynamics, hence longer prediction and training times. In this case study, we tried to minimize τ^{train} to test the NODE limits without sacrificing the prediction capability. Longer τ^{pred} values cause growing errors in forecasts as system dynamics change, but solving the ODE system instead of autoregressive prediction limits the growing errors. Sensitivity analysis on the prediction error and prediction times could help to tune the parameter.

4.4. ONDF application for online security assessment

This study investigated the performance and adaptability of the ONDF for the system security classification. The study used 1000 samples from Ω^X with $\gamma^t = 0.5$ s. The NODE-k considered the time-window k that shifted in every $\tau^{val} = 3.85$ s.

The comparison of several classification metrics for three moving windows (NODE-1,2,3), ANN, and actual system security (ground truth) showed the NODE had high performance. Even in the first measurement window, the NODE reached 94.3% accuracy in Table 4. Most errors were false alarms, as the model had a higher TPR than TNR. In the first window, the actions from frequency and damping controllers are less observable due to delays and the model's time constants. For example, in Fig. 10, NODE-1 assessed the system security incorrectly using the first window's measurement. Because the event caused low frequency occurred outside of the first window. Using the new data from the second window allowed NODE-2 forecasts for accurate assessment. The NODE did not learn this control action. Hence, the NODE-1 predictions had more false alarms than other time windows. NODE-2 and 3 showed lower false alarms and improved overall accuracy and F1 score by 3.5% compared to the NODE-1. The NODE outperformed the ANN in all metrics as the decentralized approach limited the number of features for the ANN. Moreover, the training data of ANN contained various disturbance cases that might require a larger training database for the ANN. Even with a larger dataset, the centralized ANN classifier might suffer from generalization problems. More advanced models (LSTM, GRU, CNN) would potentially provide higher accuracy, but training and data generation become infeasible for practical networks. When the performance of centralized and decentralized are comparable, the decentralized method is more applicable for large power systems.

The ONDF workflow can be used for any system security limits γ^t as protection limits are not identical. The sensitivity analysis in Fig. 12 shows the drop in TNR as the number of secure cases increases, however, the TPR remained constant. Overall, change in classification is limited and depends on the distance between the forecast results and security limits, where system operators can evaluate the event based on advanced monitoring collected from all forecasters to avoid false classification. Unfortunately, applying this sensitivity analysis to the centralized ANN model requires the computation of security labels for each security limit γ^t and retraining the model parameters. Centralized methods are less adaptive to changes in security boundaries because of the fixed security boundary.

4.5. Real-time training of neural ODEs

This study investigates the computational requirements of the NODE training. The box plot in Fig. 13 shows the DFD and the required training time of the same samples in Section 4.1 after 25 epochs for three variations of using NODEs. The first variation is the NODE (baseline) using the direct raw measurements X_M as input without preprocessing. This baseline had poor performance in DFD and times as the loss values largely varied. The NODE with preprocessed input data reduced the required training time. The proposed ONDF considers the NODE with preprocessing, model selection, and growing function. The proposed approach has the lowest DFD and fastest training. The decentralized nature of the ONDF workflow allows system operators to conduct training with moderate computational power near the measurement locations. Further optimization in software development and specific hardware for the task can reduce the required training times.

4.6. The ONDF performance in larger power grids

This study investigates the performance of ONDF to a larger grid, the Texas grid with 2000 buses. Table 5 shows the mean and standard deviation of the NODE models' performance and training time for 50 different training data from the three systems. All three power systems had similar training times as expected and slight differences in the

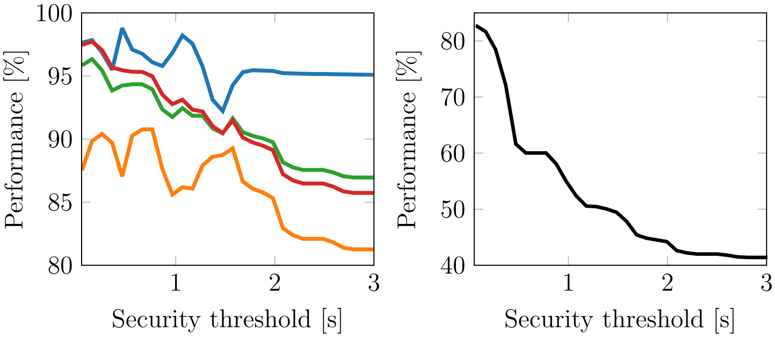

Fig. 12. Sensitivity of NODE-1 to the security threshold γ^t: Left is the TPR (🟦), TNR (🟧), accuracy (🟩) and F1 (🟥). Right is the share of insecure cases (⬛) in the ground truth.

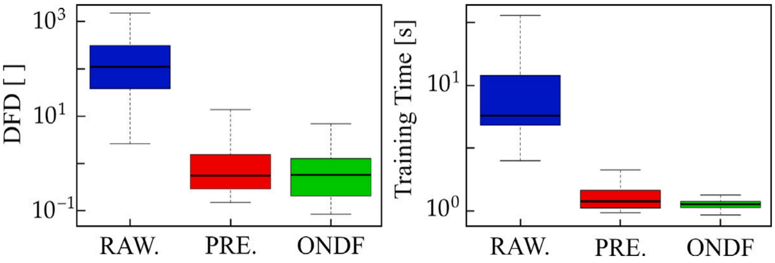

Fig. 13. DFD performance on left and the required computational training times on right for three variations of using NODEs: either with raw input data (🟦), with pre-processing (🟥) and the proposed ONDF workflow (🟩). The box plots show mean, standard deviations, and Tukey whiskers.

system responses to similar events. Higher mean and standard deviation in MSE for Texas and TS systems results from simulations with more complex behavior derived from the higher number of controllers, compared to IEEE 9. The low DFD in all three systems indicates the ONDF workflow scales to larger systems and is much faster than RMS simulations as the system size increases, which is a key finding of this work. Compared to our decentralized approach, the centralized RMS simulations accurately represent system dynamics. However, uncertainty in fault models and long solver times, approximately 30–40 times longer than online training, make this method infeasible for online operations.

4.7. Discussion of practical setup plan

The practical setup plan starts with the hardware requirements for the workflow. Due to its decentralized nature and efficient training strategies, the workflow requires low computational power for training NODEs. Furthermore, specific hardware components like an FPGA might be designed to lower the cost and optimize the algorithm's performance. The provided workflow is a low-cost solution with minimal effort compared to other centralized solutions requiring fast communication networks, big data management, and large computational power for data generation and model training.

The practical setup plan envisions local computers or processing devices integrated with the existing monitoring solutions. The computers are located in substations to avoid communication delays and are directly interlinked with the Supervisory Control and Data Acquisition (SCADA) system through energy or outage management systems. SCADA systems enable a channel for efficient data transfer. In this setup, we need further data processing to collect and convert analog data from the transformers to digital data. The recent trends in power system security include more PMU devices and wide area monitoring, protection, and control (WAMPAC) applications. WAMPAC systems are suitable for integrating machine learning tools for early warning and mitigating system instabilities [52]. In the case of existing WAMPAC, the NODE can be easily integrated as the data is already ready for computation in a phasor form. The efficient part of the model is that NODE does not have to send the computed trajectories but only model parameters, independent of the data length or event duration. Parameter sharing allows system operators to generate forecast results in the control room without additional challenges.

The practical setup plan also considers the possible issues regarding measurement data quantity and quality. Although PMU devices are highly precise, there could be timesteps where samples are missed. Missing samples endanger the integrity of the sequential data and convergence of the model. The NODE model forecasts the future dynamics using only the initial condition, not the sequential data. Since training loss can be computed for any time point, the workflow will be impacted minimally compared to other sequential machine learning models in the case of missing data. Moreover, irregularly sampled data due to unwanted delays do not affect the training because of the flexibility of the adjoint sensitivity method. The impact of noisy data is also limited for NODEs during the training since noisy forecasts are hard to obtain from solving ODEs; therefore, forecasted dynamics would be noise-free in the control room. The models' convergence capability could be verified by computing performance indices during the validation period. Models with high errors could alert system operators against data quality or model learning.

The last part of the practical setup investigates necessary adaptations for system operators. Although NODEs are complicated models, the required adaptation for system operators might be limited to the initial deployment, where system operators should get familiar with ODE systems and solvers. Operators can obtain NODE forecast results during the operation without analyzing the model architecture or feature space by solving ODE systems parametrized by the decentralized models. Online training also provides no maintenance requirements for the training database, whereas centralized data-driven solutions often require updates after system conditions and security labels change. To avoid performance drops in these centralized solutions, system operators should learn the properties of high-quality training data and model training. Moreover, the training data generation is hard for system operators due to the problem's dimensionality while maintaining a representative distribution of possible contingency and operating condition space [53]. Modifications in the dynamic model and its parameters could alter the security boundary, hence the original training data labels. The problem of maintaining the model and dataset consumes much more time and computational resources for system operators than the ONDF workflow.

4.8. Discussion

The ONDF workflows showed various benefits for the power system dynamics forecasting problem. As observed in Section 4.1, ML models with discrete regression output are more vulnerable to divergence as error grows with longer prediction horizons. The NODE model outperformed all discrete models and did not suffer from divergent forecasts when stable system response was the training data. The NODE forecasts obey ODE system properties. The fitting characteristics in Section 4.1 show us NODE starts from a simple dynamic system and approximates more complex trajectories without producing large forecast errors as other models. The training with the moving window enables NODE to replace the learned dynamics to capture changing dynamics from control actions or cascades in the system, illustrated with several examples in Section 4.2. However, NODE contains approximation errors because of lower dimensional representation, discrete events occurring in the forecast horizon, and limited training time. The approximation errors were still in acceptable margins when we observed corresponding DFD values for security analysis tasks. The preprocessing and tuning are essential for real-time operation as the mean training time is significantly reduced compared to the base model. Depending on the computational power, the number of epochs can be increased to improve the quality of predictions. ONDF utilizes local dynamics, so the scale of the system does not impact the training of a single model, as observed in Section 4.6. Still, ONDF requires more NODEs to assess the security successfully, and future work can focus on the number and location of NODEs for practical power system sizes. Security assessments in Section 4.4 show NODE forecasts can provide accurate estimations under changing security limits. Unlike data-driven classifiers, ONDF is adaptive to the security limits and does not require retraining as ONDF has no bias towards the security limits. Moreover, NODEs do not require new training database generation after system dynamics and network topology changes.

The main limitation of NODE comes from the cases where the grid experiences a devastating disturbance that causes system blackouts or cascades to happen faster than the required training time. In such a case, the system's survival chance becomes minimal due to a lack of time for any corrective actions. Another possible limitation might occur when high-frequency oscillations are present in the system, which causes a stiff ODE system that is hard to solve and learn in real time.

The proposed workflow would be applicable in the future energy transition while a higher share of IBRs will be present in the energy system. However, during this transition, power system dynamics will change, and so will the parameters of the workflow as well. The first impact on system dynamics will be faster dynamics due to the power electronic converter's controller dynamics being significantly faster than conventional electro-mechanical synchronous machine dynamics. Faster dynamics would cause a shorter training time span τ^{train}. In this work, we have used a larger timestep $\Delta_t = 50$ ms, whereas PMU devices have a sampling rate of around tens of samples per second [54]. Increasing rate compensates for the faster dynamics, but further research could investigate the impact of the high IBR share and possible control actions for future power systems.

5. Conclusion

Power system dynamics after unpredictable events will become more challenging to model, simulate, and learn from as the number of distributed IBRs grows. Our proposed approach, training multiple NODE models using only local dynamics measurements, showed to approximate the event-specific dynamics. Our approach moves the training window to learn and assess the system's security in real time; therefore can consider cascading failures and other discrete events in real time. Future research will focus on developing this workflow for advanced monitoring and possible control actions.

CRediT authorship contribution statement

Mert Karacelebi: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Jochen L. Cremer:** Writing – review & editing, Visualization, Supervision, Methodology, Investigation, Formal analysis, Conceptualization.

Funding

This work has been supported by the Dutch Research Council (NWO) (Veni Talent Program Grant 19161) and the Technical University of Delft AI Initiative Program. The sponsors did not have any relation to the study design, collection, analysis and interpretation of data, writing of the report and decision to submit the article for publication.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

[1] Kenyon RW, Bossart M, Marković M, Doubleday K, Matsuda-Dunn R, Mitova S, et al. Stability and control of power systems with high penetrations of inverter-based resources: An accessible review of current knowledge and open questions. Sol Energy 2020;210(May):149–68.

[2] Milano F, Dorfler F, Hug G, Hill DJ, Verbič G. Foundations and challenges of low-inertia systems (Invited Paper). Power Syst Comput Conf 2018.

[3] Henriquez-Auba R, Lara JD, Callaway DS, Barrows C. Transient simulations with a large penetration of converter-interfaced generation: scientific computing challenges and opportunities. IEEE Electrification Mag 2021;9(2):72–82.

[4] Podmore R. Identification of coherent generators for dynamic equivalents. IEEE Trans Power Appar Syst 1978;PAS-97(4):1344–54.

[5] Duchesne L, Karangelos E, Wehenkel L. Recent developments in machine learning for energy systems reliability management. Proc IEEE 2020;108(9):1656–76.

[6] Wehenkel L, avella M. Decision Trees and Transient Stability of Electric Power Systems. Automatica 1991;27(1):115–34.

[7] Baltas NG, Mazidi P, Ma J, De Asis Fernandez F, Rodriguez P. A comparative analysis of decision trees, support vector machines and artificial neural networks for on-line transient stability assessment. Int Conf Smart Energy Syst Technol 2018.

[8] Thams F, Venzke A, Eriksson R, Chatzivasileiadis S. Efficient database generation for data-driven security assessment of power systems. IEEE Trans Power Syst 2019;35(1):30–41.

[9] Bugaje AAB, Cremer JL, Strbac G. Generating quality datasets for real-time security assessment: Balancing historically relevant and rare feasible operating conditions. Int J Electr Power Energy Syst 2023;154:109427.

[10] Xia T, Hou Q, Zhang N, Dong Q, Li W, Kang C. Database generation for data-driven power system security assessment under uncertainty. IEEE Trans Power Syst 2024.

[11] Shimizu K, Ishigame A. Transient stability assessment application using post-disturbance voltage fluctuations in a multi-machine power system. Int J Electr Power Energy Syst 2022;139(June 2021):1–8. http://dx.doi.org/10.1016/j.ijepes.2022.107987.

[12] Ren C, Xu Y, Zhang Y. Post-disturbance transient stability assessment of power systems towards optimal accuracy-speed tradeoff. Prot Control Mod Power Syst 2018;3(1):19. http://dx.doi.org/10.1186/s41601-018-0091-3.

[13] Chen Q, Wang H. Time-adaptive transient stability assessment based on gated recurrent unit. Int J Electr Power Energy Syst 2021;133:107156. http://dx.doi.org/10.1016/j.ijepes.2021.107156.

[14] Han X, Chen Z, Wang Y, Lai X, Ding L, Wen F. LSTM-CNN-based transient stability assessment. In: 2021 international conference on power system technology. POWERCON, 2021, p. 1930–5. http://dx.doi.org/10.1109/POWERCON53785.2021.9697781.

[15] Nakas GA, Dirik A, Papadopoulos PN, Matavalam ARR, Paul O, Tzelepis D. Online Identification of Cascading Events in Power Systems With Renewable Generation Using Measurement Data and Machine Learning. IEEE Access 2023;11(July):72343–56. http://dx.doi.org/10.1109/ACCESS.2023.3294472.

[16] Shi F, Wu J, Wang Y, Li L, Zheng Y. Integrated advance assessment of power system transient voltage and transient angle stability based on two-stage ensemble spatio-temporal graph neural network. Measurement 2023;221:113447. http://dx.doi.org/10.1016/j.measurement.2023.113447.

[17] Ren C, Xu Y. Transfer learning-based power system online dynamic security assessment: Using one model to assess many unlearned faults. IEEE Trans Power Syst 2020;35(1):821–4. http://dx.doi.org/10.1109/TPWRS.2019.2947781.

[18] Zhou Y, Guo Q, Sun H, Yu Z, Wu J, Hao L. A novel data-driven approach for transient stability prediction of power systems considering the operational variability. Int J Electr Power Energy Syst 2019;107:379–94. http://dx.doi.org/10.1016/j.ijepes.2018.11.031.

[19] Susuki Y, Chakrabortty A. Introduction to koopman mode decomposition for data-based technology of power system nonlinear dynamics. IFAC- Pap 2018;51(28):327–32, 10th IFAC Symposium on Control of Power and Energy Systems CPES 2018.

[20] Chamorro HR, Member S, Orjuela-cañón AD, Member S, Ganger D, Persson M, et al. Nadir Frequency Estimation in Low-Inertia Power Systems. In: 2020 IEEE 29th international symposium on industrial electronics. ISIE, IEEE; 2020, p. 918–22.

[21] Chamorro HR, Orjuela-cañón AD, Ganger D, Persson M, Gonzalez-longatt F, Alvarado-barrios L, et al. Data-Driven Trajectory Prediction of Grid Power Frequency Based on Neural Models. Electronics 2021;10:1–21.

[22] Li J, Yue M, Zhao Y, Lin G. Machine-learning-based online transient analysis via iterative computation of generator dynamics. 2020 IEEE Int Conf Commun Control Comput Technol Smart Grids (SmartGridComm) 2020.

[23] Cui W, Yang W, Zhang B. A Frequency Domain Approach to Predict Power System Transients. IEEE Trans Power Syst 2023;39(1):465–77.

[24] Zhao T, Yue M, Wang J. Structure-Informed Graph Learning of Networked Dependencies for Online Prediction of Power System Transient Dynamics. IEEE Trans Power Syst 2022;37(6):4885–95.

[25] Chen RTQ, Rubanova Y, Bettencourt J, Duvenaud DK. Neural ordinary differential equations. In: Bengio S, Wallach H, Larochelle H, Grauman K, Cesa-Bianchi N, Garnett R, editors. Advances in neural information processing systems. vol. 31, Curran Associates, Inc.; 2018.

[26] Rackauckas C, Ma Y, Martensen J. Universal Differential Equations for Scientic Machine Learning. 2020, arXiv:arXiv:2001.04385v3.

[27] Grathwohl W, Chen RTQ, Bettencourt J, Sutskever I, Duvenaud D. FFJORD: Free-Form Continous Dynamics For Scalable Reversible Generative Models. In: 7th international conference on learning representations, ICLR 2019.. 2019, p. 1–13.

[28] Linot AJ, Burby JW, Tang Q, Balaprakash P, Graham MD, Maulik R. Stabilized neural ordinary differential equations for long-time forecasting of dynamical systems. J Comput Phys 2023;474:111838.

[29] Linot AJ, Burby JW, Tang Q, Balaprakash P, Graham D, Maulik R. Stabilized neural ordinary differential equations for long-time forecasting of dynamical systems. J Comput Phys 2023;474:111838. http://dx.doi.org/10.1016/j.jcp.2022.111838, arXiv:j.jcp.2022.111838.

[30] Yin Y, Guen VL, Dona J, de Bézenac E, Ayed I, Thome N, et al. Augmenting physical models with deep networks for complex dynamics forecasting. J Stat Mech Theory Exp 2021;2021(12):124012.

[31] Legaard C, Schranz T, Schweiger G, Drgońa J, Falay B, Gomes C, et al. Constructing Neural Network Based Models for Simulating Dynamical Systems. Assoc Comput Mach (ACM) Comput Surv 2023;55(11):1–35.

[32] Sandoval IO, Petsagkourakis P, del Rio-Chanona EA. Neural ODEs as feedback policies for nonlinear optimal control. IFAC- Pap 2023;56(2):4816–21, 22nd IFAC World Congress.

[33] Goyal P, Benner P. Neural ordinary differential equations with irregular and noisy data. R Soc Open Sci 2023;10(7). http://dx.doi.org/10.1098/rsos.221475.

[34] Xie X, Parlikad AK, Puri RS. Approach for Demand Forecasting within Power Grid Digital Twins. 2019 IEEE Int Conf Commun Control Comput Technol Smart Grids (SmartGridComm) 2019;1–6.

[35] Kong X, Yamashita K, Foggo B, Yu N. Dynamic parameter estimation with physics-based neural ordinary differential equations. In: IEEE PES general meeting. 2022, p. 1–5.

[36] Xiao T, Chen Y, Huang S, He T, Guan H. Feasibility study of neural ODE and DAE modules for power system dynamic component modeling. IEEE Trans Power Syst 2023;38(3):2666–78.

[37] Zhang S, Yamashita K, Yu N. Learning Power System Dynamics with Noisy Data Using Neural Ordinary Differential Equations. In: IEEE power and energy society general meeting. 2024, http://dx.doi.org/10.1109/PESGM51994.2024.10689132.

[38] Fadoul FF, Çağlar R. Integrating Bayesian inference and neural ODEs for microgrids dynamics parameters estimation. Sustain Energy, Grids Networks 2024;39:101498. http://dx.doi.org/10.1016/j.segan.2024.101498.

[39] Dahale S, Munikoti S, Natarajan B, Yang R. Latent neural ODE for integrating multi-timescale measurements in smart distribution grids. In: 2023 IEEE power & energy society innovative smart grid technologies conference. ISGT, 2023, p. 1–5. http://dx.doi.org/10.1109/ISGT51731.2023.10066442.

[40] Bugaje AAB, Cremer JL, Strbac G. Split-based sequential sampling for realtime security assessment. Int J Electr Power Energy Syst 2023;146:108790.

[41] Pincus SM. Approximate entropy as a measure of system complexity. Proc Natl Acad Sci 1991;88.

[42] Delgado-Bonal A, Marshak A. Approximate entropy and sample entropy: A comprehensive tutorial. Entropy 2019;21(6).

[43] Xia DY, Wu F, Zhang XQ, Zhuang YT. Local and global approaches of affinity propagation clustering for large scale data. J Zhejiang University: Sci A 2008;9(10).

[44] Bischl B, Binder M, Lang M, Pielok T, Richter J, Coors S, et al. Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges. WIREs Data Min Knowl Discov 2023;13(2).

[45] Kundur P, Malik OP. Power System Stability And Control. McGraw-Hill, Inc; 1994, p. 1167.

[46] DIgSILENT. PowerFactory. 2022.

[47] Chou CM. Complexity analysis of rainfall and runoff time series based on sample entropy in different temporal scales. Stoch Environ Res Risk Assess 2014;28(6).

[48] Buchin K, Fan C, Löffler M, Popov A, Raichel B, Roeloffzen M. Fréchet distance for uncertain curves. In: Leibniz international proceedings in informatics, lIPIcs. vol. 168, 2020, p. 1–48.

[49] Eiter T, Mannila H. Computing discrete Fréchet distance. Tech. report CD-tR 94/64, 94, Vienna: Information Systems Department, Technical University of Vienna.; 1994, p. 64.

[50] Bezanson J, Edelman A, Karpinski S, Shah VB. Julia: A fresh approach to numerical computing. SIAM Rev 2017;59(1).

[51] Lara JD, Henríquez-Auba R, Bossart M, Valverde G, Lincoln R, Roberts C, et al. NREL-SIIP/PowerSimulationsDynamics.jl: v0.10.10. 2023.

[52] Wagh S, Kazi F, Gupta S, Singh N. Augmenting WAMPAC with machine learning tools for early warning and mitigation of blackout events. Int J Humanit Technol 2018;1(1):83. http://dx.doi.org/10.1504/IJHT.2018.10011336.

[53] Xia T, Hou Q, Zhang N, Dong Q, Li W, Kang C. Database Generation for Data-Driven Power System Security Assessment Under Uncertainty. IEEE Trans Power Syst 2024;39(5):6168–82. http://dx.doi.org/10.1109/TPWRS.2024.3352825.

[54] Shai D, Chen C, Pooley M, D'Andrade BW. 10 - prognostics for the power industry. In: D'Andrade BW, editor. The power grid. Academic Press; 2017, p. 287–316.