Identifying Overlapping Successive Events Using a Shallow Convolutional Neural Network

Wenting Li, *Student Member, IEEE,* Meng Wang, *Member, IEEE*

***Abstract*—Real-time identification of successive events in power systems is crucial to avoid cascading failures. Existing identification methods are mainly designed for single events and may not accurately identify a subsequent event that occurs when the system is under going the disturbance of a previous event. Since overlapping successive events do not frequently happen in power systems, insufficient multi-event datasets exist for training. We develop a data-driven event identification method that can accurately identify the types of overlapping events. Our approach only requires a small number of recorded Phasor Measurement Unit (PMU) data of single events to train a two-layer convolutional neural network (CNN) classifier offline. We extract the dominant eigenvalues and singular values as features instead of training on time series directly. That reduces the required number of training datasets and enhances the robustness to measurement inaccuracy. In real time, our method first predicts and subtracts the impact of previous events. It then extracts the dominant features from the residual measurements and applies the classifier. We evaluate the method on simulated events in the IEEE 68-bus power system. Our classifier is demonstrated to be more accurate and stable than a direct application of CNN on time series. The robustness of the proposed method to the delay in event detection and noise is validated.**

***Index Terms*—Event identification, Phasor Measurement Unit (PMU), Convolutional Neural Network (CNN), cascading failures, dominant features**

I. Introduction

Cascading failures usually start from a slow-cascade phrase that lasts tens of seconds to hours, however, once a fast-cascade phrase is triggered by violation of the power system stability, rapid events occur successively within milliseconds to tens of seconds [1]. Some recent works have modeled cascading failures by simulating the latent physical mechanisms. Examples include the ORNL-PSerc-Alaska (OPA) model and its extensions [2], [3], the Branching Process (BP) model [4], and the Hidden Failure (HF) model [5]. The OPA model represents the power system through slow dynamics (like increasing slow loads) and fast dynamics (like line outages and load shedding) to evaluate the risk of a blackout and determine the critical conditions. The BP model uses a stochastic process to model cascading failures and predict the probability of a blackout. The HF model evaluates the effectiveness of the mitigation strategies to reduce the possibility of a blackout. These models reveal some critical factors that may cause cascading failures, but cascading failures involving many different mechanisms are difficult to capture by a single model [6].

W. Li and M. Wang are with the Dept. of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY. Email: {liw14, wangm7}@rpi.edu. This research is supported in part by the ERC Program of NSF and DoE under NSF EEC-1041877 and the CURENT Industry Partnership Program, and NSF Grant 1508875.

Another line of research is to improve monitoring and protection methods, such that events can be identified and mitigated at an early stage to prevent cascading failures [7], [8]. Traditional methods using local measurements such as relay and fault recordings to detect events suffer from malfunctions of devices [9], long communication delays [10], and improper coordination of multiple back-up protection zones [11]. Moreover, the data rate of the supervisory control and data acquisition (SCADA) system is around 1-5 seconds per sample, and it is not fast enough to identify the event in real time. These issues motivate the recent research on data-driven methods using high-resolution (30 samples per second per channel) synchronized measurements provided by Phasor Measurement Units (PMUs) [12]–[16].

Data-driven event identification methods [13]–[19] train a classifier based on measurements or extracted features. One way to train the classifier is through neural networks, which have been employed in fault diagnosis in power systems for more than twenty years [20]–[22]. Building upon advanced neural networks, deep convolutional neural networks (CNN) [23] recently demonstrated superior performance for image and video classification [24]. The application of deep learning in power system event identification has just begun [25] and is attracting increasing attention.

The above data-driven approaches for event identification suffer from some limitations. They all assume that the system is in a steady state when an event occurs. The identification performance of successive events degrades when the measurements are affected by multiple events. For example, a subsequent event occurs when the system is still under the disturbance resulting from a previous event. Moreover, a large number of training events are needed to cover all possible topology changes, initial conditions, and event locations to train a reasonable classifier, especially neural networks, which perform poorly when the number of training samples is small [26]. Since successive events with overlapping impacts do not happen very frequently in practice, historical event datasets do not cover all possible scenarios for training.

The major contribution of this paper is the development of a data-driven approach using a CNN classifier that accurately identifies event types in real time when multiple events occur close in time and location. We propose to train a classifier on compact features such as the dominant eigenvalues of the dynamical system and the singular values of the data matrix instead of time series measurements directly. These features characterize the system dynamics and are robust to pre-event conditions [27], [28]. Thus, the required number of training events and the complexity of the classifier are significantly

reduced. Different from the deep neural networks [23], [24], our proposed classifier is shallow and has a much smaller number of parameters to learn. Moreover, our method only requires recorded *single events* rather than successive events for training. When identifying the successive events in real time, a prediction-subtraction process is proposed to reduce the impacts of the former events on the subsequent events.

We show through numerical experiments that our classifier outperforms a CNN classifier trained on time series measurements, as well as other classifiers such as fully-connected neural network (NN) and multi-class support vector machine (MSVM). Moreover, our classifier performs well even when the size of the training set decreases, the time intervals between successive events changes, the detection of an event has some delay, and the measurements contain noise.

The rest of the paper is organized as follows. Section II describes the problem setup and summarizes the major components of our method. Section III illustrates the feature extraction approach and the impact of the subtraction. Section IV describes the prediction, subtraction, and feature extraction process for the event data in real time. Section V discusses the offline training of the CNN classifier. Section VI shows the numerical results on simulated events in the IEEE 68-bus power system. Section VII concludes the paper.

II. Technical Challenges and Our Approach

We first illustrate the difficulty of identifying successive events through simulated events using the Power System Simulator for Engineering (PSS/E) [29] in the IEEE 68-bus power system (Fig.1). Fig. 2(a) shows two successive events that occur spatially apart and do not significantly affect each other. The line trip between bus 17 and bus 43 only changes the voltage of some nearby buses such as bus 39. The second line trip event between buses 25 and 26 happens sufficiently far away. In this simple case, we can apply the existing single event identification methods, e.g., [27], [28], to identify the types of either event separately and obtain satisfactory results. Fig. 2(b) shows two successive events that occur temporally and spatially close such that the measurements are affected by both events simultaneously. The voltage magnitudes of buses 24, 29, and 67 fluctuate after the generator trip at bus 4. Then the line trip between buses 26 and 29 also affect these voltages. Thus, the measurements after 2 seconds are influenced by both events. If we directly apply single event identification methods on these measurements, the second event might not be correctly identified.

The main focus of this paper is to identify the types of successive events, especially when measurements after the subsequent event is still affected by the previous event. To simplify the discussion, we focus on two successive events in this paper, while our approach can be easily generalized to more successive events. We will mainly discuss the identification of the second event because the first event can be identified using existing single event identification methods.

The major contributions of our proposed identification methodology include three aspects. First, we propose to predict the measurements after the starting time of the second event,

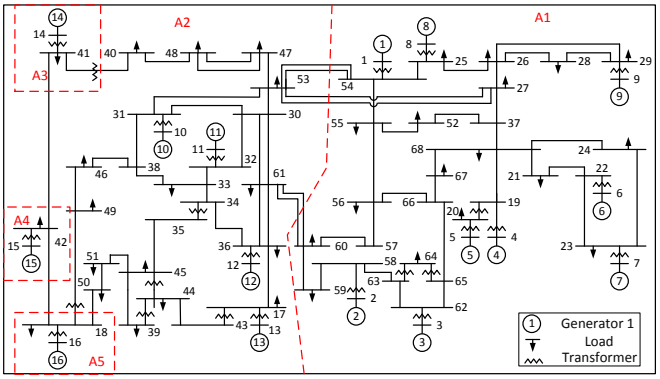


Fig. 1: The IEEE 68-bus power system with five groups [30]

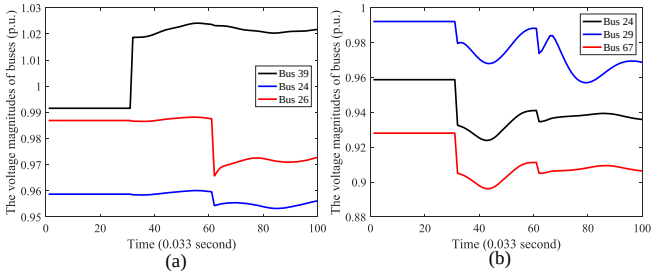


Fig. 2: (a) The voltage magnitudes in per unit (p.u.) of buses when Line 17-43 tripped at 1 second followed by line 25-26 tripped event at 2 seconds. (b) Generator at bus 4 tripped at 1 second followed by line 26-29 tripped at 2 seconds

assuming that the second event did not occur. We then subtract the predicted measurements from the actual observations at the corresponding time instants to reduce the impact of the first event on the measurements. Second, we propose to train a two-layer CNN on recorded single events to obtain a classifier to identify each event. Our approach does not require recorded successive events like cascading failures, which do not occur often in practice. Third, instead of training a classifier on measurements, we extract the dominant features such as dominant eigenvalues of the state matrix of a dynamical system and train the CNN on these features. We show that these features are compact representations of the system dynamics and are robust to pre-event system conditions and topology changes Thus, both the size of the CNN and the number of recorded events for training are significantly reduced.

III. Dominant Feature Analysis and the Subtraction of Impact

The measurements in power systems are governed by the underlying dynamical systems and are thus highly correlated. The time series can be compactly represented by low-dimensional features. We will train a classifier using the dominant features instead of the time series directly to reduce the number of parameters in the classifier and the required number of training events. Moreover, we will subtract the impact of the first event to increase the estimation accuracy of the dominant features.

A. Definition and Interpretations of Dominant Features

Modal analysis methods have been applied to dynamical systems to describe the system dynamics [31], [32]. The dominant eigenvalues of the system state matrix can reflect system stability [33] and are demonstrated to be characteristics of system events [27], [28]. These eigenvalues can be estimated by various approaches such as Prony analysis [34] and Matrix Pencil method [35]. We employ the Dynamic Mode Decomposition (DMD) method [31], due to its high efficiency and accuracy.

Let $\mathbf{y}_t \in \mathbb{R}^m$ contain the measurements of m PMU channels at discrete time t. Given any starting time t_0, define matrices

$$M_{t_0}^{t_0+T-1} = [\mathbf{y}_{t_0}, \mathbf{y}_{t_0+1}, \dots, \mathbf{y}_{t_0+T-1}], \text{ and} \tag{1}$$

$$M_{t_0+1}^{t_0+T} = [\mathbf{y}_{t_0+1}, \mathbf{y}_{t_0+2}, \dots, \mathbf{y}_{t_0+T}] \tag{2}$$

that contain sequential time series in a period of T. We assume $m > T$ for notational simplicity. Since there exists an operator $\mathcal{A}$ for the dynamical system satisfying $\mathbf{y}_i = \mathcal{A}\mathbf{y}_{i-1}$ [33], we have

$$M_{t_0+1}^{t_0+T} = \mathcal{A} M_{t_0}^{t_0+T-1}. \tag{3}$$

Note that the measurements can often be approximated by the output of a reduced order system. Specifically, let $\sigma_i, i = 1, \cdots, m$ denote the sorted singular values of $M_{t_0}^{t_0+T-1}$ in a decreasing order. Since only a few singular values are usually dominant, and most singular values are very small, $M_{t_0}^{t_0+T-1}$ can be approximated by a low-rank matrix [36]. In other words, for any fixed approximation ratio $\delta \in (0, 1)$, one can find an integer r much smaller than m and T such that

$$\frac{\Sigma_{j=1}^{r} \sigma_j}{\Sigma_{j=1}^{T} \sigma_j} \geq \delta \tag{4}$$

holds. Then $M_{t_0}^{t_0+T-1}$ can be approximated by a rank-r matrix $U_r \Sigma_r V_r^\dagger$ with a relative error of $1 - \delta$, measured in the Frobenius norm [28], i.e.,

$$M_{t_0}^{t_0+T-1} \approx U_r \Sigma_r V_r^\dagger \tag{5}$$

where $\Sigma_r = \text{diag}[\sigma_1, \cdots, \sigma_r] \in \mathbf{R}^{r \times r}$ collects the first r singular values in the diagonal entries, $U_r \in \mathbf{C}^{m \times r}$ and $V_r \in \mathbf{C}^{T \times r}$ contain the corresponding left and right singular vectors, and $\dagger$ denotes the conjugate transpose.

The projection of $\mathcal{A}$ onto the r-dimensional column subspace U_r, denoted by $\bar{\mathcal{A}}$, is defined as

$$\bar{\mathcal{A}} := U_r^\dagger \mathcal{A} U_r \approx U_r^\dagger M_{t_0+1}^{t_0+T} V_r \Sigma_r^{-1}, \tag{6}$$

where the approximate equality follows from (5). Then the r eigenvalues of $\bar{\mathcal{A}}$ are the same with the r dominant eigenvalues of $\mathcal{A}$ [31]. Let the eigendecomposition of $\bar{\mathcal{A}}$ be denoted by

$$\bar{\mathcal{A}} = R \Lambda L, \tag{7}$$

where $R = [\mathbf{r}_1, \cdots, \mathbf{r}_r] \in \mathbf{C}^{r \times r}$, $L = [\mathbf{l}_1, \cdots, \mathbf{l}_r] \in \mathbf{C}^{r \times r}$, and $\Lambda = \text{diag}(\lambda_1, \cdots, \lambda_r)$ contain the right eigenvectors, left eigenvectors, and eigenvalues, respectively.

Note that the eigenvalues of the state matrix govern the system dynamics. Events that occur spatially close and with the same event type may excite the same set of dominant eigenvalues. Although different initial conditions may lead to different time series, the underlying system remains the same. Thus, the dominant eigenvalues are representative features of the system dynamics. In fact, in our earlier work [27], [28], we showed that the dominant subspace of the spatial-temporal PMU data matrix depends entirely on the dominant eigenvalues, and events with different time series measurements may have the same subspace. Therefore, we use the dominant eigenvalues for the features in this paper.

Moreover, as shown in [27], [28], the significance of different types of events are different. For example, generator trip events usually lead to more severe fluctuations in the measurements than line trip events. Thus the corresponding singular values of the PMU data matrix are more significant in generator trip events than line trip events. In this paper, we also include the dominant singular values of the data matrix as the features.

B. Reducing the Impact of the First Event

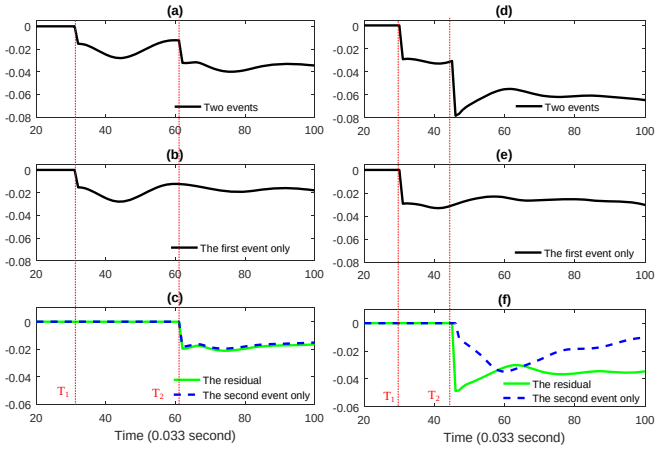


Fig. 3: (a) The voltage magnitude of bus 26 in a two-event case of a generator trip at bus 4 at T_1 and a line trip between buses 26 and 29 at T_2. (b) The second event only in (c), and the residual after T_2 by subtracting the impact of the first event. (c) The voltage magnitude of bus 25 in a two-event case of a generator trip at bus 8 at T_1 and a line trip between buses 25 and 54 at T_2, and of the second event only. (d) The second event only in (c), and the residual after T_2 by subtracting the impact of the first event.

Since we consider the case that the system is still under the disturbance caused by the first event when the second event happens, measurements after the second event is affected by the first event. We propose to predict the impact of the first event after the starting time of the second event and subtract it from the data. Fig. 3 illustrates the idea.

Fig. 3(a) shows the voltage magnitude of bus 26 (after subtracting the mean value) when a generator trip at bus 4 occurs at T_1 and a line trip between buses 26 and 29 occurs at T_2. Fig. 3(b) shows the voltage magnitude of the same first event with Fig. 3(a) but without the occurrence of the second event. Fig. 3(c) shows the residual of subtracting the time series when only the first event occurs from the time series of the two-event case. It also shows the time series when only the second event occurs for comparison. In this case, the residual time series matches the time series of the second event.

Fig. 3(d)-(f) shows a different case where the time series after subtracting the first event does not match the time series of the second event exactly. However, since we utilize dominant eigenvalues instead of time series, subtracting the impact the first event can still enhance the estimation of the dominant eigenvalues. For example, in the two-event case shown in Fig. 3(d) and (f), we compute the dominant eigenvalues of the time series that correspond to the second event only, both events, and the residual after subtracting the first event, respectively. We set $\delta = 0.999$, and the resulting approximate rank is at most 9 in these three cases. To compare the eigenvalues, we construct one vector that contains the real parts of the eigenvalues for each of these three cases. The difference of two vectors $\mathbf{x}$ and $\mathbf{y}$ are measured by the angle

$$\theta(\mathbf{x}, \mathbf{y}) = \cos^{-1}(\mathbf{x}^T\mathbf{y}/\|\mathbf{x}\|\|\mathbf{y}\|), \tag{8}$$

and a smaller angle corresponds to a stronger similarity between $\mathbf{x}$ and $\mathbf{y}$. The angle between the eigenvalues of the second event only and both events is $35°$, while the angle is reduced to $25°$ between the second event only and the residual. Thus, the subtraction increases the estimation accuracy of the dominant eigenvalues.

IV. Feature Extraction of the Second Event in Real Time

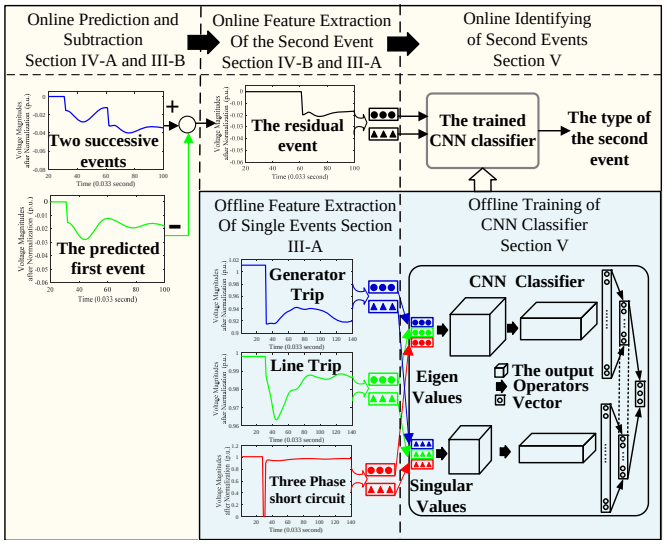


Fig. 4: The offline training and online testing of the proposed method

Fig. 4 visualizes our proposed approach for event identification. It includes the offline training of a CNN classifier and an online identification procedure. To identify the type of the second event in real-time, the impact of the first event needs to be reduced with the "Prediction and Subtraction" process, which is introduced in Section III-B and implemented in Section IV-A. The next step is to extract the dominant features from the residual measurements as described in Section III-A and implemented in Section IV-B. These extracted features become the inputs of the trained CNN classifier, the details of which are discussed in Section V, to determine the type of the second event. Notice that this CNN classifier is trained offline using the same method presented in Section III-A to extract features as inputs.

A. Online Prediction and Subtraction

Assume that the first event occurs at time T_1 and the second event occurs at time T_2. The measurements during T_1 to T_2, denoted by the matrix $M_{T_1}^{T_2}$, are affected by the first event, and $M_{T_2}^{T_2+\tau}$ are affected by both events. We first predict the measurements after T_2 using $M_{T_1}^{T_2}$ if the second event did not occur. Various methods can be employed for data prediction, such as time series analysis [37] and a data-driven approach using the low-rank Hankel matrix [38]. Here we employ the prediction technique based on the dominant eigenvalues.

Suppose the system evolves under the operator $\mathcal{A}$ after T_1. We estimate the eigenvalues of $\mathcal{A}$ using the DMD method following (5) to (7), where we select t_0 and T such that $t_0 \geq T_1$ and $t_0 + T \leq T_2$. Then compute

$$\Psi = U_r R, \tag{9}$$

where Ψ can be written as $\Psi = [\psi_1, \ldots, \psi_r]$, and $\psi_i \in \mathbf{C}^m$. Compute the inverse eigenvector matrix Φ,

$$\Phi = (\Psi^\dagger \Psi)^{-1}\Psi^\dagger, \tag{10}$$

where Φ can be represented by $[\phi_1^T; \cdots; \phi_r^T]$ with $\phi_i \in \mathbf{C}^m$. Then $\mathcal{A}$ can be approximated by

$$\mathcal{A} \approx \Sigma_{i=1}^r \psi_i \lambda_i \phi_i, \tag{11}$$

and $M_{T_1}^{T_2}$ can be approximated by

$$M_{T_1}^{T_2} \approx \Psi \text{diag}(\alpha)\Gamma, \tag{12}$$

where $\alpha = [\alpha_1; \ldots; \alpha_r]$, $\alpha_i = \phi_i^\dagger \mathbf{y}_{T_1}$, and $\Gamma \in \mathbf{R}^{r\times(T_2-T_1)}$ is a Vandermonde matrix with its ith row being $\gamma_i = \left[\lambda_i, \lambda_i^2, \ldots, \lambda_i^{T_2-T_1}\right]$ [39].

If the second event did not happen, the system would keep evolving under $\mathcal{A}$. Then the measurements from time T_2 to $T_2 + \tau$ for some positive τ were only affected by the first event and could be represented by the matrix $\bar{M}_{T_2}^{T_2+\tau}$,

$$\bar{M}_{T_2}^{T_2+\tau} \approx \Psi \text{diag}(\alpha)\Gamma', \tag{13}$$

where Γ' is a Vandermonde matrix with its ith row being $\gamma_i' = \left[\lambda_i^{T_2+1}, \ldots, \lambda_i^{T_2+\tau}\right]$.

After the second event happens at T_2, the system evolves under a different operator, denoted by $\mathcal{A}'$. Since $M_{T_2}^{T_2+\tau}$ contains the actual measurements after the second event starts, we subtract the impact of the first event from $M_{T_2}^{T_2+\tau}$ and obtain the residual

$$\tilde{M}_{T_2}^{T_2+\tau} = M_{T_2}^{T_2+\tau} - \bar{M}_{T_2}^{T_2+\tau}. \tag{14}$$

Then residual $\tilde{M}_{T_2}^{T_2+\tau}$ is mainly governed by the dominant eigenvalues excited by the second event.

The prediction accuracy depends on the accuracy of the estimated eigenvalues λ_i's. We first evaluate the accuracy of λ_i's by reconstructing data between T_1 and T_2 using λ_i's.

Let $\bar{M}_{T_1}^{T_2}$ denote the reconstructed data, and the reconstruction error $\epsilon_{T_1}^{T_2}$ is defined as,

$$\epsilon_{T_1}^{T_2} = \frac{\|\bar{M}_{T_1}^{T_2} - M_{T_1}^{T_2}\|_F}{\|M_{T_1}^{T_2}\|_F} \tag{15}$$

If $\epsilon_{T_1}^{T_2}$ is smaller than a predefined threshold ϵ_0, the estimated eigenvalues $\lambda_i, i = 1, \cdots, r$ are sufficiently accurate and can be used to predict $M_{T_2}^{T_2+\tau}$. If $\epsilon_{T_1}^{T_2}$ is larger than ϵ_0, then some estimated eigenvalues are not accurate. The inaccuracy is due to the limitations of the DMD method. When the time of the measurements is short, the slow modes[1] have larger estimation errors [40]. Thus we set the modes slower than a predetermined frequency f_0 to be zero to reduce the prediction errors in $\bar{M}_{T_2}^{T_2+\tau}$.

B. Online Feature Extraction

We next compute the dominant features of the second event from $\tilde{M}_{T_2}^{T_2+\tau}$. Let $\tilde{\sigma}_1, ..., \tilde{\sigma}_r$ denote the dominate r singular values of $\tilde{M}_{T_2}^{T_2+\tau}$ with a fixed approximation ratio of δ. We apply the DMD method in (5) to (7) to estimate the dominant eigenvalues of $\mathcal{A}'$. $M_{t_0}^{t_0+T-1}$ and $M_{t_0+1}^{t_0+T}$ in (1)-(2) correspond to two consecutive data blocks in $\tilde{M}_{T_2}^{T_2+\tau}$. The resulting ith eigenvalue, denoted by $\tilde{\lambda}_i$, can be viewed as $\tilde{\lambda}_i = e^{(\rho_i + j2\pi f_i)\Delta t} = \Re(\tilde{\lambda}_i) + j\Im(\tilde{\lambda}_i)$, where ρ_i, f_i are the ith damping ratio and oscillation frequency, respectively, and Δt is the sampling period. Since $\Re(\tilde{\lambda}_i)$ is the rotation transformation of $\Im(\tilde{\lambda}_i)$, and both $\Re(\tilde{\lambda}_i)$ and $\Im(\tilde{\lambda}_i)$ depend on ρ_i and f_i, we utilize $\Re(\tilde{\lambda}_i)$ instead of $\tilde{\lambda}_i$ as the feature to avoid the computation of complex numbers. Thus, we employ

$$\xi = [\Re(\tilde{\lambda}_1), \cdots, \Re(\tilde{\lambda}_r)]^T \in R^r, \quad \text{and} \tag{16}$$

$$\zeta = [\tilde{\sigma}_1, \cdots, \tilde{\sigma}_r]^T \in R^r \tag{17}$$

to represent the features of the second event.

V. Classification Through CNN

One challenge of classifying successive events is that large-scale successive training datasets are not available, and most historical datasets are single events. Thus, we will train a classifier on recorded single events to identify each of the successive events. Popular choices of nonlinear classifiers include support vector machine (SVM) [17], [41], Recurrent neural network (RNN) [42], and CNN. The performance of SVM depends on the proper selection of a kernel [41]. RNN has a sequential structure to compute the weighted combination of all inputs, and CNN has a hierarchical structure to capture the most informative values of the input. We compared the performance of CNN and RNN on a large number of numerical experiments, and the results indicate that CNN is more suitable for our problem.

[1]"slow modes" means the eigenvalues $\lambda = \exp(\rho + j2\pi f)\Delta t$ of the discrete model have small oscillation frequency f.

A. A General CNN Classifier

A typical CNN for classification has n_c convolutional layers, n_c Rectified linear units (ReLU) layers, n_c pooling layers, a fully connected layer and an output layer. The input is a 3-dimensional volume $Z \in \mathbf{R}^{w \times h \times d}$ with width w, height h and depth d. The output is a vector of class scores, and the class with the highest score indicates the type of the event.

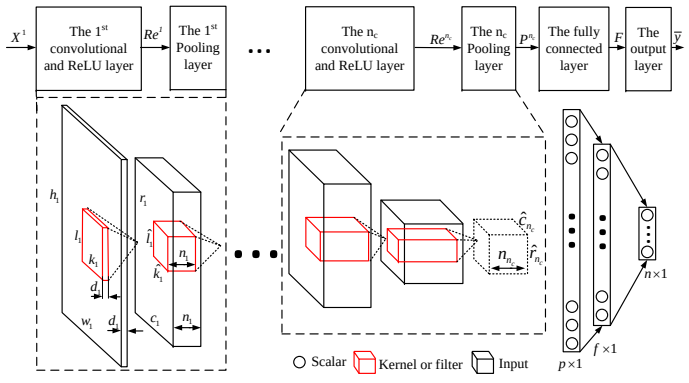


Fig. 5: The structure of a general CNN classifier

Let $X^i \in \mathbf{R}^{w_i \times h_i \times d_i}$ denote the ith input of the convolutional layer, and $X^1 = Z$ for the first layer. Let $W^{i,j} \in \mathbf{R}^{k_i \times l_i}$ and $B^{i,j} \in \mathbf{R}^{n_i}, j = 1, \cdots, n_i$ be the jth kernel and bias for the ith layer, respectively. Each kernel is slided on the input to perform the dot product in the overlapping part. The kernel is moved along the width and height directions of X^i until all values of X^i are reached. The step size is determined by a hyperparameter "stride" s_i. If one dimension of X^i is smaller than the corresponding dimension of the kernel, users can pad zeros to the border of X^i to match the size of the kernel, and a hyperparameter "zero-padding" p_i determines the number of padded zeros. All the convolution results of these n_i kernels are stacked together into an output volume $C^i \in \mathbf{R}^{c_i \times r_i \times n_i}$, where $c_i = (w_i - k_i + 2p_i)/s_i + 1$, and $r_i = (h_i - l_i + 2p_i)/s_i + 1$. $C_{c,r,m}^i$ with $1 \le c \le c_i$, $1 \le r \le r_i$, and $1 \le m \le n_i$ can be computed from

$$C_{c,r,m}^i = \Sigma_{k=1}^{k_i} \Sigma_{l=1}^{l_i} \Sigma_{d=1}^{d_i} X_{i_x, j_x, d}^i W_{k,l,d}^{i,m} + B_{c,r,m}^i, \tag{18}$$

where $i_x = (c-1)s_i + k$, and $j_x = (r-1)s_i + l$. Then C^i is fed to the ith ReLU layer to increase the sparsity, and the output of the ReLU layer is

$$Re^i = \max(C^i, 0), \tag{19}$$

where $\max(\cdot)$ is performed on each element of C^i.

The maximum pooling layer can reduce the size of Re^i further. Let the size of the pooling filter be $\hat{k}_i \times \hat{l}_i$. We move the filter along the width and height directions of Re^i at each depth slice[2] independently with the stride $\hat{s}_i$. The output has $\hat{c}_i = (c_i - \hat{k}_i)/\hat{s}_i + 1$ columns, $\hat{r}_i = (r_i - \hat{l}_i)/\hat{s}_i + 1$ rows, and the same depth n_i. Let $P_{c,r,m}^i$ be the cth column, rth row, and mth depth of the output of the pooling layer, where $1 \le c \le \hat{c}_i, 1 \le r \le \hat{r}_i, 1 \le m \le n_i$, $i_r = (c-1)\hat{s}_i + k, j_r = (r-1)\hat{s}_i + l$, then we have

$$P_{c,r,m}^i = \max_{1 \le k \le \hat{k}_i, 1 \le l \le \hat{l}_i} Re_{i_r, j_r, m}^i. \tag{20}$$

[2]A depth slice is defined as a two dimensional slice of a certain depth [23].

P^i then becomes the input of the $(i+1)$th convolutional layer, i.e., $X^{i+1} = P^i$. The output of the n_cth pooling layer is reshaped into a vector $Q \in \mathbf{R}^p$ with $p = \hat{c}_{n_c} \times \hat{r}_{n_c} \times n_{n_c}$. Q is the input of the fully connected layer, and the output is

$$F = \max((W^f)^T Q + B^f, 0), \tag{21}$$

where the parameter matrices $W^f \in \mathbf{R}^{p\times f}, B^f \in \mathbf{R}^f$ are the parameters to train.

The output class scores $\bar{\mathbf{y}} \in \mathbf{R}^n$ are computed from

$$\bar{\mathbf{y}} = g((W^o)^T F + B^o), \tag{22}$$

where $W^o \in \mathbf{R}^{f\times n}$ denotes the output parameter matrix, and $B^o \in \mathbf{R}^n$ denotes an output bias matrix. $g(\cdot)$ is an output function and is often selected as a sigmoid function $g(x) = \frac{1}{(1+e^{-x})}$ when $n = 2$ or a softmax function $g(x) = \frac{e^x}{1+e^x}$ when $n \geq 2$. $\bar{\mathbf{y}}$ includes n scores of all classes for the input Z, and the highest score is the classified class of Z.

The training dataset $\mathbb{D} = \{Z^s \in \mathbf{R}^{w\times h\times d}, y^s \in \mathbf{R}^n, s = 1, \cdots, N\}$ contains N pairs of event data and the corresponding labels. y^s is a binary vector which only has the jth entry to be 1 if Z^s belongs to class j. When the input to the CNN classifier is Z^s, the output class score for Z^s is $\bar{\mathbf{y}}^s$. To train a suitable parameter set $\Theta = \{W^i \in \mathbf{R}^{k_i\times l_i\times d_i\times n_i}, B^i \in \mathbf{R}^{c_i\times r_i\times n_i\times n_i}, i = 1, \cdots, n_c, W^f, B^f, W^o, B^o\}$, we minimize the cross-entropy loss function, and the optimal parameter set Θ can be computed as follows,

$$\Theta = \arg\min_{\Theta} \frac{1}{N} \Sigma_{s=1}^N \Sigma_{i=1}^n \mathbf{y}_i^s \log \bar{\mathbf{y}}_i^s + \lambda \|\Theta\|_F^2, \tag{23}$$

where λ is the regulation coefficient to avoid over-fitting. Since (23) requires solving a non-convex optimization problem, finding the global optimum is an open issue. Stochastic gradient decent methods such as Adam [43] and RMSprop [44] with a decay coefficient γ and a learning rate α are widely applied. Some experiential techniques can be applied to enhance the numerical performance. For example, the "early stop" method stops iterations if the loss function does not reduce for l^* consecutive times, and the "batch normalization" normalizes the output of each convolutional layer to reduce the covariate shift effects [45].

B. Our CNN Classifiers

As the dominant features include the eigenvalues and singular values, we design a two-path CNN classifier, referred to as CNN-F, with each path having two layers, and these two paths are concatenated in the fully connected layer, as shown in Fig. 6 (a). The top path is for the eigenvalues, and the bottom one for the singular values. To compare the performance, we also design a classifier, referred to as CNN-T, that is trained on time series directly, as shown in Fig. 6 (b). The inputs of each layer of these CNN are visualized as blocks, and their sizes are labeled by the notations. The superscript $'$ in the bottom path of CNN-F and $''$ of CNN-T are marked to avoid confusion. The specific values of these parameters in simulation are summarized in Section VI-A. Each solid arrow represents a collection of a convolution layer, a ReLU layer, and a pooling layer. A solid arrow is an abstraction of a dotted block in Fig. 5.

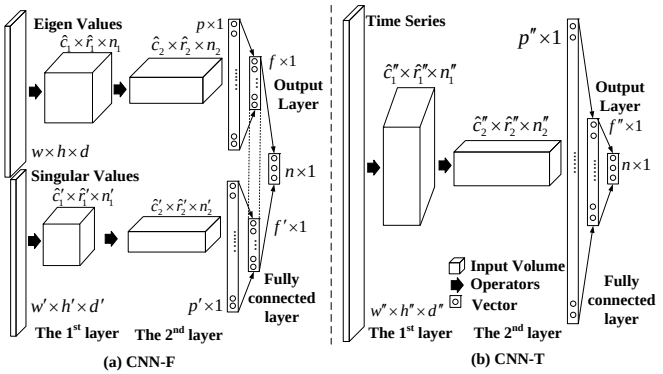


Fig. 6: CNN Classifier CNN-F and CNN-T

The total number of parameters of CNN-F and CNN-T are denoted by N_F and N_T respectively, where

$$N_F = \Sigma_{i=1}^2 (n_i l_i k_i + n_i' l_i' k_i' + n_i c_i r_i + n_i' c_i' r_i') + pf + p'f' + (f + f')n, \text{ and} \tag{24}$$

$$N_T = \Sigma_{i=1}^2 (n_i'' l_i'' k_i'' + n_i'' c_i'' r_i'') + p''f'' + f''n. \tag{25}$$

VI. Numerical Results

As the power system is influenced by large disturbances like line faults and some abnormal perturbations of the control inputs [46], [47], we consider four types of events, including line trips (LTs), generator trips (GTs), three-phase short circuit (TPs), and generator reference-voltage changes (GRCs) as the first events. The classifiers CNN-F and CNN-T are trained by these four types of single events, and then tested on various successive events with different time intervals between events.

A. Simulation setup and computational complexity

We generated 1087 single events through PSSE [29] in the IEEE-68 bus power system (Fig. 1), which has 16 generators, five areas, and 64 branches. PSSE has the dynamic simulation functions of *disturbances*. These functions include *line fault* that can be used for generating TP events, *trip line* for LT events, *disconnect machine* for GT and *change reference-voltages* for GRC events. TPs are cleared after 0.1 seconds, and the reference-voltage changes in the range of $0-0.1$ p.u. Notice that these functions have the corresponding Application Program Interfaces (API) based on Python that can automatically generate a large number of datasets. The initial conditions are varied by changing power injections of some buses or randomly disconnecting some lines. The data rate is 30 samples per second. We use one second of data to identify an event, and thus $\tau = 30$. 70% of these datasets are used for training data and 30% for validation. A total number of 1808 testing two-event datasets are summarized in Table I, where "LT+GT" means a line trip event is followed by a generator trip event. For successive events, the first event occurs at $T_1 = 1$ second, the second event occurs at $T_2 = T_1 + \Delta T$, ΔT varies from 0.5 to 2 seconds. The $\epsilon_0 = 0.5$, and the estimated eigenvalues less than $f_0 = 1Hz$ are defined as slow modes. The identification performance is measured by the identification accuracy rate (**IAR**), which is the ratio of the number of correctly classified events to the total number of events.

Table I: Total Number of the Testing Datasets

Types	LT+GT	LT+TP	LT+LT	GT+GT	GT+TP
Number	169	347	378	107	117
Types	TP+GT	GRC+LT	GRC+GT	GRC+TP	Total
Number	146	188	120	236	1808

Table II: Parameters Values of CNN-F and CNN-T

Parameter	(k_i, l_i, d_i)	(c_i, r_i, n_i)	(k_i', l_i', d_i')	(c_i', r_i', n_i')
$i=1$	(2, 1, 4)	(8, 3, 4)	(2, 1, 4)	(8, 2, 4)
$i=2$	(2, 1, 8)	(3, 3, 8)	(2, 1, 8)	(3, 2, 8)
Parameter	(k_i'', l_i'', d_i'')	(c_i'', r_i'', n_i'')	$(\hat{k}_i, \hat{l}_i)$	$(\hat{k}_i', \hat{l}_i')$
$i=1$	(5, 5, 8)	(105, 13, 8)	(2, 1)	(2, 1)
$i=2$	(3, 3, 16)	(26, 3, 16)	(2, 1)	(2, 1)
Parameter	$(\hat{k}_i'', \hat{l}_i'')$	$(s_i, \hat{s}_i)$	$(s_i', \hat{s}_i')$	$(s_i'', \hat{s}_i'')$
$i=1$	(2, 2)	(1, 2)	(1, 2)	(2, 2)
$i=2$	(2, 2)	(1, 2)	(1, 2)	(2, 2)
Parameter	(p, f)	(p', f')	(p'', f'')	n
-	(72, 16)	(48, 16)	(416, 16)	4

Extraction of features. δ is set to be 0.999. The approximate rank r is no greater than 9 for all the simulated datasets for the approximation error $1-\delta$. Following (14), we compute the residual matrices of voltage magnitudes $\tilde{M}^v \in \mathbf{R}^{m\times\tau}$, frequency $\tilde{M}^f \in \mathbf{R}^{m\times\tau}$ and active power measurements $\tilde{M}^p \in \mathbf{R}^{l\times\tau}$, where m is the bus numbers, and l is the number of lines. $m=67$ and $l=80$ in the simulation. We compute the eigenvalues of the state matrix using voltage magnitudes, frequency, and active power measurements, respectively. The resulting eigenvalues are stacked together to form a matrix $\Xi = [\xi_v, \xi_f, \xi_p] \in \mathbf{R}^{9\times 3}$, where 9 is an upper bound of the approximate rank r in different datasets. We also compute the singular values of $\tilde{M}^v \in \mathbf{R}^{m\times\tau}$ and $\tilde{M}^f \in \mathbf{R}^{m\times\tau}$ and form matrix $B = [\zeta_v, \zeta_f] \in \mathbf{R}^{9\times 2}$. Note that when r is less than 9, some eigenvalues and singular values in Ξ and B are set to zero.

Training of CNN. CNN-F takes Ξ and B as the input. The corresponding parameters of the input size are $w=9$, $h=3$, $d=1$, $w'=9$, $h'=2$, and $d'=1$. CNN-T takes the time series as the input, denoted by matrix $M_{vfp} = [\tilde{M}^v; \tilde{M}^f; \tilde{M}^p] \in \mathbf{R}^{214\times 30}$. Then $w''=214$, $h''=30$, and $d''=1$.

Table II records the actual values of the parameters of CNN-F and CNN-T in our simulation. Then the total number of parameters of CNN-F and CNN-T are computed following (24) and (25), and $N_F = 2344$, $N_T = 19216$. Since the number of parameters of CNN-T is eight times larger than that of CNN-F, the training complexity and the memory requirement is significantly reduced in CNN-F by training on the features instead of the time series.

CNNs are trained offline. $\alpha = 0.001$, $\lambda = 0.001$, and the RMSprop optimizer with a decay coefficient $\gamma = 0.9$ is adopted. The "Xavier" initialization [48], batch normalization and an early stop with $l^* = 12$ are applied to improve the performance. We choose different initialization values of CNN parameters and obtain multiple sets of parameters. The resulting CNNs are evaluated on 10% of the two-event datasets, and the best one is selected as the trained model.

In the online identification of the second event, we first predict and subtract the impact of the first event. We then extract the features and apply the trained classifier. On a desktop computer with Intel Core i7, 3.60 GHz inner storage and 32.0 GB memory, classifying one testing dataset takes 4 millisecond (ms), and the process of prediction, subtraction and feature extraction takes 3 ms.

B. Performance of Classifying the First Event

Table III: Performance of CNN-F, CNN-T based on identifying the first events.

IAR (%)	LT	GT	TP	GRC	Averaged
CNN-F	100.0	98.0	100.0	100.0	99.3
CNN-T	99.3	100.0	100.0	100.0	99.7

Classifying the first event is as the same as identifying single events. Both the CNN-F and CNN-T can identify the first events with IAR greater than 90% as shown in Table III.

C. Performance of Classifying the Second Event

Table IV: Performance of CNN-F and CNN-T based on classifying the second events of 1627 two-event test cases with different processes. The time interval $\Delta T = 0.5 \sim 2$ second.

Second Event Type	Time Interval ΔT	CNN-F		CNN-T	
		NS	SP	NS	SP
LT	0.5	82.9	**92.6**	71.8	51.8
GT	0.5	70.9	**93.0**	79.1	75.0
TP	0.5	**95.9**	93.6	87.2	60.1
Overall	0.5	85.5	**93.1**	80.1	61.0
LT	1	77.7	**94.4**	83.3	60.4
GT	1	77.7	**95.1**	77.7	69.4
TP	1	94.7	**95.3**	84.3	72.8
IAR	1	84.7	**95.0**	81.9	67.7
LT	1.5	88.2	**95.6**	67.6	70.5
GT	1.5	78.2	**85.4**	95.1	70.1
TP	1.5	**95.4**	94.7	93.6	90.1
Overall	1.5	88.2	**92.4**	85.9	78.2
LT	2	87.5	**94.8**	65.6	60.5
GT	2	76.2	**84.3**	68.7	70.6
TP	2	98.1	**98.7**	87.9	79.5
Overall	2	87.4	**92.6**	74.7	70.8

Table V: Performance of CNN-F and CNN-T based on classifying the second events with mild and severe first events.

First Event Type	CNN-F		CNN-T	
	NS	SP	NS	SP
LT or GRC	91.8	**93.8**	82.4	75.7
GT or TP	76.3	**90.6**	69.2	49.6

We next evaluate the identification performance of the second event in successive events. We use the prediction and subtraction approach to compute the residual in (14) and extract the features in (16)-(17).

To validate the effectiveness of feature extraction and the prediction-subtracting process of the first event, we compare CNN-F and CNN-T on measurements with and without subtracting the impact of the first event. "**Not Subtract**

(NS)" means using the measurements $M_{T_2}^{T_2+\tau}$ after time T_2. "**Subtract the Prediction (SP)**" means using the residual $\tilde{M}_{T_2}^{T_2+\tau}$ in (14) after subtracting the first event. CNN-T is trained on these measurements directly, while CNN-F is trained on the extracted features of these measurements. We test CNN-F and CNN-T on 1627 two-event testing cases with varying time intervals $\Delta T = 0.5 \sim 2$ seconds. The IARs of CNN-F and CNN-T with NS and SP processes are summarized in Table IV.

The IARs of the proposed CNN-F with SP process achieve the best overall performance. The IARs of CNN-F are better than that of CNN-T in most cases, indicating that feature extraction can improve identification accuracy. For CNN-F, the IARs with SP are mostly better than those with NS and have smaller variations when ΔT changes. For CNN-T, the IARs with NS is better than those with SP as the time series are sensitive to the inaccuracy of the prediction inaccuracy. Hence, the CNN-F combined with SP improves the identification accuracy rate and the robustness to varying time intervals.

In addition, to illustrate the effect of the SP process when the first events are mild or serve, we summarize the second event identification results in Table V, where the LT and GRC are treated as "mild events," and GT and TP as "severe events". When the first events are relatively mild, the IAR of CNN-F with SP has a 2% improvement over that with NS, and at least 11.4% higher than CNN-T. When the first events are serious, the increase is more obvious. The IAR of CNN-F with SP has a 14.3% improvement over that with NS and is more than 20% higher than CNN-T.

D. Separability of Multiple Event Types

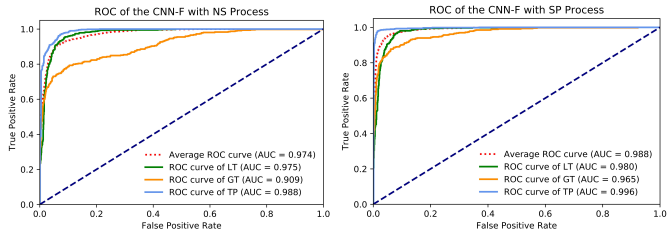


Fig. 7: Receiver Operating characteristics (ROC) of the CNN-F with NS (a) and SP (b) processes

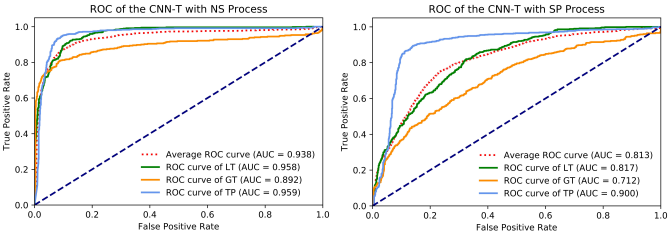


Fig. 8: Receiver Operating characteristics (ROC) of the CNN-T with NS (a) and SP (b) processes

The Received Operating Characteristics (ROC) curves and the Area Under the ROC Curve (AUC) can determine the capability of a classifier to distinguish multiple classes. ROC was initially introduced for signal detection [49], and evaluation of binary classification performance and then was extended to multi-classification problems [50], [51]. ROC measures the True Positive Rate (TPR) against the False Positive Rate (FPR). ROC for type i is desirable to be far away from the diagonal line, indicating a strong separability of type i. The AUC $\in [0, 1]$ reveals the capability of the classifier to distinguish multiple classes. A larger AUC demonstrates stronger separability of different classes.

The ROC and the AUC of the classifiers CNN-F with different processes are drawn in Fig. 7. The ROC of each type is calculated by assuming type i as the positive class while all others as the negative class [52]. Let TP, FN, FP, TN denote the True Positive number, False Negative number, False Positive number, and True Negative number. Then the average ROC is TPR_{aver} against FPR_{aver}, where

$$\text{TPR}_{\text{aver}} = \frac{\Sigma_{i=1}^{n} \text{TP}_i}{\Sigma_{i=1}^{n} (\text{TP}_i + \text{FN}_i)}, \tag{26}$$

$$\text{FPR}_{\text{aver}} = \frac{\Sigma_{i=1}^{n} \text{FP}_i}{\Sigma_{i=1}^{n} (\text{FP}_i + \text{TN}_i)}, \tag{27}$$

n is the total number of types, and TP_i denotes the TP of the ith type. In Fig. 7 (a), the AUC of TP is much larger than that of GT, indicating CNN-F with NS has a stronger capability of identifying TP than GT events. The ROCs of different types in Fig. 7 (b) demonstrate the separability of CNN-F since the AUCs are all close to 1. Comparing Fig. 7 (a) and (b), the AUCs with SP for each type are larger than that with NS, indicating the effectiveness of SP for CNN-F. The average AUCs of CNN-F in Fig. 7 are more significant than that of CNN-T with the same process in Fig. 8, thus the overall separability of CNN-F is better.

E. Performance with a Partial Training Dataset

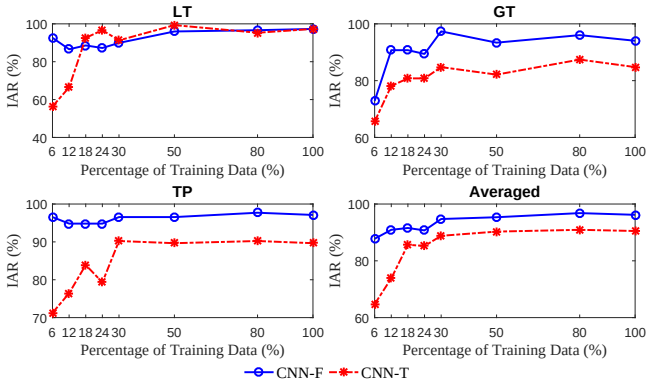


Fig. 9: Performances of CNN-F and CNN-T when partial training datasets are available

Since the practical recorded events might not include all the possible event locations and pre-event conditions, we study the impact on the identification performance when the training set only has a small number of recorded events. We compare the best CNN-F and CNN-T classifiers when randomly selecting $6\% \sim 100\%$ of the events in the training dataset to train the classifier. The results in Fig. 9 indicate that the IAR of CNN-F is much higher than that of CNN-T on average. CNN-T is sensitive to the size of the training dataset, while the IAR of

CNN-F does not change much even when the size of training set varies significantly. When the size of the training is reduced to only 12%, the IAR of CNN-F is still above 90%, while the IAR of CNN-T is less than 70%. CNN-F is trained on the dominant features of the time series, and these features are robust to pre-event initial conditions. Different time series may correspond to similar dominant features. Thus, the required number of training events is significantly reduced in CNN-F.

F. Robustness to the Time Delay in Event Detection

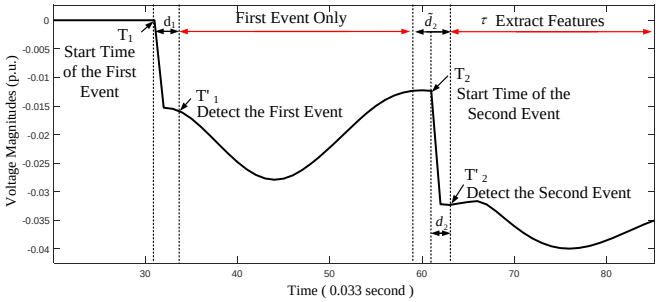


Fig. 10: Sequence of events with detection delay

Table VI: The IAR of CNN-F tested on the second events with time interval ΔT = 1 second when the start time of the first and the second events has d_1 and d_2 ms of time delay

(d_1, d_2)	(200, 33)	(200, 100)	(200, 133)	(200, 167)
LT	92.4	93.1	84.0	80.6
GT	88.9	94.4	95.8	92.4
TP	94.8	97.1	–	–
(d_1, d_2)	(300, 33)	(300, 100)	(300, 133)	(300, 167)
LT	93.8	95.1	93.8	88.9
GT	88.2	90.3	96.5	97.2
TP	96.5	97.1	–	–
(d_1, d_2)	(500, 33)	(500, 100)	(500, 133)	(500, 167)
LT	81.9	81.9	77.8	72.2
GT	84.7	92.4	93.1	85.4
TP	97.7	97.7	–	–

Since a delay may exist between the start time of an event and the time that the event is detected, depending on the types of the detection methods, we study the robustness of the proposed method to the delay in detecting an event. As shown in Fig. 10, T_1 is the time that the first event occurs. T_1' is the time the first event is detected. The delay is $d_1 = T_1' - T_1$. Let T_2 and T_2' denote the time the second event occurs and is detected, respectively. Our approach has an estimated detection delay $\tilde{d}_2$ that shall be greater than the actual delay d_2. Then $T_2' - \tilde{d}_2$ is less than T_2. The time window of "first event only" is T_1' to $T_2' - \tilde{d}_2$. Our prediction and subtraction process uses measurements in this window and applies the method in Section IV-A to predict the impact of the first event in time window T_2' to $T_2' + \tau$. Then the features of the second event are extracted from the residual matrix $\tilde{M}_{T_2'}^{T_2'+\tau}$, which is obtained by subtracting the prediction $\bar{M}_{T_2'}^{T_2'+\tau}$ from the measured time series $M_{T_2'}^{T_2'+\tau}$.

The robustness of the CNN-F to the time delay d_i, $i = 1, 2$ from 33 to 500 ms is validated on the two-event datasets of 1 second time intervals. Here we set $\tilde{d}_2 = 2d_2, \delta = 0.98$, and the results are summarized in Table VI, where 33 ms is the time between two PMU measurements with a sampling rate of 30 samples per second. Notice that since three-phase short circuit events are usually cleared within $0.1 \sim 0.2$ second [53], we assume that such faults are detected before the clearance and $d_2 \leq 0.1$ second for TP events. In Table VI, "–" denotes "not applicable" when $d_2 > 0.1$ second. When the delay d_1 and d_2 both exist, the IARs of the second events are mostly higher than 85% when $d_2 < 167$ ms and $d_1 < 500$ ms.

G. Comparison of Different Classifiers

Next, we compare the CNN with NN and multi-class SVM (MSVM). Prediction-subtraction and feature extraction processes apply to all these classifiers. The IARs of the CNN, NN, and MSVM are compared in Table VII, where "-F" denotes with the extracted features as the input. Notice that the structure of NN-F is similar to CNN-F, and the main difference is to replace the convolutional layers with the fully connected layers in NN for each path. The weight and bias parameters of the first layer of NN-F are $W_{\text{NN}}^1 \in R^{27\times 13}, B_{\text{NN}}^1 \in R^{13}$ for the top path and $\hat{W}_{\text{NN}}^1 \in R^{18\times 9}, \hat{B}_{\text{NN}}^1 \in R^9$ for the bottom path. In the second layer $W_{\text{NN}}^2 \in R^{13\times 6}, B_{\text{NN}}^2 \in R^6$ for the top path, and $\hat{W}_{\text{NN}}^2 \in R^{9\times 4}, \hat{B}_{\text{NN}}^2 \in R^4$ for the bottom path. The parameters for the fully connected layer before the output layer are $W_{\text{NN}}^f \in R^{4\times 2}, B_{\text{NN}}^f \in R^2, \hat{W}_{\text{NN}}^f \in R^{4\times 2}, \hat{B}_{\text{NN}}^f \in R^2$ for the top and bottom paths, and the output layer has the parameters $W_{\text{NN}}^o \in R^{4\times 4}, B_{\text{NN}}^o \in R^4$. The same RMSprop optimizer is employed to train the NN-F. The other hyper-parameters include $\lambda = 0.001, \gamma = 0.9, \alpha = 0.001$, which are the same as that of CNN-F. The MSVM classifier is the extension of the binary-class SVM with the coupling pairwise method [54].

Table VII: Performances of CNN-F, NN-F, MSVM-F of classifying the second event. 1627 two-event test cases. $\Delta T = 0.5 \sim 2$ second.

Classifier	CNN-F		NN-F		MSVM-F	
Process	NS	SP	NS	SP	NS	SP
LT	82.5	**93.5**	76.1	80.3	53.6	54.2
GT	78.2	**89.2**	76.7	87.5	77.5	75.6
TP	97.1	95.8	**97.3**	92.0	67.5	66.6
Overall	86.9	**93.1**	84.0	87.8	65.3	65.2

The results in Table VII indicate that the CNN-F achieves the highest overall IARs among these three classifiers. With the SP process, the IARs of NN-F and CNN-F have more than 3% and 6% improvement respectively. The IARs of MSVM-F are less than 80% with both NS and SP processes.

H. Robustness to Noise

Table VIII: IAR of the second event by CNN-F with noisy measurements

SNR (dB)	40	50	60	70	80	90	100
IAR of LT (%)	91.2	88.2	90.2	90.2	92.1	96.1	96.1
IAR of GT (%)	73.1	75.9	73.1	80.5	82.4	79.6	80.5
IAR of TP (%)	91.9	94.6	94.6	96.4	96.4	97.3	97.3
Overall IAR (%)	85.4	86.3	86.0	89.1	90.4	91.0	91.3

The signal-noise-ratio (SNR) of PMU data vary in different regions. It is reported that the SNR of recorded PMU data is

45 dB in New Mexico [55], and 87 dB in New England [28]. Thus we study the sensitivity of CNN-F when the SNR varies from 40 dB to 100 dB. Gaussian noise with zero mean and a standard deviation selected according to the SNR is added to both the training and the testing datasets.

As shown in Table VIII, the overall IAR of CNN-F is more than 80% for different noise levels, and the IARs are more than 90% when the SNR is higher than 80 dB. Significant events such as TP events are less sensitive to noise, and the IAR of TP events are always more than 90% with different SNRs.

VII. Conclusions

Identifying overlapping successive events in power systems is crucial to prevent cascading failures at an early stage. To efficiently identify events within seconds, this paper proposes a two-path shallow CNN classifier with extracted features as input, referred to as CNN-F. Training on the extracted features, including the dominant eigenvalues and singular values of the post-event time series, instead of time series measurements, significantly reduces the number of training datasets, the complexity of the classifier, and the training time.

Successive events do not occur very frequently and are insufficient for training. The proposed CNN-F is trained using single events. When the earlier events impact the subsequent ones, "Subtract the Prediction (SP)" process is proposed to improve the performance of CNN-F.

Compared with the classifier trained on the time series directly, the proposed CNN-F with SP requires a smaller number of training datasets, reaches higher identification accuracy for the second events, and has stronger separability of multiple event types. The robustness of CNN-F to delays in event detection and noise are validated by the simulated data in the IEEE 68-bus power system. Moreover, CNN-F outperforms the NN and SVM classifiers of a similar size.

The proposed method can be extended to identify more than two events when a short time window is available after each event to extract features. Future work includes identifying multiple events occurring almost simultaneously, or events of more types, such as capacitor banking, asymmetric faults.

References

[1] P. Henneaux, P.-E. Labeau, and J.-C. Maun, "A level-1 probabilistic risk assessment to blackout hazard in transmission power systems," *Reliability Engineering & System Safety*, vol. 102, pp. 41–52, 2012.

[2] I. Dobson, B. Carreras, V. Lynch, and D. Newman, "An initial model for complex dynamics in electric power system blackouts," in *hicss*. IEEE, 2001, p. 2017.

[3] S. Mei, F. He, X. Zhang, S. Wu, and G. Wang, "An improved opa model and blackout risk assessment," *IEEE Trans. Power Syst.*, vol. 24, no. 2, pp. 814–823, 2009.

[4] I. Dobson, B. A. Carreras, and D. E. Newman, "Branching process models for the exponentially increasing portions of cascading failure blackouts," in *Proc. the 38th Annual Hawaii International Conference on System Sciences*. IEEE, 2005, pp. 64a–64a.

[5] J. Chen, J. S. Thorp, and I. Dobson, "Cascading dynamics and mitigation assessment in power system disturbances via a hidden failure model," *International Journal of Electrical Power & Energy Systems*, vol. 27, no. 4, pp. 318–326, 2005.

[6] J. Song, E. Cotilla-Sanchez, G. Ghanavati, and P. D. Hines, "Dynamic modeling of cascading failure in power systems," *IEEE Trans. Power Syst.*, vol. 31, no. 3, pp. 2085–2095, 2016.

[7] D. Novosel, G. Bartok, G. Henneberg, P. Mysore, D. Tziouvaras, and S. Ward, "IEEE PSRC report on performance of relaying during wide-area stressed conditions," *IEEE Trans. Power Del.*, vol. 25, no. 1, pp. 3–16, 2010.

[8] H. H. Alhelou, M. E. Hamedani-Golshan, T. C. Njenda, P. Siano *et al.*, "A survey on power system blackout and cascading events: Research motivations and challenges," *Energies*, vol. 12, no. 4, pp. 1–28, 2019.

[9] M. K. Neyestanaki and A. Ranjbar, "An adaptive PMU-based wide area backup protection scheme for power transmission lines," *IEEE Trans. Smart Grid*, vol. 6, no. 3, pp. 1550–1559, 2015.

[10] Z. Li, X. Yin, Z. Zhang, and Z. He, "Wide-area protection fault identification algorithm based on multi-information fusion," *IEEE Trans. Power Del.*, vol. 28, no. 3, pp. 1348–1355, 2013.

[11] R. Dubey, S. R. Samantaray, and B. K. Panigrahi, "An spatiotemporal information system based wide-area protection fault identification scheme," *International Journal of Electrical Power & Energy Systems*, vol. 89, pp. 136–145, 2017.

[12] P. Bhui and N. Senroy, "Online identification of tripped line for transient stability assessment," *IEEE Trans. Power Syst.*, vol. 31, no. 3, pp. 2214–2224, 2016.

[13] M. Garcia, T. Catanach, S. Vander Wiel, R. Bent, and E. Lawrence, "Line outage localization using phasor measurement data in transient state," *IEEE Trans. Power Syst.*, vol. 31, no. 4, pp. 3019–3027, 2016.

[14] M. Rafferty, X. Liu, D. M. Laverty, and S. McLoone, "Real-time multiple event detection and classification using moving window PCA," *IEEE Trans. Smart Grid*, vol. 7, no. 5, pp. 2537–2548, 2016.

[15] Y. Song, W. Wang, Z. Zhang, H. Qi, and Y. Liu, "Multiple event detection and recognition for large-scale power systems through cluster-based sparse coding," *IEEE Trans. Power Syst.*, 2017.

[16] Y. Wang, M. Liu, Z. Bao, and S. Zhang, "Stacked sparse autoencoder with PCA and SVM for data-based line trip fault diagnosis in power systems," *Neural Computing and Applications*, 2018.

[17] P. G. Axelberg, I. Y.-H. Gu, and M. H. Bollen, "Support vector machine for classification of voltage disturbances," *IEEE Trans. Power Del.*, vol. 22, no. 3, pp. 1297–1303, 2007.

[18] A. K. Ghosh and D. L. Lubkeman, "The classification of power system disturbance waveforms using a neural network approach," *IEEE Trans. Power Del.*, vol. 10, no. 1, pp. 109–115, 1995.

[19] M. Mishra and P. K. Rout, "Detection and classification of micro-grid faults based on HHT and machine learning techniques," *IET Generation, Transmission & Distribution*, vol. 12, pp. 388–397, 2018.

[20] G. Cardoso, J. G. Rolim, and H. H. Zurn, "Application of neural-network modules to electric power system fault section estimation," *IEEE Trans. Power Del.*, vol. 19, no. 3, pp. 1034–1041, 2004.

[21] K.-H. Kim and J.-K. Park, "Application of hierarchical neural networks to fault diagnosis of power systems," *International Journal of Electrical Power & Energy Systems*, vol. 15, no. 2, pp. 65 – 70, 1993.

[22] T. Dalstein and B. Kulicke, "Neural network approach to fault classification for high speed protective relaying," *IEEE Trans. Power Del.*, vol. 10, no. 2, pp. 1002–1011, 1995.

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[24] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proc. IEEE conference on Computer Vision and Pattern Recognition*, 2014, pp. 1725–1732.

[25] F. Li and Y. Du, "From AlphaGo to power system AI: What engineers can learn from solving the most complex board game," *IEEE Power and Energy Magazine*, vol. 16, no. 2, pp. 76–84, 2018.

[26] Y. Chen, H. Jiang, C. Li, X. Jia, and P. Ghamisi, "Deep feature extraction and classification of hyperspectral images based on convolutional neural networks," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 10, pp. 6232–6251, 2016.

[27] W. Li, M. Wang, and J. H. Chow, "Fast event identification through subspace characterization of PMU data in power systems," in *Proc. IEEE Power and Energy Society (PES) General Meeting*, 2017.

[28] W. Li, M. Wang, and J. H. Chow, "Real-time event identification through low-dimensional subspace characterization of high-dimensional synchrophasor data," *IEEE Trans. Power Syst.*, vol. 33, no. 5, pp. 4937–4947, 2018.

[29] *PSSE Aplication Program Interface (API)*, 34th ed., Siemens Industry, Inc., Schenectady, NY 12301-1058 USA, 2018.

[30] G. Rogers, *Power system oscillations*. Springer Science & Business Media, 2012.

[31] J. H. Tu, C. W. Rowley, D. M. Luchtenburg, S. L. Brunton, and J. N. Kutz, "On dynamic mode decomposition: theory and applications," *Journal of Computational Dynamics*, vol. 1, no. 2, 2014.

[32] M. Hernandez and A. R. Messina, "Nonlinear power system analysis using koopman mode decomposition and perturbation theory," *IEEE Trans. Power Syst.*, 2018.

[33] Y. Susuki and I. Mezić, "Nonlinear koopman modes and power system stability assessment without models," *IEEE Trans. Power Syst.*, vol. 29, no. 2, pp. 899–907, 2014.

[34] D. Trudnowski, J. Johnson, and J. Hauer, "Making prony analysis more accurate using multiple signals," *IEEE Trans. Power Syst.*, vol. 14, no. 1, pp. 226–231, 1999.

[35] Y. Hua and T. K. Sarkar, "Matrix pencil method and its performance," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 1988, pp. 2476–2479.

[36] P. Gao, M. Wang, S. G. Ghiocel, J. H. Chow, B. Fardanesh, and G. Stefopoulos, "Missing data recovery by exploiting low-dimensionality in power system synchrophasor measurements," *IEEE Trans. Power Syst.*, vol. 31, no. 2, pp. 1006–1013, 2016.

[37] J. B. Elsner, "Analysis of time series structure: SSA and related techniques," 2002.

[38] Y. Hao, M. Wang, J. H. Chow, E. Farantatos, and M. Patel, "Modelless data quality improvement of streaming synchrophasor measurements by exploiting the low-rank hankel structure," *IEEE Trans. Power Syst.*, vol. 33, no. 6, pp. 6966–6977, 2018.

[39] M. R. Jovanović, P. J. Schmid, and J. W. Nichols, "Sparsity-promoting dynamic mode decomposition," *Physics of Fluids*, vol. 26, no. 2, p. 024103, 2014.

[40] J. N. Kutz, X. Fu, and S. L. Brunton, "Multiresolution dynamic mode decomposition," *SIAM Journal on Applied Dynamical Systems*, vol. 15, no. 2, pp. 713–735, 2016.

[41] N. Cristianini and J. Shawe-Taylor, *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[42] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[43] D. P. Kingma and J. L. Ba, "Adam: Amethod for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Representations*, 2014.

[44] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," COURSERA: Neural Netw. Mach. Learn, 2012.

[45] Y. Bengio, "Practical recommendations for gradient-based training of deep architectures," in *Neural networks: Tricks of the trade*. Springer, 2012, pp. 437–478.

[46] M. Wang, P. Gao, S. Ghiocel, J. H. Chow, B. Fardanesh, G. Stefopoulos, and M. P. Razanousky, "Identification of "unobservable" cyber data attacks on power grids," 2014. [Online]. Available: http://ecse.rpi.edu/ wang/pub/SmartGridComm14.pdf

[47] S. Pan, T. Morris, and U. Adhikari, "Classification of disturbances and cyber-attacks in power systems using heterogeneous time-synchronized data," *IEEE Trans. Ind. Informat.*, vol. 11, no. 3, pp. 650–662, 2015.

[48] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[49] J. P. Egan, *Signal Detection Theory and ROC Analysis Academic Press Series in Cognition and Perception*. London, UK: Academic Press, 1975.

[50] K. A. Spackman, "Signal detection theory: Valuable tools for evaluating inductive learning," in *Proc. the sixth international workshop on Machine learning*. Elsevier, 1989, pp. 160–163.

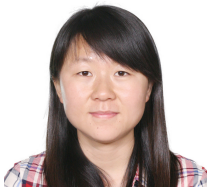
[51] D. J. Hand and R. J. Till, "A simple generalisation of the area under the roc curve for multiple class classification problems," *Machine learning*, vol. 45, no. 2, pp. 171–186, 2001.

[52] T. Fawcett, "An introduction to roc analysis," *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.

[53] P. Kundur, N. J. Balu, and M. G. Lauby, *Power system stability and control*. McGraw-hill New York, 1994, vol. 7.

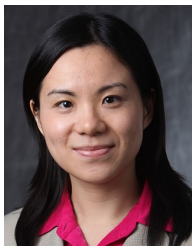
[54] S. Pöyhönen, A. Arkkio, P. Jover, and H. Hyötyniemi, "Coupling pairwise support vector machines for fault classification," *Control Engineering Practice*, vol. 13, no. 6, pp. 759–769, 2005.

[55] M. Brown, M. Biswal, S. Brahma, S. J. Ranade, and H. Cao, "Characterizing and quantifying noise in PMU data," in *Proc. IEEE Power and Energy Society (PES) General Meeting , 2016*. IEEE, 2016, pp. 1–5.



Wenting Li (S'16) received the B.E. degree from Harbin Institute of Technology, Heilongjiang, China, in 2013.

She is pursuing the Ph.D. degree in electrical engineering at Rensselaer Polytechnic Institute, Troy, NY. Her research interests include high-dimensional data analytics, feature extraction, application of machine/deep learning methods to identify and locate events in power grids.



Meng Wang (M'12) received B.S. and M.S. degrees from Tsinghua University, China, in 2005 and 2007, respectively. She received the Ph.D. degree from Cornell University, Ithaca, NY, USA, in 2012.

She is an Assistant Professor in the department of Electrical, Computer, and Systems Engineering at Rensselaer Polytechnic Institute, Troy, NY, USA. Her research interests include high-dimensional data analytics, machine learning, power systems monitoring, and synchrophasor technologies.