An Influence Model for Failure Cascade Prediction and Analysis in Power Systems

Abstract—**Power systems are shown under the risk of widespread failure cascades which are challenging to model and predict. The ability to predict the failure cascade is important for contingency analysis and corrective control designs to prevent large blackouts. In this paper, we study an influence model framework to predict failure cascades and try to figure out their underlying pattern in real power systems. A hybrid learning scheme is proposed to train the influence model from simulated failure cascade sample pools. The learning scheme firstly applies a Monte Carlo approach to quickly acquire the pairwise influences in the influence model. Then, a convex quadratic programming formulation is implemented to obtain the weight of each pairwise influence. Finally, an adaptive selection of threshold for each link is proposed to tailor the influence model to better fit different initial contingencies. We test our framework on a number of large scale power networks under both DC and AC flow models, and verify its prediction performance through numerical simulations in both accuracy and efficiency: Under limited training samples, the proposed framework is capable of predicting the final state of links within 10% error rate, and the failure cascade size within 7% error rate in most cases, along with around two magnitude of time cost reduction in large systems. Besides, we also show that the trained influence model can unveil instructive insights on cascade properties such as influence sparsity, non-local propagation, critical initial outages and components.**

I. Introduction

Modern power systems frequently experience unexpected component failures which are caused by falling tree branches, storms, lightening strikes, aged devices, wrong protective actions, etc. These random failures, if not treated properly, can propagate to other system components and eventually incur enormous blackouts. In Northeast America on Aug. 14, 2003, a 345kV line tripped off after touching a tree limb, which, within 1 hour, led to expansion of failures from several links to over 500 generating units in US and Canada [1]. In 2019, the Midtown Manhattan in New York sank into sudden darkness on Jul. 13 most likely owing to a disabled transformer [2], and within a month, on Aug. 9, London underwent a similar experience due to the loss of two generators [3].

Past events have shown that large scale blackouts are usually accompanied by a rapid propagation of failures among many system components. This "rolling snowball" phenomenon is referred to as a *failure cascade* in the power system literature. Failure cascades happened in the history share some common features. Firstly, they are subject to environments. Several components may be tripped off simultaneously by winds or lightening, which contains combinatorial many possible cases [1]–[3]. Secondly, they are subject to man-made decisions, including load shedding and power flow re-dispatch. Such decisions may vary among different systems and operators [4]. Thirdly, the dynamic interaction among hundreds of thousands of components in large scale systems may be highly nonlinear and varying with time [5]. Finally, they do not necessarily propagate locally [6], [7]. Spatially remote components may also shut down, invalidating protection simply over components topologically adjacent to initial contingency. All these features pose great difficulty in modeling and predicting failure cascades.

Many efforts have been devoted to constructing reasonable models to capture failure cascades. A commonly used failure cascade model attempts to solve the static power flow problem step-by-step to determine the sequence of (quasi) static transmission link overflows that occur before the system finally satisfies all the engineering constraints [6]. In this failure cascade model, solving the AC power flow problem suffers from a heavy computational burden. It is also difficult to find an AC power flow solution when the network configuration changes drastically [5]. Thus, most works applied the simplified DC power flow model in the failure cascade analysis. Although the DC power flow is a linearized version of the AC model, it still bears some computational efforts when solving it repeatedly in the failure cascade screening process. A few works have been done to derive a closed form solution for the link flows when some links are tripped off [8], [9]. This approach has to take the pseudo-inverse of the system admittance matrix, and has to reformulate the solution when islanding occurs.

In view of the above difficulties on applying power flow models to analyze cascade process, a number of existing works attempt to ignore the power flow constraints and concentrate on the contagious cascade process in which new failures only happen over adjacent components. It is referred to as the "contagion model" which originates from the percolation model [10]–[12]. However, these abstract models overlook the nonadjacent correlations among geographically distant failures, and the complete decoupling with power system dynamics renders these models to be rather inaccurate.

To capture the nonadjacent failure propagation, Savathiratham *et. al.* proposed the Influence Model (IM) [13]. It is a special graphic model that considers the influence of all the network components on each individual component. It first establishes the pairwise correlation between any two network components, and then summarizes all the pairwise correlations associated with each single component. This aggregated correlation value of each component, which is called the *influence*, is used to determine the state of this component in the failure cascade prediction. The influence model is essentially a Markovian model which is easy to construct and implement for large scale applications. Thus, it is a powerful tool for analyzing and predicting failure cascades in power networks [14]–[16].

In order to construct the influence model that can be used for power system failure cascade predictions, Hines et.al.

applied the Monte Carlo method to learn the influences from historical cascade records [17]. They focused on generating the appropriate distribution of failure cascade sizes that best match existing records, and applying the model to identify critical components. Zhou et.al. further extended the pairwise influences to capture more complex influences for failure size prediction [18].

In this paper, we propose a hybrid learning framework that can be used to train the influence model from either historical data or synthetic data. Then, we apply our trained influence model to predict the failure cascade sequences for a few large scale power system test cases. Numerical simulations show that our model results in relatively accurate prediction under metrics characterizing different facets, with significant speed advantage compared to the flow computation of the power dynamics. The major contributions are summarized below.

- We proposed a hybrid learning framework that can efficiently train the influence model for very large systems. The proposed learning framework integrates Monte Carlo method with quadratic programming, and an adaptive threshold selection scheme to quickly train the model for making valid predictions.
- We applied the influence model to a few large scale power system test cases to predict their failure cascade sequences under both DC and AC cases. The prediction performances are thoroughly evaluated at different levels of granularity: failure size, final state, and failure time prediction. Moreover, we not only evaluate the prediction performance of the whole system for any new cascade sequence, but also assess the prediction over the state of any specific transmission link we are interested in the system. It is the first time that such comprehensive evaluation is investigated on cascade failure prediction in large systems as far as we are concerned.
- We demonstrated that the influence model trained by the hybrid learning framework can unveil valuable insights about cascade properties in the system, including influence sparsity, relationship between influence and physical distance, and critical initial contingencies. Such discoveries may pave the way for more efficient management and corrective design on power systems.

Besides, the influence model trained by the hybrid learning framework promises to be meaningful in industrial applications. Firstly, the learning process can be implemented in parallel, suitable for large cascade systems. Secondly, it needs very small portion ($< 5\%$ generally) of cascading sequence samples for training the model among the huge pool of cascade sequences induced by all the possible initial contingencies. Thirdly, it predicts around two orders of magnitude faster than simulation based on power flow calculation, and such gap is even sharper in power systems with large size.

The rest of the paper is organized as follows: Section II summarizes related works about failure cascade modeling, analysis, prediction, and control in power systems. Section III introduces the basics of the influence model and its parameters we need to learn. Section IV presents the procedure we generate cascade sequences under DC/AC power flow for training the influence model. Section V introduces our hybrid learning framework in detail. Section VI presents the performance evaluation on several power systems under different metrics. Section VII reveals the insights on the property of cascading failures that the trained influence model can reflect.

II. Related Works

Failure cascade in power systems has been under extensive concern in the past decade. Most of the works about it are motivated by historical large blackouts. Vaiman et.al. listed possible sources of initially tripped links, and further summarized the large blackouts happened every year from 1984 to 2006 incurred by different natural disasters [19]. Hines et.al. further discovered from cascade records that the blackout size follows the Weibull distribution when the size is small, and the power-law distribution when the size is large [17]. These works, originated from the historical records, reveal basic properties of cascades and pave the way for quantitative modeling and analysis.

However, the destitution of enough historical cascade records inhibits researchers to gain intrinsic factors that determine failure cascade. To tackle with it, a bunch of works utilized power flow models to generate simulated failure cascade sequences for analysis. Zussman et.al. showed that AC flow model is suitable in characterizing real failure cascade, while DC flow model, much less complex, may underestimate the ultimate failure size [5], [20]. For simplicity, most works are based on DC power model. Zussman et.al. analytically revealed how failure propagates with multiple tripped links based on solving flow equations [8], [9], while Beinstock and Varma gained insights in a mixed-integer optimization view [21]. Zussman et.al. also formulated a general failure cascade model predicated on DC flow calculation [6].

Another track for cascade analysis is to utilize flow-free models, getting away from flow calculations. Most of such models are probabilistic. Firstly, percolation-based analysis on random graph model is of great popularity, primarily in the perspective of local cascade whose failure probability depends on neighboring link degree [10], [11], [22]. Moreover, the branching process is a popular tool that measures the distribution of the number of outages in a cascade [23], [24], and the random chemistry algorithm together with such distribution can efficiently estimate the overall blackout risk [25], [26]. Besides, Hines et.al. applied differential equations to model the cascade process [27] while Zhang et.al. did so by mean field theory [28]. Moreover, Tse et.al. tried chemical master equation model to grasp the collective behavior of failure propagation [29] while Qi et.al. learned such behavior via EM algorithm [30].

The influence model also takes its role in the study of failure cascade process, especially analyzing cascade dynamics and predicting the ultimate blackout size. Naeini et.al. conducted a series of works about modeling the transition probability between two network states [14]–[16]. Dobson et.al. further proposed an efficient Monte-Carlo based method to acquire reasonable estimation for pairwise influence in the system and demonstrated its capability to identify critical components [17], [18].

III. Influence Model

The IM is a Markovian model whose dynamics are described by the state variable transitions. In power system failure cascade analysis, the state variable of the IM is typically chosen to be the binary operational state of each transmission link[1], which takes on values of either 0 (failed) or 1 (normal) respectively [23]. Given the i-th link, we use $s_i[t]$ to denote its state at time step t. The collection $s[t] := [s_1[t], \dots, s_M[t]]' \in \{0,1\}^{M\times 1}$ of all the M link state variables represents the *network state* at time t, and we define $s := \{s[t]\}_{t=0}^{T}$ as the state (failure cascade) sequence where T is the termination time of the cascade.

According to the IM, the transition of a state variable $s_i[t]$ from the current time t to the next time $t+1$ is described by

$$\tilde{s}_i[t+1] = \sum_{j=1}^{M} d_{ij}\big(\mathbf{A}_{ji}^{11} s_j[t] + \mathbf{A}_{ji}^{01}(1 - s_j[t])\big), \qquad (1)$$

where

$$\mathbf{A}_{ji}^{11} := P\big(s_i[t+1] = 1 \mid s_j[t] = 1\big) \qquad (2)$$

$$\mathbf{A}_{ji}^{01} := P\big(s_i[t+1] = 1 \mid s_j[t] = 0\big) \qquad (3)$$

are the transition probabilities from the state of link j at t to the state of link i at $t+1$; $\tilde{s}_i[t+1]$ is the estimated value of $s_i[t+1]$[2]. We collect all the $\mathbf{A}_{ji}^{11}$'s into an $M \times M$ matrix $\mathbf{A}^{11}$, and similarly $\mathbf{A}_{ji}^{01}$'s to obtain $\mathbf{A}^{01}$. We refer to each of $\mathbf{A}^{11}$ and $\mathbf{A}^{01}$ as the "pairwise influence matrix".

The weight d_{ij} in (1) represents the proportional affect from the link pair-(i,j). If we further require

$$d_{ij} \geq 0, \qquad (4a)$$

$$\textstyle\sum_{j=1}^{M} d_{ij} = 1, \qquad (4b)$$

then with given link i, d_{ij} can be regarded as a probability mass function for the pairwise influence of any link j on i. We collect all the d_{ij}'s into an $M \times M$ matrix $\mathbf{D}$, referred to as the "weighted influence matrix".

When $\tilde{s}_i[t+1]$ is obtained, the IM assigns 1 to $s_i[t+1]$ with a probability $\tilde{s}_i[t+1]$, and 0 to $s_i[t+1]$ with a probability $1 - \tilde{s}_i[t+1]$. This is a randomized mapping from the unit interval $[0,1]$ to the binary set $\{0,1\}$

$$\hat{s}_i[t+1] = \begin{cases} 1 & \text{w.p.} \quad \tilde{s}_i[t+1] \\ 0 & \text{w.p.} \quad 1 - \tilde{s}_i[t+1] \end{cases}, \qquad (5)$$

where $\hat{s}_i[t+1]$ is the predicted binary value of $s_i[t+1]$ after the randomized mapping.

This process can yield different prediction results from the same initial contingencies. To make predictions consistent, we reduce the randomized mapping to a deterministic bisection scheme with a threshold ϵ_i for each link i:

$$\hat{s}_i[t+1] = \begin{cases} 1 & \text{if } \tilde{s}_i[t+1] \geq \epsilon_i \\ 0 & \text{if } \tilde{s}_i[t+1] < \epsilon_i \end{cases}, \qquad (6)$$

We collect $\hat{s}_i[t]$ in $\hat{s}[t]$ for all the links and refer to it as the prediction of the network state at time t. Then, the time sequence of $\hat{s}[t]$, denoted as $\hat{s}$, is a predicted failure cascade sequence.

An illustrative example of the IM can be found in Fig. 1.

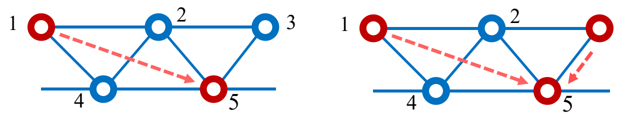


Fig. 1. We consider a 5-component network and take the influence on component 5 as an example. The left subfigure depicts the pairwise influence of component 1 on 5, concerning $\mathbf{A}_{15}^{11}$ and $\mathbf{A}_{15}^{01}$ in (1). The right subfigure reflects that component 5 is mutually influenced by component 1 and 3, where $\mathbf{A}_{15}^{11}$ and $\mathbf{A}_{15}^{01}$ take weight d_{51}, while $\mathbf{A}_{35}^{11}$ and $\mathbf{A}_{35}^{01}$ take weight d_{53} in (1).

To apply the IM in failure cascade analysis, the pairwise influence matrices $\{\mathbf{A}^{11}, \mathbf{A}^{01}\}$, the weighted influence matrix $\mathbf{D}$, and the bisection thresholds ϵ_i's need to be specified, which serves as the main theme for the following two sections.

IV. Generating Sample Cascade Sequences

The IM parameters $\mathbf{A}^{11}$, $\mathbf{A}^{01}$, $\mathbf{D}$ and ϵ_i's are learned from known failure cascade data. However, the historical failure data of a real power grid is usually inaccessible because the power grid is a critical infrastructure whose data should be kept confidential, and because in a particular power grid large scale failure cascades rarely happen. To get enough data for training the IM, we generate synthetic failure cascades by the power flow based simulation approach [17]. Specifically in this paper, we solve the DC/AC power flow problem using the MATPOWER Toolbox [31]. The procedures of generating synthetic failure cascades are summarized as follows.

1) Given the loading condition, compute the initial link flows.
2) Randomly initiate an $M-k$ contingency (k initially failed links) where $k = 2$ or 3.
3) Detect all the islands (disconnected sub-graphs) that appear. If the whole network is connected, then it can be viewed as the only island.
4) Specify a new bus that is connected to the ground in each island.
   - In our setting we specify the bus with lowest loading to be the one connected to the ground.
5) Re-balance the power in each island by either generation curtailment or load shedding depending on whether the supply exceeds the demand.
6) Recompute the link flows in each island.
7) Detect the newly overflowed links, remove the overflowed links and return to Step 3). Otherwise, terminate.

Repeating this procedure we can build up the training sample pool $\mathcal{S}_{train}$. We denote the k-th cascade sequence in $\mathcal{S}_{train}$ as

$$s^k := \{s^k[t]\}_{t=0}^{T_k}, \quad k = 1, 2, \dots, K,$$

where superscript k is the sample index; K is the total number of training samples; T_k denotes the final time step that the k-th cascade terminates; $s^k[t]$ is the network state at time t; and

[1] Other component can also be considered, for example, the generators and transformers.

[2] $\tilde{s}_i[t+1]$ can be interpreted as the expectation of $s_i[t+1]$, provided equation (4).

s^k records the k-th cascade sequence. We will use this sample pool to train our IM parameters.

V. Learning Influence Model Parameters

In this section, we explore how to learn the best values of $\mathbf{A}^{11}$, $\mathbf{A}^{01}$, $\mathbf{D}$ and ϵ_i's from $\mathcal{S}_{train}$. Recall (1) that $\tilde{s}_i[t+1]$ estimates $s_i[t+1]$ from the IM parameters $\mathbf{A}^{11}$, $\mathbf{A}^{01}$, $\mathbf{D}$ and the network state $s[t]$ at time t. Hence, the objective is to identify the values of $\mathbf{A}^{11}$, $\mathbf{A}^{01}$, $\mathbf{D}$ such that the estimation $\tilde{s}_i[t+1]$ can best fit the existing sample $s_i[t+1]$ for every link at all the time steps. We achieve this goal by formulating a constrained optimization problem as follows.

$$\begin{aligned}\min_{\mathbf{D},\mathbf{A}^{11},\mathbf{A}^{01}} \quad & \frac{1}{K}\sum_{k=1}^{K}\sum_{t=1}^{T_k} f(s^k[t], \tilde{s}^k[t]) \\ s.t. \quad & \tilde{s}_i^k[t+1] = \sum_{j=1}^{M} d_{ij}\left(\mathbf{A}_{ji}^{11} s_j^k[t] + \mathbf{A}_{ji}^{01}(1 - s_j^k[t])\right), \forall i, k; \\ & \sum_{j=1}^{M} d_{ij} = 1, \forall i; \quad d_{ij}, \mathbf{A}_{ji}^{11}, \mathbf{A}_{ji}^{01} \geq 0, \forall i, j,\end{aligned} \tag{7}$$

where $f(s^k[t], \tilde{s}^k[t])$ is the cost function that quantifies the distance between $s^k[t]$ and $\tilde{s}^k[t]$; M is the number of links; K is the size of $\mathcal{S}_{train}$; and T_k is the termination time step in the k-th cascade sample.

The problem size of (7) is very large because for each link pair (i, j) there exist two independent pairwise influence values $\mathbf{A}_{ji}^{11}$ and $\mathbf{A}_{ji}^{01}$. For example, in a system with $1,000$ links, we have altogether 3×10^6 variables concerning $\{\mathbf{A}^{11}, \mathbf{A}^{01}, \mathbf{D}\}$. In order to improve the computational efficiency, we train the pairwise influence $\{\mathbf{A}^{11}, \mathbf{A}^{01}\}$ and relative influence $\mathbf{D}$ separately.

A. Learning Pairwise Influence Matrices $\mathbf{A}^{11}$ and $\mathbf{A}^{01}$

We apply the Monte Carlo method to learn the pairwise influence matrices $\mathbf{A}^{11}$ and $\mathbf{A}^{01}$ from the sample pool $\mathcal{S}_{train}$ with size K.

Let τ_i^k be the time step that link i changes to failure state in the k-th cascade sequence s^k. If link i does not fail in s^k, we set τ_i^k to be the termination time of s^k. Then, the value of $\mathbf{A}_{ji}^{11}$ for any link i and j is computed by

$$\mathbf{A}_{ji}^{11} := \frac{\sum_{k=1}^{K} C_{ji}^{11}(s^k, \tau_i^k)}{\sum_{k=1}^{K} C_j^1(s^k, \tau_i^k)} \tag{8}$$

where $C_j^1(s^k, \tau_i^k)$ is the number of time steps before τ_i^k in s^k such that link j is normal; $C_{ji}^{11}(s^k, \tau_i^k)$ is the number of time steps before τ_i^k in s^k such that link i is normal, given link j is normal on the adjacent upstream time step.

Similarly, we can estimate $\mathbf{A}_{ji}^{01}$ via

$$\mathbf{A}_{ji}^{01} := \frac{\sum_{k=1}^{K} C_{ji}^{01}(s^k, \tau_i^k)}{\sum_{k=1}^{K} C_j^0(s^k, \tau_i^k)} \tag{9}$$

where $C_j^0(s^k, \tau_i^k)$ is the number of time steps before τ_i^k in s^k such that link j is failed; $C_{ji}^{01}(s^k, \tau_i^k)$ is the number of time steps before τ_i^k in s^k such that link i is normal, given link j is failed on the adjacent upstream time step. In this counting process, we should exclude the samples where link i fails initially because it results from external factors independent of flow dynamics.

We take the following toy example to gain a more intuitive view of (8) and (9). We consider two cascade sequences, s^1 and s^2, over 2 links, where

$$s^1 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad s^2 = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix}.$$

We can observe that $\tau_1^1 = 6$ and $\tau_1^2 = 4$. For $\mathbf{A}_{21}^{11}$, we have $C_2^1(s^1, \tau_1^1) = 2$ and $C_{21}^{11}(s^1, \tau_1^1) = 2$ in s^1, and meanwhile $C_2^1(s^2, \tau_1^2) = 3$ and $C_{21}^{11}(s^2, \tau_1^2) = 2$ in s^2. According to (8) and (9),

$$\mathbf{A}_{21}^{11} = \frac{C_{21}^{11}(s^1, \tau_1^1) + C_{21}^{11}(s^2, \tau_1^2)}{C_2^1(s^1, \tau_1^1) + C_2^1(s^2, \tau_1^2)} = \frac{2+2}{3+2} = \frac{4}{5},$$

$$\mathbf{A}_{21}^{01} = \frac{C_{21}^{01}(s^1, \tau_1^1) + C_{21}^{01}(s^2, \tau_1^2)}{C_2^0(s^1, \tau_1^1) + C_2^0(s^2, \tau_1^2)} = \frac{2+0}{3+0} = \frac{2}{3}.$$

B. Learning Weighted Influence Matrix $\mathbf{D}$

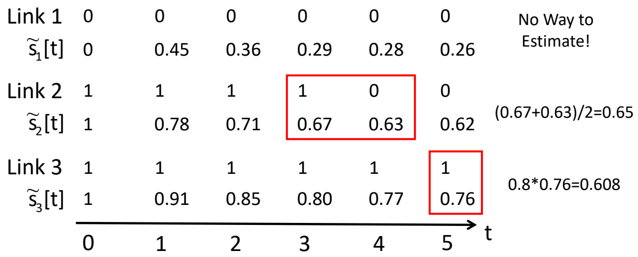
Once $\{\mathbf{A}^{11}, \mathbf{A}^{01}\}$ have been obtained, their values can be substituted into (7) to form a reduced optimization problem whose decision variables are only from $\mathbf{D}$. We choose the objective function $f(\cdot)$ to be the least square error function, which induces a convex quadratic programming problem as follows.

$$\begin{aligned}\min_{\mathbf{D}} \quad & \frac{1}{K}\sum_{k=1}^{K}\sum_{t=1}^{T_k}\sum_{i=1}^{M}\Bigg(s_i^k[t+1] \\ & \qquad - \sum_{j=1}^{M} d_{ij}(\mathbf{A}_{ji}^{11} s_j^k[t] + \mathbf{A}_{ji}^{01}(1 - s_j^k[t]))\Bigg)^2 \\ s.t. \quad & \sum_{j=1}^{M} d_{ij} = 1, \forall i; \qquad d_{ij} \geq 0, \forall i, j.\end{aligned} \tag{11}$$

The formulation (11) can be solved efficiently by numerical methods such as the Frank-Wolfe algorithm [32]. The optimal solution to this problem serves as the Bayes least-squares estimator when $\tilde{s}_i[t+1] = \mathbf{E}[s_i[t+1] \mid s[t]]$. Moreover, the weighted influences on each link i are independent of the influences on any other link j, which further supports a problem decomposition into M sub-problems. We solve these small optimization problems in parallel in practice to further reduce the training time.

C. Learning Bisection Threshold ϵ_i

To make a deterministic prediction of a failure cascade, the value of ϵ_i in (6) should be provided. A naive way is to set an universal $\epsilon_i = 0.5$ for every link i. However, this undifferentiated threshold value can easily incur wrong predictions. For example in Fig. 2, the third row shows that link 2 fails at the fourth time step. However, the fourth row indicates that the predicted state value at the fourth time step is 0.63 which is greater than 0.5. Thus, by (6) the state of link 2 will be assigned to 1 instead of 0, misidentifying the failure. Therefore, the threshold value ϵ_i should be selected adaptively according to differential initial contingencies. We summarize



Link 1	0	0	0	0	0	0	No Way to Estimate!
$\tilde{s}_1[t]$	0	0.45	0.36	0.29	0.28	0.26	
Link 2	1	1	1	1	0	0	
$\tilde{s}_2[t]$	1	0.78	0.71	0.67	0.63	0.62	(0.67+0.63)/2=0.65
Link 3	1	1	1	1	1	1	
$\tilde{s}_3[t]$	1	0.91	0.85	0.80	0.77	0.76	0.8*0.76=0.608
t	0	1	2	3	4	5	

Fig. 2. This is an example of determining thresholds on all 3 link categories in a cascade sample. Each link representative has two rows of records: the first row denotes the real state value at each time step, while the second row denotes the iterative values of $\tilde{s}_i[t]$ for every link i based on (1). For link 2 we set ϵ_2^k as the average of $\tilde{s}_2[4]$ and $\tilde{s}_2[5]$, while for link 3 we set ϵ_3^k to be $0.8 \times \tilde{s}_3[T_k] = 0.8 \times 0.76 = 0.608$, where 0.8 can be replaced by any real value within $(0, 1)$.

our adaptive threshold selection scheme for a specific link i as follows.

1) **Identifying a threshold value of link i in a sample sequence s^k.** Three situations can happen. 1) Link i fails initially in the sample sequence s^k. In this situation, there is no way to know the threshold value. 2) Link i fails but not from the beginning of the sample sequence s^k. Then, we recursively compute the estimated state variable $\tilde{s}_i^k[t+1]$ by assigning $\tilde{s}_j^k[t]$ to $s_j^k[t]$ on the right hand side of (1). The critical time step where link i fails is determined. We choose the threshold value for this sequence to be the intermediate value of the estimated state at the critical time step and the estimated state at its upstream adjacent time step. 3) Link i never fails in the sample sequence s^k. In this situation, we recursively compute the estimated state variable $\tilde{s}_i^k[t]$, and choose the threshold value to be $\alpha \times \tilde{s}_i^k[T_k]$ at the final time T_k, where $\alpha \in (0, 1)$ can be selected arbitrarily. Fig. 2 shows the identification of ϵ_i among these three situations by a toy example.
2) **Forming the threshold pool of link i from all sample sequences.** For each link i, we compute the threshold value ϵ_i^k for every sample sequence s^k and collect them in a set Ω_i.
3) **Selecting an appropriate threshold value of link i for a new contingency.** The basic idea is to select the threshold value ϵ_i from the known threshold set Ω_i such that the associated known contingency is "closest" to the new contingency denoted as $s^{new}[0]$. The closest known contingency to $s^{new}[0]$ is defined by[3]

$$k^* = \arg\min_{k=1,2,\cdots,K} ||s^{new}[0] - s^k[0]||_1 \tag{12}$$

where k is the index of the known contingency; $s^k[0]$ is the known contingency; $s^{new}[0]$ is the new contingency; and $||s^{new}[0] - s^k[0]||_1$ is the L_1-norm, denoting the number of links that have different initial states in $s^{new}[0]$ and $s^k[0]$. Then, we select the threshold value $\hat{\epsilon}_i$ to be

$$\hat{\epsilon}_i = \epsilon_i^{k^\star}. \tag{13}$$

Sometimes multiple solutions for $k^\star$ exist for (12). We choose $\hat{\epsilon}_i$ to be the median value among multiple options.

$$\hat{\epsilon}_i = \text{median}\{\epsilon_i^k\}_{k \in K^*} \tag{14}$$

where $K^\star$ is the optimal solution set of (12).

D. Overall Procedure

The overall procedure of learning the IM and using it for failure cascade predictions is presented in Algorithm 1. In the learning modular, finding the weighted influence matrix $\mathbf{D}$ is the most computationally expensive part because it requires solving a convex quadratic programming problem. After that, the remaining computational efforts are much less demanding. The prediction modular in Algorithm 1 is computationally inexpensive since it only requires multiplication and addition manipulations. Therefore, using the IM to predict failure cascades is potentially much faster than using flow equations to make predictions. This claim will be numerically verified later in the performance evaluations in Section V.

Algorithm 1: Learning Approach and Failure Cascade Prediction based on Influence Model

```
Input: Training Sample Pool $\mathcal{S}_{train} = \{s^k[t]\}_{t=0,\ldots,T_k}^{k=1,\ldots,K}$;
       New Initial State $s^{new}[0]$.
Output: Weighted Influence Matrix $\mathbf{D}$; Pairwise
        Influence Matrices $\{\mathbf{A}^{11}, \mathbf{A}^{01}\}$; Sequence
        Prediction $\hat{s}^{new}$.
// Learning Influence Model Parameters
Estimate $\mathbf{A}^{11}$ and $\mathbf{A}^{01}$ based on Monte-Carlo Method;
Learn $\mathbf{D}$ from the quadratic optimization (11);
Build the threshold set $\Omega_i$ for each link $i$;
Find $k^*$ based on equation (12) and form the set $K^*$
  containing all $k^*$s;
Obtain $\hat{\epsilon}_i$ by equation (14) for each link $i$;
// Failure Cascade Prediction
$t \leftarrow 0$;
while there are new links predicted failed at time $t$ do
    $\tilde{s}_i^{new}[t+1] \leftarrow \sum_{j=1}^{M} d_{ij}(\mathbf{A}_{ji}^{11} s_j^{new}[t] + \mathbf{A}_{ji}^{01}(1 - s_j^{new}[t]))$
    for each link $i$;
    $\hat{s}_i^{new}[t+1] \leftarrow 0$ if $\tilde{s}_i^{new}[t+1] < \hat{\epsilon}_i$, $\hat{s}_i^{new}[t+1] \leftarrow 1$
    otherwise;
    $s^{new}[t+1] \leftarrow \hat{s}^{new}[t+1]$, $t \leftarrow t+1$;
$T \leftarrow t - 1$;
return $\mathbf{D}$, $\mathbf{A}^{11}$, $\mathbf{A}^{01}$, $\hat{s}^{new}$.
```

VI. Performance Evaluation

In this section, we conduct a comprehensive numerical study on the proposed method for failure cascade prediction. We first introduce the power systems we test on and the performance metrics, and then present the performance evaluation results with detailed explanations.

[3] Other ways to select the closest contingency is also possible.

A. Dataset Information

We apply the standard datasets in Matlab MATPOWER toolbox [31]. We choose four systems: 118-bus, 1354-bus, 2383-bus, 3012-bus. The 118-bus system represents small systems. In this case, link capacity information is not provided. We set the capacity of each link to be a certain times of its flow value based on the default power injection and loading provided in the dataset, which is termed as 'Factor of Safety' (FoS). We vary the FoS values to set the capacity values, where lower FoS implies lower capacity values and thus lower robustness. The other 3 cases are large scale systems with only few link capacity values not given[4], hence we generally use the provided capacity values. For links lacking of capacity information, we set capacity values to be large enough such that they are free from failure during the cascade process.

Since in the 118-bus system, we set the link capacity values proportional to the link flow calculated by the loading and power generation, so the robustness is not ensured as that in large systems where capacity values are mostly provided officially. Therefore in 118-bus system, the cascading failures may even be broader and thus harder to predict.

We consider both DC and AC power flow models. The generation of cascade sequences has been shown in Section IV. Here we supplement more details:

- For DC case, the power flow on a link (i,j) is exactly the active power, denoted as P_{ij}, and thus the flow $f_{ij} = P_{ij}$. For AC case, the power flow on a link (i,j) contains both active power P_{ij} and reactive power Q_{ij}, and from the dataset we find $P_{ij} \neq P_{ji}$ and $Q_{ij} \neq Q_{ji}$, hence we refer to [5] to calculates the power flow f_{ij} as
$$f_{ij} = \frac{1}{2}(\sqrt{P_{ij}^2 + Q_{ij}^2} + \sqrt{P_{ji}^2 + Q_{ji}^2}).$$
- We use Newton-Raphson method [33] to solve nonlinear equations in AC power model in each island during cascade. However sometimes it does not converge. In this case, we should do load shedding. Specifically, we proportionally decrease the power injection and loading until convergence [5]. In our setting, we decrease 20% power and loading in every trial of calculation before convergence, and shed all power and loading to be 0 in the corresponding island after 10 unconverged trials to avoid unnecessary time waste.

Under both DC and AC models, we observe that in each network system, there exist some transmission links that have never experienced failure induced by the cascade process initiated by the outages of other links. Such links do not provide any information about their failures, hence we exclude them and term the rest as *effective links* which we consider in performance evaluation.

Note that there are many possible settings in each system, including power injection, loading, and capacity values. Therefore for convenience, we define a *default setting* for each system. We set FoS$=1.2$ as the default setting for 118-bus system, and set the loading and capacities provided in MATPOWER toolbox for the other 3 systems as their corresponding default settings. Table I presents the information for cascade samples in 4 systems under DC model and their default settings, where *#Eff. Rate* is the portion of effective links in the system, and *Fail Size* is the total number of link outages in a cascade sequence sample. The data is calculated from $50,000$ cascade samples induced by different initial contingencies. We can find that the failure size varies in a wide range, thus nontrivial to predict the cascade sequences induced by different initial contingencies.

TABLE I
DEFAULT SAMPLE INFORMATION UNDER DC

System	118-Bus	1354-Bus	2383-Bus	3012-Bus
#Generators	54	260	327	297
#Links	179	1710	2886	3566
#Eff. Links	171	762	2088	2083
Eff. Rate	95.5%	44.6%	72.4%	58.4%
Avg. Fail Size	98	179	598	263
Max Fail Size	140	314	862	792
Min Fail Size	3	2	110	11

In our experiment, we consider at most 50000 cascade sequence samples for training in each system, and will vary the size of training set. We focus on $M-3$ initial contingencies in 118-bus system while $M-2$ initial contingencies in larger systems, where $M-k$ denotes k links fail initially among all M links in the system. Note that the ratio of the training samples compared with the set of all possible $M-k$ initial contingencies is very low, for example in 1354-bus system, the ratio is $50,000/\binom{1710}{2} = 3.4\%$, and such ratio is even lower in other cases.

B. Performance Metrics

To evaluate prediction performance, we consider two sets of metrics.

- Sample-based metrics: This set of metrics capture the prediction performance for each cascade sequence samples in the test set. Acceptable performance on these metrics shows the applicability to well predicting the cascade sequence caused by any set of initial contingencies in general.
- Link-based metrics: This set of metrics capture the prediction performance on each link among all the cascade sequences in the test set. Acceptable performance on these metrics shows the accuracy of the prediction mechanism on each link we target at.

1) Sample-based metrics:

Sample-based metrics include:

- **Avg. Failure Size Error Rate** l_{size}: $l_{size} = \frac{1}{K}\sum_{k=1}^{K} l_{size}^k$, where l_{size}^k is the failure size prediction error, relative to real failure size, in the k-th test sample.
- **Avg. Final State Error Rate** l_f: $l_f = \frac{1}{K}\sum_{k=1}^{K} l_f^k$, where l_f^k is the ratio of links whose final states are mistakenly predicted, in the k-th test sample.
- **Avg. Failure Time Error** l_t: $l_t = \frac{1}{K}\sum_{k=1}^{K} l_t^k$, where l_t^k is the failure time prediction error among all links that fail eventually in the k-th test sample.

Fig. 3 illustrates the way we calculate l_{size}^k, l_f^k, and l_t^k for each test sample s_{test}^k and each link i. For l_{size}^k and l_f^k, we

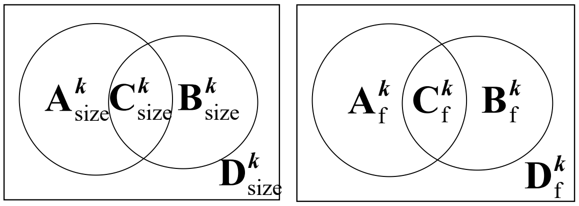
[4] The `rateA` in each system data in MATPOWER toolbox denotes the given capacity values [31].

use Venn graph, divided into four disjoint subsets A, B, C, D, to show the calculation. Inside, $A \cup C$ denotes the set of real failures, $B \cup C$ denotes the set of predicted failures, C is the set of correctly predicted failures, while D represents links not failed and meanwhile not predicted to be failed.

- For l^k_{size}, $|A^k_{size} \cup C^k_{size}|$ is the real failure size while $|B^k_{size} \cup C^k_{size}|$ is the predicted failure size in s^k_{test}, and
$$l^k_{size} = \frac{\big||B^k_{size} \cup C^k_{size}| - |A^k_{size} \cup C^k_{size}|\big|}{|A^k_{size} \cup C^k_{size}|} \times 100\%.$$
- For l^k_f, $|A^k_f \cup B^k_f|$ is the number of effective links whose final states we mistakenly predict in s^k_{test}, while $|A^k_f \cup B^k_f \cup C^k_f \cup D^k_f|$ denotes the number of effective links, and
$$l^k_f = \frac{|A^k_f \cup B^k_f|}{|A^k_f \cup B^k_f \cup C^k_f \cup D^k_f|} \times 100\%.$$

For l^k_t, we take an example on a cascade sequence over 6 links that fail during the cascade. Each number in the first row denotes the time step a link becomes failed, while each number in the second row denotes our prediction on the time step it fails[5].

We ignore counting in initially failed links in all the metrics.



	Link 1	Link 2	Link 3	Link 4	Link 5	Link 6
Real	4	3	5	4	2	1
Pred.	2	5	5	1	5	3

$$l^k_t = \frac{1}{6}\left(|4-2|+|3-5|+|5-5|+|4-1|+|2-5|+|1-3|\right) = 2$$

Fig. 3. An illustrative example of all metrics

These metrics represent three levels of granularity shown in Fig. 4. l_{size} focuses on the number of failures but does not care about the accuracy on each link. l_f reflects the binary prediction accuracy on each link in each cascade, while l_t further unveils the prediction performance at each time step. To the best of our knowledge, this is the first attempt to evaluate these three levels of metrics.

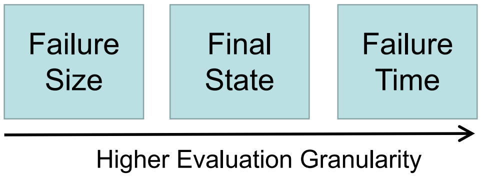


Fig. 4. 3 Layers of Evaluation Granularity

2) Link-based metrics: We consider the following four link-based metrics. We use an example to explain these metrics. We focus on a specific link i, and its final state in 8 different cascade sequences is respectively $[0, 1, 1, 0, 0, 1, 0, 1]$, in which link i fails initially in the 4th sample.

- **Link i Final State Prediction Error Rate** $l_{f,i}$: $l_{f,i}$ denotes the ratio of corrected final state prediction for link i among all test samples except for cases that link i trips initially. In the example, if we predict its final states to be $[1, 1, 0, 0, 0, 0, 1, 0]$, then $l_{f,i} = 4/7 \approx 57\%$.
- **Link i Final State Miss Detection Rate** $l_{md,i}$: $l_{md,i}$ denotes the ratio of predicting link i to be normal among all test samples that link i fails, except for cases that link i trips initially. It reflects the ratio of failure misidentification. In the example, if we predict its final states to be $[1, 1, 0, 0, 0, 0, 1, 0]$, $l_{md,i} = 2/3 \approx 67\%$.
- **Link i Final State False Alarm Rate** $l_{fa,i}$: $l_{fa,i}$ denotes the ratio of predicting link i to be failed among all test samples that link i is normal during the cascade process except for cases that link i trips initially. In the above example, if we predict its final states to be $[1, 1, 0, 0, 0, 0, 1, 0]$, $l_{fa,i} = 3/4 = 75\%$.
- **Link i Time Prediction Error** $l_{t,i}$: $l_{t,i}$ denotes the average failure time prediction error for link i among all test samples that link i fails except for cases that link i trips initially. Suppose in the above example, the time record of link i is $[2, 7, 8, 1, 5, 7, 4, 6]$ and our prediction is $[3, 5, 9, 1, 4, 2, 7, 6]$, then $l_{t,i} = \frac{1}{3}(|2-3| + |5-4| + |4-7|) \approx 1.67$, as link i fails during the cascade process only in the 1st, 5th, and 7th sample.

Compared with the sample-based metrics, link-based metrics capture different prediction difficulties among different links. For example, if a link i fails in 5% of all test samples, then it is trivial to achieve $l_{f,i}$ to be 5% by always regarding i to be normal, hence for this link, our prediction is valid only if $l_{f,i} < 5\%$ by our method. In addition, to achieve low $l_{md,i}$ is hard for such a link, as within 100 test samples, if in 5 of them link i fails, and there is only one misidentified failure of i, then $l_{md,i} = 20\%$. In the opposite, if for link i, $l_{f,i}$ is very large, then to achieve low $l_{fa,i}$ is hard. Moreover, link i which fails at frequency 50% will lead to the largest $l_{f,i}$ under the trivial prediction. Therefore, we should evaluate the prediction performance on each link based on their failure frequency.

C. Performance Evaluation on DC systems

In this part, we present the evaluation results for failure cascade prediction over 4 power systems under DC flow model. In each system, we test on 3 different cases of loading, and evaluate both sample-based and link-based metrics.

1) Sample-based metrics: We summarize our main results for failure cascade prediction in order of granularity. For 118-bus system, we proportionally increase the capacity value of all the links while keeping the power generation and loading invariant, and induce FoS to be 1.2, 1.5 and 1.8. For larger systems, we proportionally increase power generation and loading to study the performance under different loading conditions. In the 1354-bus system, we take $1\times$, $1.5\times$ and

[5] If we predict a link to be normal, then the corresponding number in the second row is our predicted termination time of this cascade.

2× of its original power injection and loading, while in the 2383-bus and the 3012-bus system we take 1×, 1.25×, and 1.5×, where the lower power injection and loading increment than that in 1354-bus system is because the default settings for these two systems are measured at peak power in winter.

Under all the settings, we evaluate the prediction performance on $2,000$ test samples (different from any training sequence in $\mathcal{S}_{train}$). For each metric, we present two categories of results. The first is how the performance varies from $5,000$ to $50,000$ training samples. The second is the cumulative distribution function (CDF) of the corresponding error metric on each test sample with $50,000$ training samples.

Level 1–Failure Size l_{size}: Fig. 5 shows the results on l_{size}. The column on the left captures the value of l_{size} under more training samples. Generally, by incorporating more training samples, l_{size} becomes lower and drops below 7%. For example, if 200 links fail eventually, then our predicted failure size will be within $[186, 214]$ on average. Besides, the decreasing trend of l_{size} is obvious before $20,000$ in most cases, and then diminishes after that, indicating that our approach can offer good enough failure size prediction with limited cascade records in these cases. The column on the right shows the CDF plots. We can find that in each case, l_{size} is less than 10% in more than 85% of test samples, and almost all predictions will not make l_{size} larger than 30%.

Level 2–Final State l_f: Fig. 6 presents the results on l_f. We can observe that l_f declines generally as more training samples are involved, and l_f can be lower than 10% in most cases. This means that among $1,000$ effective links, we can predict more than 900 of them correctly in average under different initial outages. The CDF plots further demonstrate that for at least half of the test samples, l_f will be smaller than 10% among all tested conditions, and altogether more than 95% of all the predictions cause l_f smaller than 30%.

Level 3–Failure Time l_t: Fig. 7 illustrates the results on l_t. l_t mostly decreases under more training samples, while in other settings the error keeps stable at a low level. In most of cases, l_t is within 1 time step, showing that failure time prediction is valid. For example, in 1354-bus system the cascade generally lasts for 10 time steps, if it lasts for 20 minutes in reality, then our prediction can cause error within 2 minutes in most situations. In perspective of CDF, we can further draw that in most cases, around 90% of test samples can achieve time error within 2 steps, while larger l_t generally appears in medium loading conditions. This may stem from longer cascade duration under medium loading, i.e., more time steps before the cascade terminates, since under high loading the failure propagates rapidly and terminates in few steps, while under low loading it also stops early due to the limited failure size.

In addition, note that in each of the 3-level evaluation above, the decreasing trend of the metrics is not strict. This is because the training samples we gather only accounts for very limited amount, hence there might exist deviations and fluctuations. However, the general decreasing trend has evinced the validity of prediction based on learning the influence model.

2) Link-based metrics: In this part, we focus on the four link-based metrics. In the evaluation, we classify all the links in the systems into 10 groups based on their frequency of experiencing failure caused by the cascade among all the training samples. The groups include failure frequency intervals: $[0\%, 10\%), [10\%, 20\%), \ldots, [90\%, 100\%)$. For example, among all $10,000$ cascade samples, if link i fails $1,202$ times while it fails at the beginning before cascade starts for 2 times, then its failure frequency is calculated as $(1202-2)/10000 = 12\%$, hence belonging to the frequency interval $[10\%, 20\%)$. We will evaluate the prediction performance on each group of links, as we have mentioned earlier that different link groups turn in different prediction difficulties under different metrics.

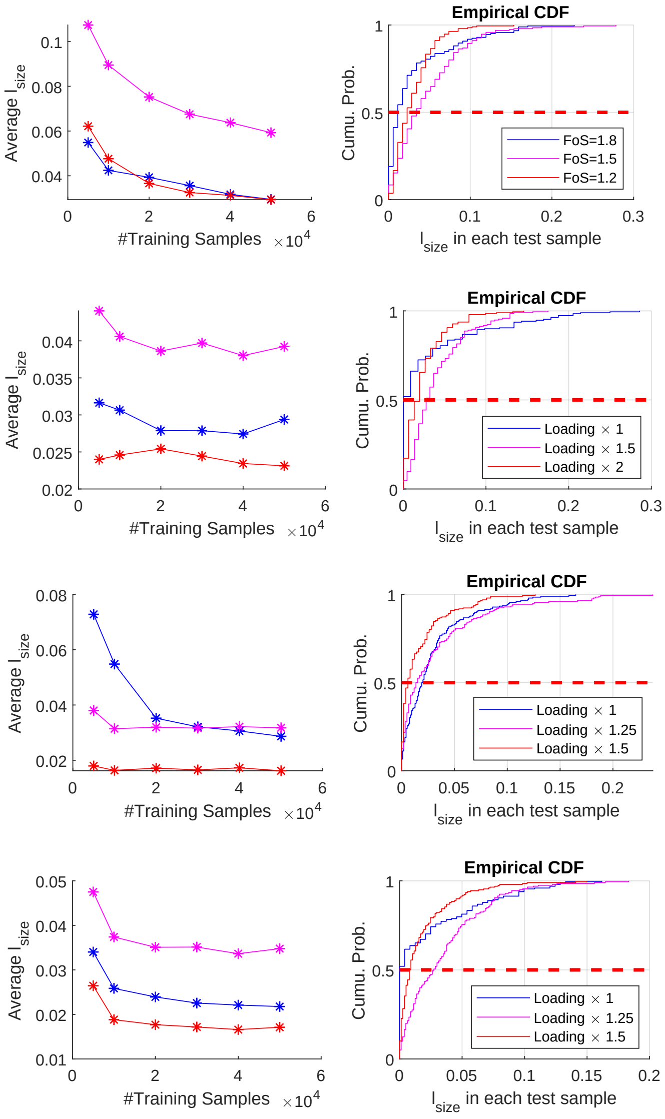


Fig. 5. Results on l_{size}: 118, 1354, 2383, 3012-bus systems in order

Fig. 8, 9, 10 and 11 present the results on link-based metrics over the four systems under DC flow model. We only present one loading condition for each system, where FoS$=1.8$ for 118-bus case, 2× default loading for 1354-bus case, and 1.5× default loading for 2383-bus and 3012-bus cases. Other loading conditions lead to results with similar pattern. For each system, the four subfigures represent the evaluation on $l_{f,i}$, $l_{md,i}$, $l_{fa,i}$ and $l_{t,i}$ in order. For all the metrics, we calculate

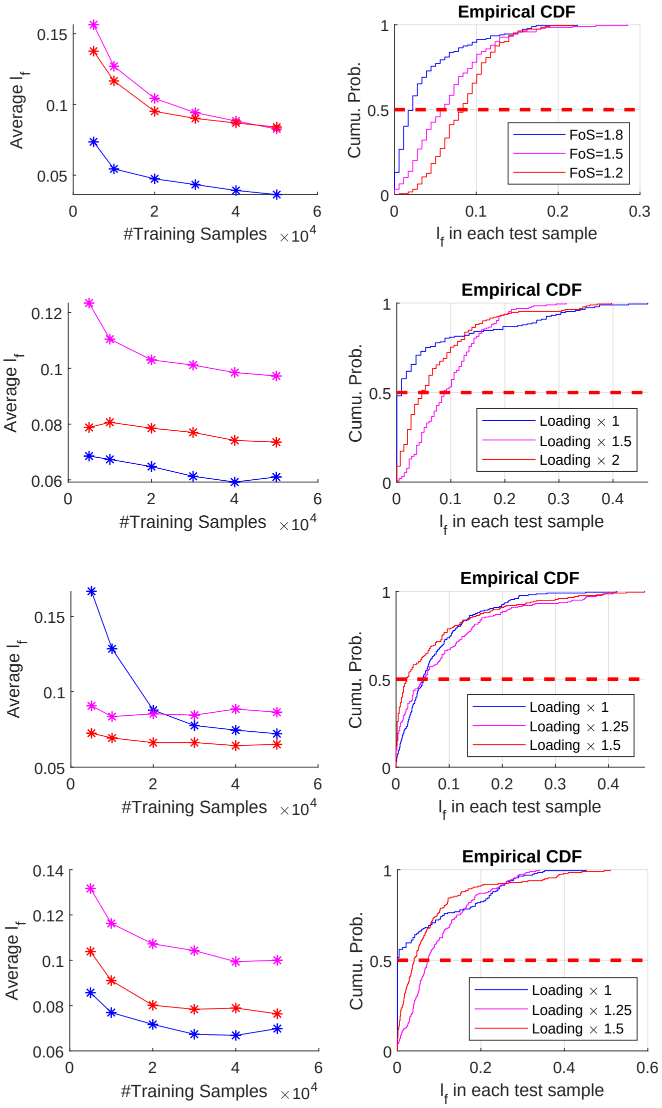


Fig. 6. Results on l_f: 118, 1354, 2383, 3012-bus systems in order

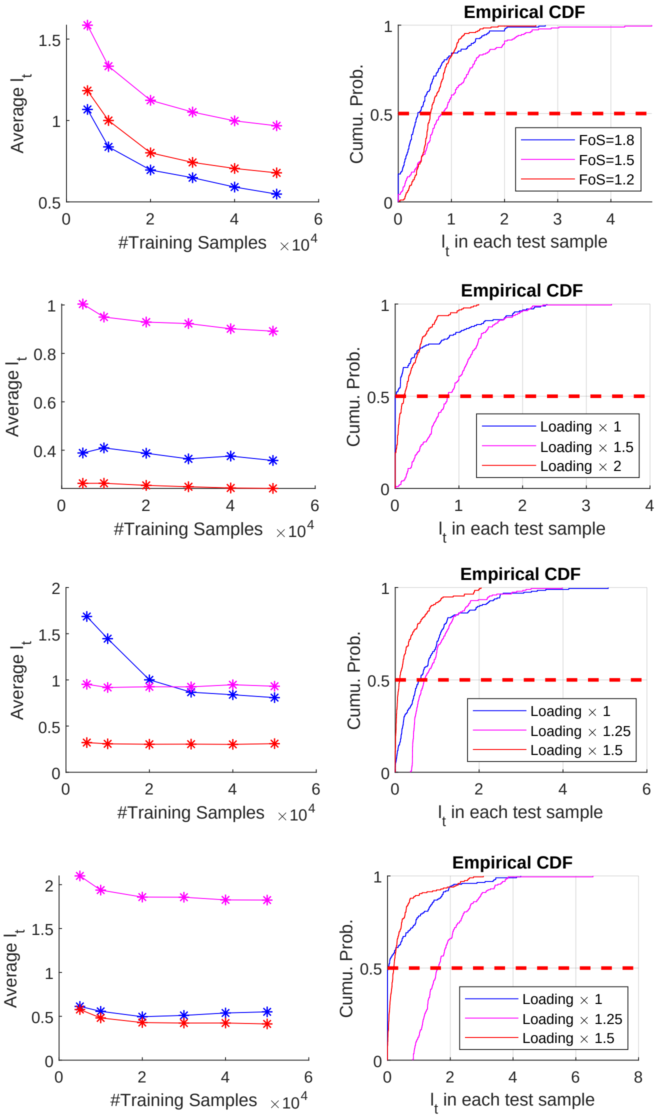


Fig. 7. Results on l_t: 118, 1354, 2383, 3012-bus systems in order

the average metric value among all the links in each failure frequency interval, marked as asterisks in each subfigure.

In our experiment we calculate the failure frequency of each link based on the 50,000 training cascade samples, and we evaluate the prediction using 5000, 10000, 30000, 50000 of them. Besides, to show that our prediction mechanism literally works, we compare it with two trivial prediction methods: deterministic trivial prediction and randomized trivial prediction. Deterministic trivial prediction is to always predict a link to be failed if its failure frequency is larger than 50%, and to be normal if its failure frequency is smaller than 50%. Randomized trivial prediction is to predict a link to be failed with probability equal to its failure frequency. In each subfigure, we use the light blue curve to represent the deterministic one, and the red curve to represent the randomized one.

We explain the results on the four metrics in order.

- For the final state error rate $l_{f,i}$, we can observe that among all the failure frequency regions, the prediction error based on the trained influence model is lower than the trivial methods. Meanwhile, with increasing number of training samples, $l_{f,i}$ in each interval decreases, especially for links with failure frequency around 50%. This shows that our prediction mechanism can effectively reduce the prediction error mostly for links hard to predict its final state by trivial methods.
- For the miss detection error rate $l_{md,i}$ and the false alarm error rate $l_{fa,i}$, they are in fact metrics with mirror symmetry to some extent. For $l_{md,i}$, the deterministic trivial method will have zero error rate for links with failure frequency higher than 50%, as it always predicts these links to be failed. In the opposite, the prediction error rate is 100% for the links with failure frequency lower than 50%. Comparing the trivial methods with our prediction mechanism, we can observe that the prediction based in influence model induces lower miss detection rate than the randomized naive method. Meanwhile, links with higher failure frequency leads to generally lower

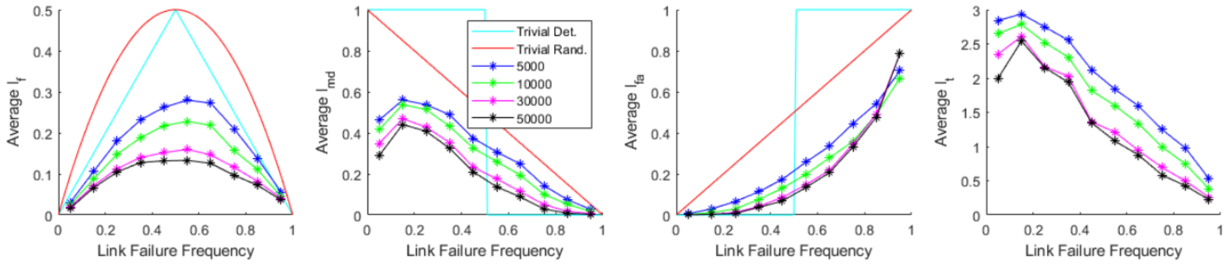


Fig. 8. Results on each link for 118-bus DC system with FoS=1.5

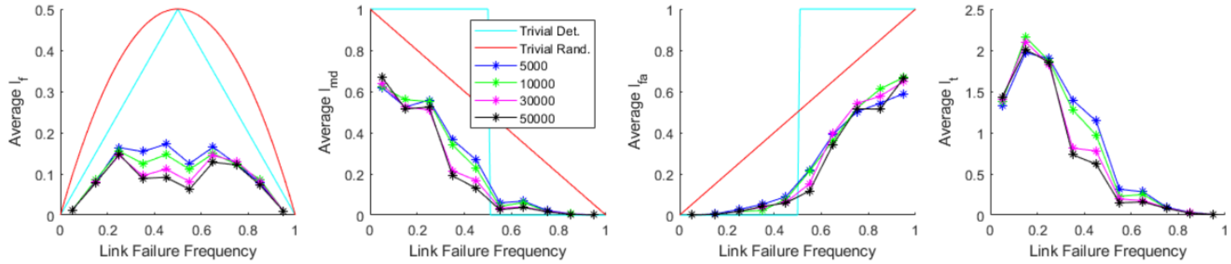


Fig. 9. Results on each link for 1354-bus DC system with 2× default loading

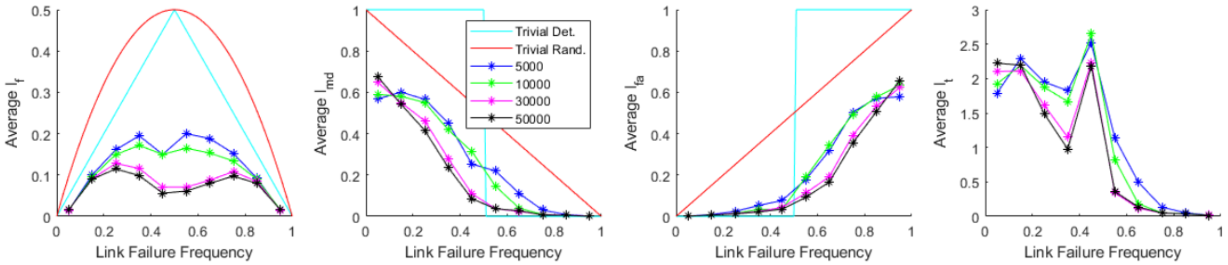


Fig. 10. Results on each link for 2383-bus DC system with 1.5× default loading

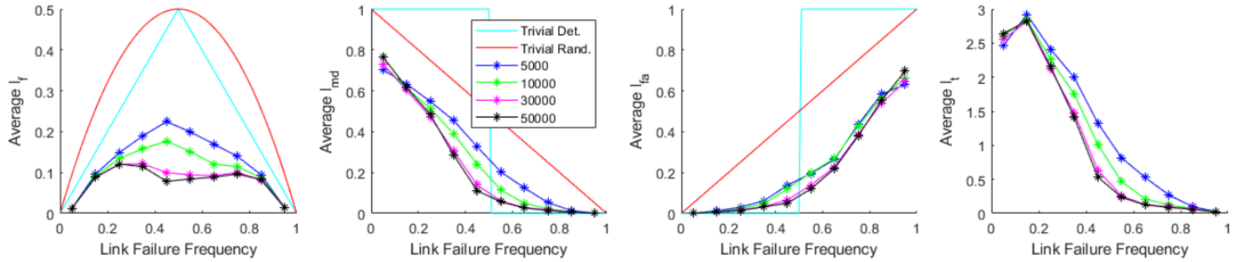


Fig. 11. Results on each link for 3012-bus DC system with 1.5× default loading

miss detection rate. This is because such links have more failure record in the training set, which instructs the model to be prone to predict the failure of these links. In addition, the links with lower failure frequency itself fail less frequently, hence as we have mentioned, whenever we fail to identify one failure of it in the test set, the miss detection rate will be large. For the false alarm error rate $l_{fa,i}$, we will have similar analysis but opposite results, where links fail with high frequency generally lead to large false alarm rate. Therefore we should combine $l_{md,i}$ and $l_{fa,i}$ together to evaluate the performance and explain the results.

- For the time error $l_{t,i}$, we also use the prediction error of time steps to represent. We can find that the prediction error for links with higher failure frequency is generally lower. The reason is similar to that for the miss detection: links fail with high frequency feeds more information about its failure to the model. Besides, the prediction error for links with lower failure frequency is large, as they tend to be predicted to be normal, and whenever it is predicted normal, the time prediction error is large. For example if the cascade lasts for 9 generations, our prediction on its time step is 10 (since it does not fail in the previous 9 generations) while the ground truth might be that this link fails at step 2, thus inducing an error of 8 time steps.

Summarizing the performance patterns in all the systems, we can draw that the trained influence model is really conducive to the failure cascade prediction compared with trivial methods, and it benefits most to the links whose failure is hard to predict in trivial methods.

D. Performance Evaluation on AC systems

The evaluation in AC systems is similar to that in DC systems. We consider both sample-based and link-based metrics, and consider different FoS and loading conditions. Due to the similar pattern as DC systems, we only present the result over two systems: 118-bus and 2383-bus systems, and in each system we only consider two FoS/loading conditions: FoS$=1.2, 1.5$ for 118-bus system, and $1\times$, $1.25\times$ default loading for 2383-bus systems.

1) Sample-based metrics: Fig. 12 to 14 show the result of the three sample-based metrics in 118-bus, 1354-bus, and 2383-bus system under AC flow. In 118-bus system, we can observe that as we increase the training samples to 50,000, the failure size error rate can be reduced below 8%, the final state prediction error rate can be down to 12%, and the time prediction error is lower than 1 time step. Meanwhile, the CDF under 50,000 training samples informs us that most of the test samples can be predicted within reasonable range. In 1354-bus system, note that we only consider at most 5,000 cascade samples for training. This is because the cascade pattern under the AC flow model in this system is not complex, hence with 5,000 samples the prediction accuracy decreases to some stable level. In 2383-bus system, due to the heavy time burden to generate a cascade sample based on calculating the AC flow, we only present the results training at most 30,000 samples. In both of these two large systems, we can still observe a decreasing trend generally on all the metrics with more training samples.

Besides, by comparing the DC and AC system under the same capacity and loading conditions, we can find that the prediction performance in AC systems is a bit inferior to that in DC systems, which matches to the fact that the AC flow model is more complex.

2) Link-based metrics: Fig. 15 and 17 present the results of link-based metrics in 118-bus and 2383-bus system under AC flow. Similar in the trend compared with DC cases, we can find that prediction based on the trained influence model is much more accurate than the trivial methods, and more training samples will reduce the error rate among all these four metrics. The performance is also inferior compared with that under DC flow under the same condition.

Two additional things should be noted. (i) We lack the miss detection rate for links with failure frequency less than 10% in 118-bus system under AC flow, which is because these links do not experience any cascading failure, i.e, their failure frequency is 0. (ii) Under both DC and AC cases, while we increase the number of training samples, the miss detection rate for links fail with low frequency may even be larger than that under fewer training samples, and similar in the false alarm rate for links fail with high frequency. This may be

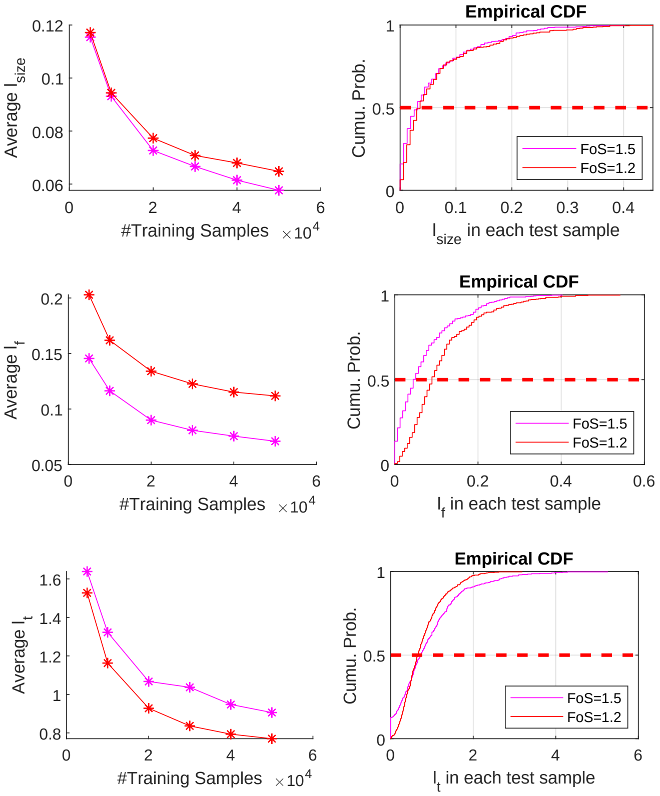


Fig. 12. Results on l_{size}, l_f, l_t in order for 118-bus AC system

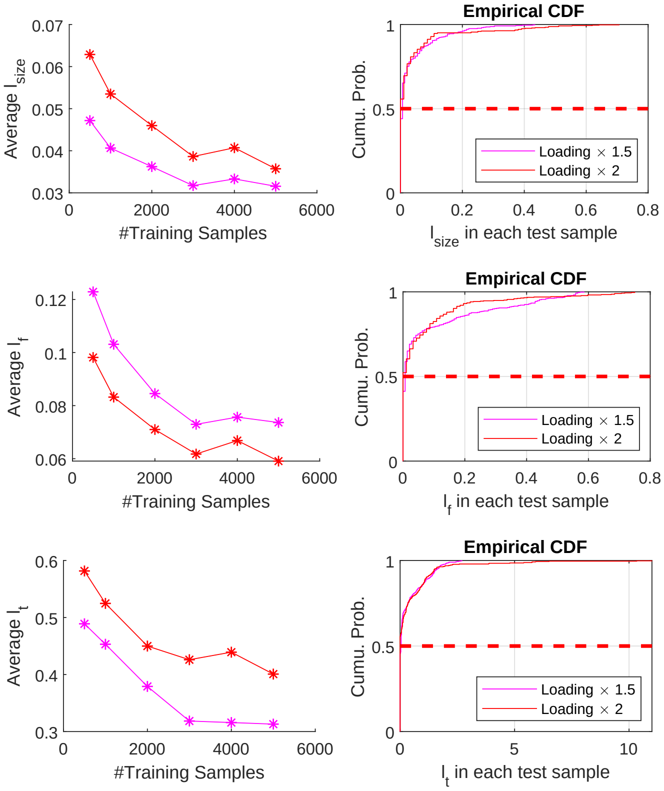


Fig. 13. Results on l_{size}, l_f, l_t in order for 2383-bus AC system

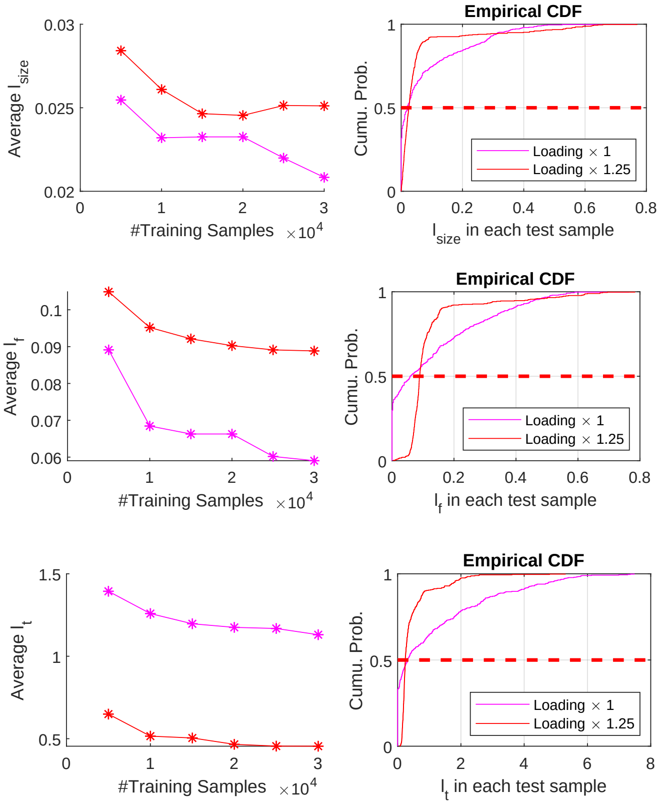


Fig. 14. Results on l_{size}, l_f, l_t in order for 2383-bus AC system

because for the links rarely fail, more training samples may furthermore 'dilutes' the cases of the cascading failure on such links, hence leading the prediction of the final state of such links to be normal and thus arousing even larger miss detection rate. The interpretation on false alarm rate is similar.

E. Prediction Time Comparison

We show the superiority of the influence model based prediction to power flow calculation by MATPOWER Toolbox in time cost reduction. Specifically, we test both methods in MATLAB 2019a on Intel(R) Core(TM) i9-7920X CPU@2.90GHz Processor with 128GB installed memory. In each case, we run $1,000$ test sample via each method and compute its total time cost.

We summarize the results in Table II. We use '$a \mid b \mid \mathbf{c}$' structure to present our results, where a denotes total seconds by DC flow calculation, b denotes total seconds by our method, and $\mathbf{c} = a/b$, reflecting how many times faster the IM is compared to the traditional power flow method. Results show that our method works better in larger systems and under higher loading conditions that tend to cause large failures, which demonstrates its effectiveness and scalability. For example, the time cost by our method is $1/136$ of that by flow calculation under medium loading in the 3012-bus system. The main reason is that more islands appears under these conditions, which requires to solve DC flow equations for more times, while the prediction by influence model discards all such calculations.

TABLE II
PREDICTION TIME COST ON 1,000 SAMPLES IN DC SYSTEM

	Low Load	Medium Load	High Load
1354-bus	808 | 21.3 | **38**	1930 | 19.8 | **97**	1740 | 19.6 | **89**
2383-bus	2597 | 43.3 | **60**	3490 | 37.8 | **92**	3603 | 34.7 | **104**
3012-bus	3891 | 59.4 | **66**	8020 | 58.9 | **136**	5864 | 46.9 | **125**

VII. CASCADE PROPERTY ANALYSIS

In this section, we probe into the insights that the influence model trained by the hybrid learning framework can offer, which we cannot directly obtain from observation. We firstly observe the sparsity of influence in power systems, and further explore the relationship between influence and physical distance among two links. Then we identify critical initial contingencies simply based on the trained parameters. Such analysis uncovers broader usage of the influence model.

A. Influence Sparsity

A very interesting observation of the influence is that the learned relative influence matrix $\mathbf{D}$ by (11) appears to be *sparse*, where most of the elements are very close to 0. This means that in power systems, each link is mainly influenced by very small number of other links. To study on this phenomenon, we test on 118-bus and 2383-bus cases under DC flow, and show the spatial distribution of $\mathbf{D}$ values that are larger than 0.01 in Fig. 18. We define an influence to be *visible* from link i to j if $d_{ji} > 0.01$. Note that there are 310 and 9904 visible influences in each system respectively. Under such setting, as there are 171 and 2088 effective links respectively in 118-bus and 2383-bus systems, hence in each row of $\mathbf{D}$, there are averagely $310/171 = 1.8$ and $9904/2088 = 4.7$ visible influences, accounting for $1.8/179 = 1.0\%$ and $4.7/2886 = 0.2\%$ among all links in each system (note that ineffective links may also fail initially to affect others). Based on the sum of each row in $\mathbf{D}$ is 1, the result indicates that only very few links dominates the influence on a specific link. In all, network influence can be sparse enough to represent most of correlation within the power systems.

Additionally, we can observe that $\mathbf{D}$ is generally but not completely symmetric in terms of its diagonal, which implies that the influence between two links are not always bidirectional. Meanwhile, many visible influences are gathering around diagonals, and based on indexing rule, many adjacent links have very close link indexes, hence the result shows that the local influence is still a major role in power networks. To this observation, we will show in next part with more details by quantifying the relationship between influence and physical distance, a step further than focusing solely on links adjacent or not.

B. Influence and Distance

Indicated by Kirchhoff's law, adjacent links should be correlated more tightly, hence share a stronger influence. However, as capacity values vary among different links, the cascade process may not follow in this way. In this part we explore the

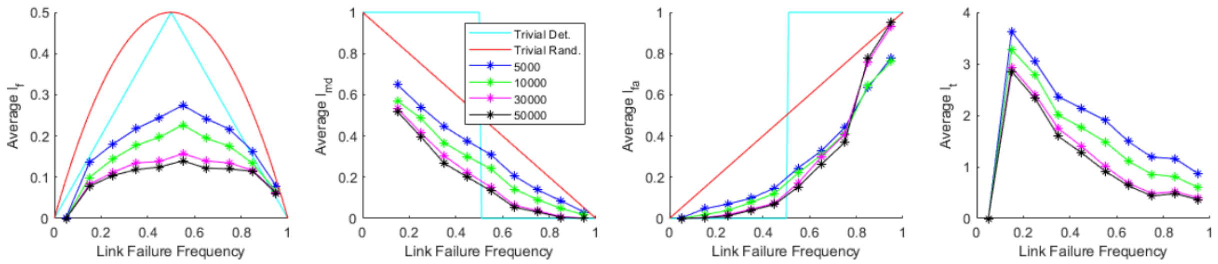


Fig. 15. Results on each link for 118-bus AC system with FoS=1.2

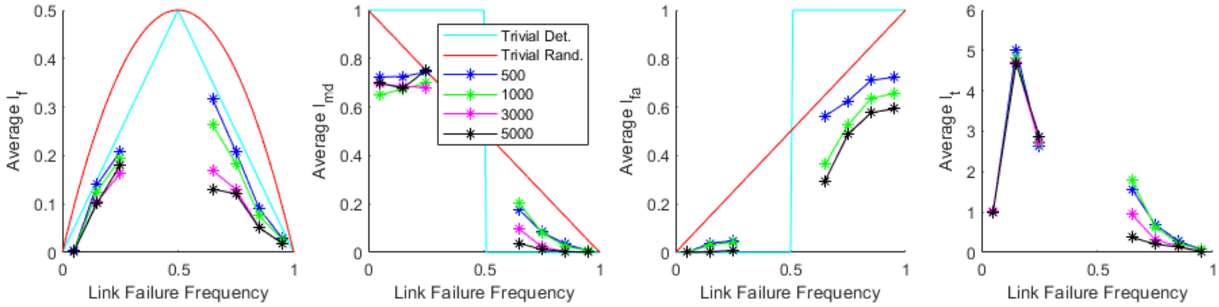


Fig. 16. Results on each link for 1354-bus AC system with 2× default loading

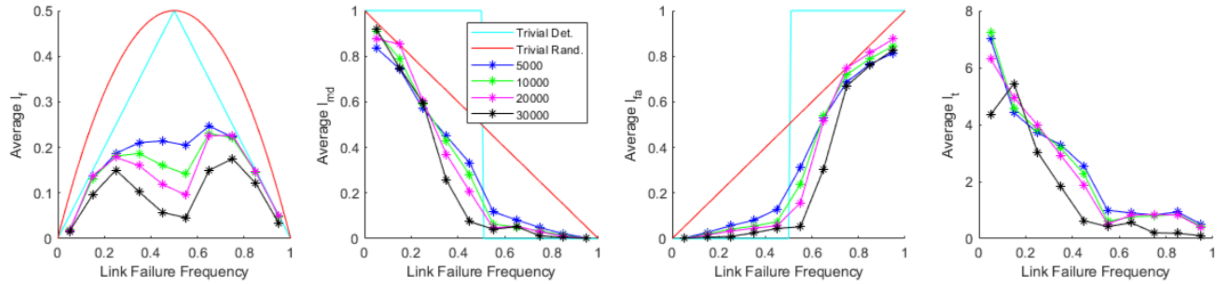
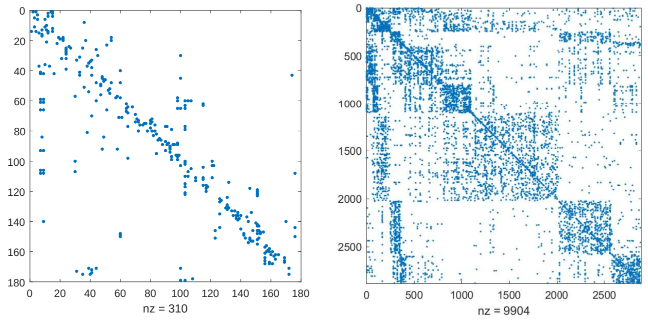


Fig. 17. Results on each link for 2383-bus AC system with 1× default loading



(a) 118-bus

(b) 2383-bus

Fig. 18. Spatial distribution of $\mathbf{D}$ values larger than 0.01 in 118-bus and 2383-bus system in default setting under DC flow.

relationship between influence and physical distance between any pair of links. The influence is characterized by $\mathbf{D}$, $\mathbf{A}^{11}$ and $\mathbf{A}^{01}$. Based on (1), we can find that $\tau_{ij} \triangleq d_{ij}(\mathbf{A}_{ji}^{11} - \mathbf{A}_{ji}^{01})$ reflects the *total influence* from link j to i, as when $s_j[t]$ turns from 1 to 0, the value of $\tilde{s}_i[t+1]$ will be decreased by such total influence. Here we term the total influence from link j to link i is *visible* if $\tau_{ij} > 0.01$. The physical distance is defined as the minimum topologically hops between two links. For example, if link i and j are adjacent, then the distance between them is 1; if they are not but connected by another common link k, then the distance between link i and j is 2.

We present the results over 118-bus and 1354-bus systems under DC systems and their default settings in Table III, Fig. 19 and 20. Table III lists our measurement on 4 aspects on both systems: the total number of link pairs with a specific distance (#LinkPairs), the rate of visible link pairs compared with all those under such distance (Vis. rate), the average of τ among all link pairs ($\mathbf{E}[\tau]$), and that among all visible ones ($\mathbf{E}[\text{Vis. } \tau]$). We use '$a \mid b$' structure to present the result in 118-bus and 1354-bus systems respectively. Fig. 19 and 20 illustrate the distribution of τ values larger than 0.01, where the abscissa axis denotes different τ values. We discard the part where $\tau < 0.01$ as it takes very large proportion, implied in the discussion about influence sparsity in Section VII-A. Including it will make the distribution concentrate near 0.

Such truncation, however, does not matter, as Fig. 19 and 20, together with Table III, can present all the related information. For example, in 118-bus case, suppose we consider links that have distance 1, then there are $1018 \times 11.8\% \approx 120$ effective links. Then by Fig.19, we can calculate that the number of link pairs whose total influence τ is within $(0.5, 0.55)$ should be $120 \times 1.67\% = 2$. From the records, especially the column (**E**[Vis. τ]) in Table III, we can clearly identify that in both systems, the major influence still exists in adjacent link pairs, and as distance becomes larger, the influence values decline generally but does not approach 0. This shows that under both capacity settings, local influence is stronger while far-away influence can never be neglected.

TABLE III
INFLUENCE-DISTANCE ANALYSIS RESULTS FOR 118|1354-BUS SYSTEM IN DEFAULT SETTING.

	#LinkPairs		**Vis. Rate**		**E[τ]**		**E[Vis. τ]**	
	118	**1354**	**118**	**1354**	**118**	**1354**	**118**	**1354**
Dist.= 1	1018	10242	11.8%	2.4%	.030	0.003	0.25	0.33
Dist.= 2	2166	22892	4.2%	1.4%	.007	0.003	0.16	0.26
Dist.= 3	3180	42308	1.2%	0.7%	.002	0.001	0.15	0.21
Dist.= 4	3810	71360	0.5%	0.4%	.001	0.001	0.13	0.16

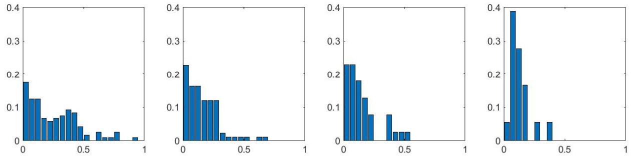


Fig. 19. Visible Influence Value Distribution in 118-Bus System.

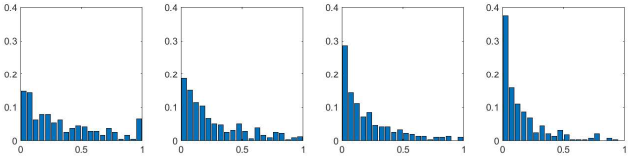


Fig. 20. Visible Influence Value Distribution in 1354-Bus System.

C. Critical Initial Outages

As the trained influence model can predict the cascade sequence based on any combination of initial outages, we consider whether it is possible to identify the critical $M-k$ initial contingencies, which can arouse failure cascade with large size. Intuitively based on (1), a link whose failure that has enormous negative effects should greatly reduce the intermediate state values $\tilde{s}[t]$ of other links in (1) during the cascade, as this will drag these value down below the threshold of link i with higher chances. Therefore, a natural idea to identify the links that may cause large failure size is to find the link j^* such that

$$j^* = \arg\max_j \sum_{i=1}^{M} \tau_{ij}. \tag{15}$$

Such link j^* denotes the link whose failure is most prone to lead large failures. Note that we can order all links $j = 1, 2, \ldots, M$ based on $\sum_{i=1}^{M} \tau_{ij}$. Then to find the set of links that may lead to large failures, and thus identify the most critical $M-k$ initial contingency, we can find k links $\{j_l\}_{l=1}^{k}$ in terms of maximizing $\sum_{i=1}^{M} \tau_{ij}$. On the other hand, if we replace $\max$ with $\min$, then it finds the links and the initial contingencies that may cause fewest failures.

To verify this, we test on the 118-bus systems, under both DC and AC flow. For DC flow, we consider FoS being 1.5 and 1.8. In each setting, we construct $50,000$ cascade sequences and obtain the empirical distribution of ultimate failure size, denoted as p_0. Then we find top 10 links that maximize/minimize $\sum_{i=1}^{M} \tau_{ij}$ in terms of j, denoted as $J_{max} \triangleq \{j_l^{max}\}_{l=1}^{10}$ and $J_{min} \triangleq \{j_l^{min}\}_{l=1}^{10}$ respectively. For J_{max}, we enumerate all $M-k$ contingencies where initially failed links are all in J_{max}, and find the empirical failure size distribution p_{max}. For example, if $k = 3$, then it involves $\binom{10}{3}$ initial outage combinations. Similarly we can obtain p_{min} from J_{min}. By comparing p_{min}, p_{max} with p_0, we can directly figure out whether our choices of J_{min} and J_{max} based on the trained influence model can really identify the most critical and non-critical $M-k$ initial contingencies.

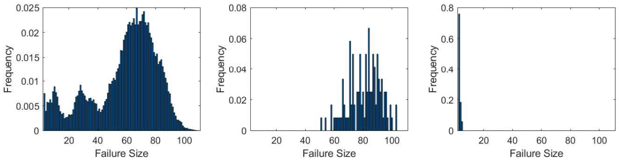


Fig. 21. Failure Size Distribution for 118-Bus FoS=1.5 (DC).

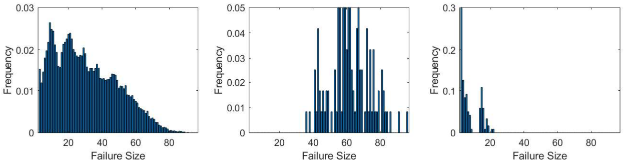


Fig. 22. Failure Size Distribution for 118-Bus FoS=1.8 (DC).

Fig. 21 and 22 present the results for 118-bus case under DC flow. The leftmost subfigure shows the empirical failure size distribution p_0, while the middle and rightmost reflect p_{max} and p_{min} respectively. We can observe that in both cases, p_{max} tilts to the large failure size, with mean size 81 and 62, while p_{min} concentrates on very small failure size, with mean size 11 and 8 in these two settings respectively. The violent difference between p_{max} and p_{min} demonstrates the validity of our method to identify initial outages that are critical and non-critical. Fig. 23 shows the results over AC flow case where FoS$=$ 1.8. We can find that it can also identify the initial contingencies that will lead to large and small cascade failure size. However its identification is not so accurate as that in DC flow, which remains to be further enhanced in future.

Furthermore, we show the spatial distribution of J_{min} and J_{max} for both FoS conditions in 118-bus system in Fig. 24(a) and 24(b). The yellow links are 10 most critical links, while the green ones are the 10 most non-critical. For the two loading conditions, some links keep to be very critical, while there also

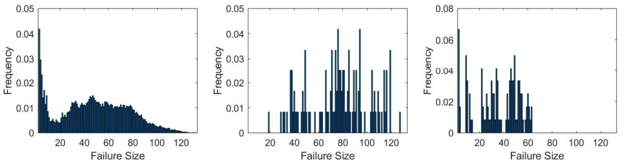
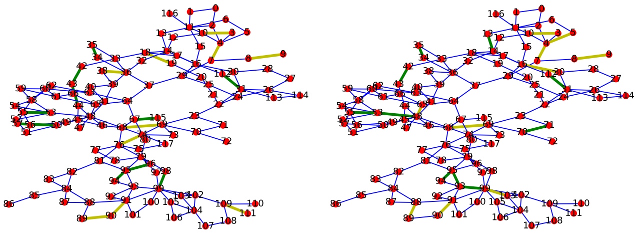


Fig. 23. Failure Size Distribution for 118-Bus FoS=1.5 (AC).



(a) (b)

Fig. 24. Spatial Distribution of Critical/Non-Critical Links in 118-Bus System under DC flow: (a) FoS=1.5, (b) FoS=1.8

exists some difference in the links. This can tell us the links consistently critical under load fluctuation. Meanwhile, the critical links are spatially remote from each other, indicating that local management is not enough to identify them.

VIII. Conclusion

In this paper, we build an influence model framework to study and predict failure cascades in large scale power systems. We propose a hybrid learning scheme to train the influence model based on simulated failure cascade samples. Experimental results on large scale power systems demonstrate acceptable prediction performance over metrics with different levels of evaluation granularity, and we further show that our method bears a two-order-of-magnitude time cost reduction compared with power flow simulation based approach. In addition, we show that the influence model trained by the hybrid framework can reflect cascade properties such as influence sparsity, the relationship between influence and physical distance, and the critical and non-critical initial contingencies.

However, some unsolved issues still exist in our methodology. For example, when a link rarely fails, it is hard to identify its failure based on the current learning framework without additional operations, and thus leading to high miss detection rate. This enlightens us to renovate our training structures and even details including more refined way to estimate the link threshold values ϵ. Moreover, the application of this framework into other network systems, including communication systems and even interdependent network systems, is worth investigation as this learning framework over influence model does not rely on any specific network model.

References

[1] "NorthEast US Failure Cascade," https://www.bostonglobe.com/magazine/2012/02/03/anatomy-blackout-august/mAsrr41nLAjGFIU3IF440O/story.html.

[2] "Manhattan, New York Failure Cascade," https://www.theatlantic.com/technology/archive/2019/07/manhattan-blackout-reveals-infrastructure-risk/594025/.

[3] "London Failure Cascade," https://www.bloomberg.com/news/articles/2019-08-09/london-blackout-occurred-amid-drop-in-wind-and-natural-gas-power.

[4] Z. Wang, M. Rahnamay-Naeini, J. M. Abreu, R. A. Shuvro, P. Das, A. A. Mammoli, N. Ghani, and M. M. Hayat, "Impacts of operators behavior on reliability of power grids during cascading failures," *IEEE Transactions on Power Systems*, vol. 33, no. 6, pp. 6013–6024, 2018.

[5] H. Cetinay, S. Soltan, F. A. Kuipers, G. Zussman, and P. Van Mieghem, "Comparing the effects of failures in power grids under the ac and dc power flow models," *IEEE Transactions on Network Science and Engineering*, vol. 5, no. 4, pp. 301–312, 2017.

[6] A. Bernstein, D. Bienstock, D. Hay, M. Uzunoglu, and G. Zussman, "Power grid vulnerability to geographically correlated failuresłanalysis and control implications," in *IEEE INFOCOM 2014-IEEE Conference on Computer Communications*. IEEE, 2014, pp. 2634–2642.

[7] ——, "Sensitivity analysis of the power grid vulnerability to large-scale cascading failures," *ACM SIGMETRICS Performance Evaluation Review*, vol. 40, no. 3, pp. 33–37, 2012.

[8] S. Soltan, D. Mazauric, and G. Zussman, "Analysis of failures in power grids," *IEEE Transactions on Control of Network Systems*, vol. 4, no. 2, pp. 288–300, 2015.

[9] S. Soltan, A. Loh, and G. Zussman, "Analyzing and quantifying the effect of k-line failures in power grids," *IEEE Transactions on Control of Network Systems*, vol. 5, no. 3, pp. 1424–1433, 2017.

[10] Z. Kong and E. M. Yeh, "Correlated and cascading node failures in random geometric networks: A percolation view," in *2012 Fourth International Conference on Ubiquitous and Future Networks (ICUFN)*. IEEE, 2012, pp. 520–525.

[11] H. Xiao and E. M. Yeh, "Cascading link failure in the power grid: A percolation-based analysis," in *2011 IEEE International Conference on Communications Workshops (ICC)*. IEEE, 2011, pp. 1–6.

[12] Z. Wang, D. Zhou, and Y. Hu, "Group percolation in interdependent networks," *Physical Review E*, vol. 97, no. 3, p. 032306, 2018.

[13] C. Asavathiratham, S. Roy, B. Lesieutre, and G. Verghese, "The influence model," *IEEE Control Systems Magazine*, vol. 21, no. 6, pp. 52–64, 2001.

[14] M. Rahnamay-Naeini, Z. Wang, N. Ghani, A. Mammoli, and M. M. Hayat, "Stochastic analysis of cascading-failure dynamics in power grids," *IEEE Transactions on Power Systems*, vol. 29, no. 4, pp. 1767–1779, 2014.

[15] P. Das, R. A. Shuvro, Z. Wang, M. R. Naeini, N. Ghani, and M. M. Hayat, "Stochastic failure dynamics in communication network under the influence of power failure," in *2017 IEEE 13th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. IEEE, 2017, pp. 1–8.

[16] R. A. Shuvro, Z. Wang, P. Das, M. R. Naeini, and M. M. Hayat, "Modeling impact of communication network failures on power grid reliability," in *2017 North American Power Symposium (NAPS)*. IEEE, 2017, pp. 1–6.

[17] P. D. Hines, I. Dobson, and P. Rezaei, "Cascading power outages propagate locally in an influence graph that is not the actual grid topology," *IEEE Transactions on Power Systems*, vol. 32, no. 2, pp. 958–967, 2016.

[18] K. Zhou, I. Dobson, Z. Wang, A. Roitershtein, and A. P. Ghosh, "A markovian influence graph formed from utility line outage data to mitigate cascading," *arXiv preprint arXiv:1902.00686*, 2019.

[19] M. Vaiman, K. Bell, Y. Chen, B. Chowdhury, I. Dobson, P. Hines, M. Papic, S. Miller, and P. Zhang, "Risk assessment of cascading outages: Methodologies and challenges," *IEEE Transactions on Power Systems*, vol. 27, no. 2, p. 631, 2012.

[20] H. Cetinay, S. Soltan, F. A. Kuipers, G. Zussman, and P. Van Mieghem, "Analyzing cascading failures in power grids under the ac and dc power flow models," *ACM SIGMETRICS Performance Evaluation Review*, vol. 45, no. 3, pp. 198–203, 2018.

[21] D. Bienstock and A. Verma, "The nk problem in power grids: New models, formulations, and numerical experiments," *SIAM Journal on Optimization*, vol. 20, no. 5, pp. 2352–2380, 2010.

[22] J. Zhang, E. Yeh, and E. Modiano, "Robustness of interdependent random geometric networks," *IEEE Transactions on Network Science and Engineering*, 2018.

[23] I. Dobson, "Estimating the propagation and extent of cascading line outages from utility data with a branching process," *IEEE Transactions on Power Systems*, vol. 27, no. 4, pp. 2146–2155, 2012.

[24] H. Ren and I. Dobson, "Using transmission line outage data to estimate cascading failure propagation in an electric power system," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 55, no. 9, pp. 927–931, 2008.

[25] M. J. Eppstein and P. D. Hines, “A random chemistry algorithm for identifying collections of multiple contingencies that initiate cascading failure,” *IEEE Transactions on Power Systems*, vol. 27, no. 3, pp. 1698–1705, 2012.

[26] P. Rezaei, P. D. Hines, and M. J. Eppstein, “Estimating cascading failure risk with random chemistry,” *IEEE Transactions on Power Systems*, vol. 30, no. 5, pp. 2726–2735, 2014.

[27] J. Song, E. Cotilla-Sanchez, G. Ghanavati, and P. D. Hines, “Dynamic modeling of cascading failure in power systems,” *IEEE Transactions on Power Systems*, vol. 31, no. 3, pp. 2085–2095, 2015.

[28] D.-X. Zhang, D. Zhao, Z.-H. Guan, Y. Wu, M. Chi, and G.-L. Zheng, “Probabilistic analysis of cascade failure dynamics in complex network,” *Physica A: Statistical Mechanics and its Applications*, vol. 461, pp. 299–309, 2016.

[29] X. Zhang, C. Zhan, and K. T. Chi, “Modeling the dynamics of cascading failures in power systems,” *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 7, no. 2, pp. 192–204, 2017.

[30] J. Qi, J. Wang, and K. Sun, “Efficient estimation of component interactions for cascading failure analysis by em algorithm,” *IEEE Transactions on Power Systems*, vol. 33, no. 3, pp. 3153–3161, 2017.

[31] “IEEE Power Systems,” https://github.com/MATPOWER/matpower.

[32] M. Jaggi, “Revisiting frank-wolfe: Projection-free sparse convex optimization.” in *ICML (1)*, 2013, pp. 427–435.

[33] “Newton-Raphson Method,” https://wiki.openelectrical.org/index.php?title=Newton-Raphson_Power_Flow.