Predicting Failure Cascades in Large Scale Power Systems via the Influence Model Framework

Xinyu Wu, *Student Member, IEEE,* Dan Wu, *Member, IEEE,* Eytan Modiano, *Fellow, IEEE*

***Abstract*—Large blackouts in power grids are often the consequence of uncontrolled failure cascades. The ability to predict the failure cascade process in an efficient and accurate manner is important for power system contingency analysis. In this paper, we propose to apply the influence model for the prediction and screening of failure cascades in large scale DC and AC power networks. Then, the trained influence model is applied to some large power grids with thousands of buses and transmission lines. The prediction performance is evaluated in four different aspects, from the global perspective of the overall failure size to the individual information regarding the link failure time. The results show that under limited training samples, the proposed framework is capable of predicting the failure cascade size with a 7% error rate, the final state of links with a 10% error rate, and the failure time within 1 time unit for both the DC and AC model. One major advantage of the proposed method is that it can reduce the computational time of the failure cascade prediction by a few orders of magnitude with limited compromise in accuracy, as compared to the power flow based contingency analysis. Another important feature of the proposed method is that the trained influence parameters can reveal the critical initial contingencies. This information is very helpful for identifying the worst contingency scenarios for system operators.**

***Index Terms*—Failure cascade, influence model, large scale DC/AC power flow model, contingency analysis**

I. Introduction

Large blackouts in modern power systems are often the result of uncontrolled failure cascades from some initial contingencies. In 2003, a 345kV line tripped off after touching a tree limb. The initial failure expanded from several links to over 500 generating units in the US and Canada within 1 hour [1]. In 2019, midtown Manhattan suffered a widespread blackout which resulted from a disabled transformer [2]. These blackouts are shown to be associated with a rapid propagation of failures from initial contingencies. Such propagation process, termed *failure cascade*, is difficult to analyze and predict in large power systems due to its enormous scale of involved components with complex underlying interactions and time-varying parameters [3].

A number of studies have been carried out from historical records of large blackouts. Vaiman *et al.* listed possible sources of initially tripped links, and further summarized large blackouts that occurred every year from 1984 to 2006 due to different natural disasters [4]. Hines *et al.* discovered from cascade records that the blackout size follows the Weibull distribution when the size is small, and power-law distribution when it is large [3]. These works, based on the historical records, reveal basic statistical properties of failure cascades and paved the way for quantitative modeling and analysis.

However, the scarce historical records of cascading failures are far from fully representing all the possibilities, leaving potential blackouts concealed. To tackle this challenge, numerical simulations and analysis of different initial outages are studied based on power flow models [5]–[10]. The AC flow model suffers from heavy computational burden as it requires solving nonlinear equations iteratively, and it is difficult to find an AC power flow solution when the network configuration changes drastically [5]. For the DC flow model, Zussman et.al. analytically revealed how failure propagates with multiple tripped links [6], [7] based on solving flow equations, while Beinstock and Varma gained insights using the mixed-integer linear optimization [8]. Zussman et.al. also formulated a general failure cascade model under both the deterministic and probabilistic cases predicated on DC flow calculation [9]. Although, the DC power flow model can achieve analytic results and fast simulation speed, Cetinay *et al.* showed that the DC flow model may underestimate the failure size and the AC flow model is more accurate in characterizing real failure cascades [5], [10].

To overcome the high complexity of flow-model-based methods, researchers sought various flow-free approaches. For example, the percolation-based analysis on random graph models has been used to model local cascades whose failure probability depends on neighboring link degree [11]–[14]. This model overlooks the nonadjacent correlations among geographically distant failures, and the analysis does not capture the power system dynamics, which may render the model rather inaccurate in practice. The branching process is a popular tool that measures the distribution of the number of outages in a cascade [15], [16], and the random chemistry algorithm together with such a distribution can efficiently estimate the overall blackout risk [17], [18]. In addition, Hines et.al. applied differential equations to model the cascade process [19] while Zhang et.al. did so by mean field theory [20]. Moreover, Tse et.al. tried chemical master equation model to grasp the collective behavior of failure propagation [21] while Qi et.al. learned such behavior via EM algorithm [22]. These works aspired to propose models that capture the flow dynamics and the failure cascade process. However they still lack explicit quantification of how network components influence each other to render a failure sequence. Moreover, most of the existing flow-free approaches are only specialized in characterizing one or two aspects of failure cascades such as failure size distributions or final states [3], [15], [23]–[27], lacking a comprehensive evaluation at different levels of granularity.

In order to (i) explicitly quantify the superimposed influence

from all the components in power systems, (ii) evaluate failure cascades at different levels of granularity, and (iii) predict failure cascades in a highly efficient way, in this paper, we focus on modeling failure cascades through a flow-free approach using the Influence Model [28], [29]. It is a special graphic model that captures the underlying influence of all the network components on each individual component. It is essentially a Markovian-like 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 [23]–[25]. To apply the influence model to the prediction of failure cascades in power systems, Hines *et. al.* utilized the Monte Carlo method to learn the pairwise influence values [3]. Their trained model generates the appropriate distribution of failure cascade sizes that best match the existing records. Zhou *et. al.* further extended the pairwise influences to capture more complex influences [26], [27] for failure size prediction.

The interaction model [30], [31] shares a similar idea to the influence model approach. For any component j, the interaction model specifies the *cause* of the failure of component j as the components whose failure most frequently leads to the failure of j in the sample pool. Then the probability that the failure of component i causes the failure of j can be computed, and is applied in the failure cascade generation process. The difference between the interaction model and the influence model in [3], [26], [27] is that the influence model first determines the number of outages in the next generation, and samples which component will fail based on the influence value, without specifying the exact cause of the failure of any component j.

In this paper, we propose a hybrid learning framework that can efficiently train the influence model for large systems. The proposed framework integrates the Monte Carlo method, a regression approach, and a novel adaptive threshold selection scheme to train the model in an efficient way and to achieve a better prediction accuracy. A preliminary version of this work appeared in [29]. Here, we extend [29] in the following significant ways: (i) We extend our framework to large-scale power systems in the nonlinear AC flow model, and demonstrate its good prediction performance at all levels of granularity. (ii) By comparing our framework to an appropriate flow-model-based oracle, we show that the proposed method is able to predict failure cascades at a much faster speed (at least two orders of magnitude faster) with acceptable small error rates. (iii) We propose a ranking scheme to sort out the most critical initial contingencies from the learned influence values. It works well in both the DC and the AC power flow models with higher computational efficiency compared with prior works [3], [27], which demonstrates the applicability of our influence model framework.

The rest of the paper is organized as follows: Section II introduces the basics of the influence model. Section III describes our hybrid learning framework. Section IV presents two sets of evaluation metrics that capture different aspects of the failure cascade prediction. Section V summarizes the result of cascade prediction accuracy and time efficiency based on the trained influence model over large scale DC and AC systems. Section VI demonstrates the ability of this model to identify the critical initial contingencies.

II. Influence Model

The influence model is a Markovian-like model whose dynamics are described by the state variable transitions. In the failure cascade analysis, the state variable of the influence model is chosen to be the binary operational state of each transmission line*, taking on values of either 0 (failed) or 1 (normal) respectively [15]. 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.

The transition of a state variable $s_i[t]$ from the current time t to the next time $t+1$ is given by

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

where $\mathbf{A}^{11}$ and $\mathbf{A}^{01}$ are interpreted as

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

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

are the transition probabilities from the state of link j at t to the state of link i at $t+1$; The weight d_{ij} represents the proportional affect from the link j to i, under the constraint that for any link i and j, $\sum_{j=1}^{M} d_{ij} = 1,\ d_{ij} \geq 0$. The d_{ij} can be regarded as the ratio of the influence from link j to i among all the influence over link i. $\tilde{s}_i[t+1] \in [0,1]$ is the estimated value of $s_i[t+1]$. The physical meaning of eqn. (1) is that $\tilde{s}_i[t+1]$ is the weighted sum of the probability that link i being normal at time $t+1$ (i.e. $s_i[t+1] = 1$) given the network state $s[t]$ at time t. Specifically the term $\mathbf{A}^{11}_{ji} s_j[t] + \mathbf{A}^{01}_{ji}(1 - s_j[t])$ can be interpreted as follows. Consider link j for example. Given that $s_j[t] = 1$, then this term becomes $\mathbf{A}^{11}_{ji}$, which is exactly the conditional probability that link i is normal at $t+1$ given link j is normal at t. Similarly when $s_j[t] = 0$.

We collect all the $\mathbf{A}^{11}_{ji}$'s into an $M \times M$ matrix $\mathbf{A}^{11}$, and collect all the $\mathbf{A}^{01}_{ji}$'s into $\mathbf{A}^{01}$. We refer to each of $\mathbf{A}^{11}$ and $\mathbf{A}^{01}$ as the "pairwise influence matrix". We collect $\{d_{ij}\}_{i,j=1,\dots,M}$ into an $M \times M$ matrix $\mathbf{D}$, referred to as the "weighted influence matrix".

An illustrative 5-node example of the influence model can be found in Fig. 1. The left subfigure depicts the pairwise influence of component 1 on 5, concerning $\mathbf{A}^{11}_{15}$ and $\mathbf{A}^{01}_{15}$ in (1). The right subfigure reflects that component 5 is mutually influenced by component 1 and 3, where $\mathbf{A}^{11}_{15}$ and $\mathbf{A}^{01}_{15}$ take weight d_{51}, while $\mathbf{A}^{11}_{35}$ and $\mathbf{A}^{01}_{35}$ take weight d_{53} in (1).

After $\tilde{s}_i[t+1]$ is obtained, to make predictions consistent, we apply the following deterministic bisection scheme with a threshold ϵ_i for each link i rather than doing randomized rounding to $\{0,1\}$:

$$\hat{s}_i[t+1] = \begin{cases} 1 & \text{if } \tilde{s}_i[t+1] \geq \epsilon_i \\ 0 & \text{if } \tilde{s}_i[t+1] < \epsilon_i \end{cases}, \qquad (4)$$

*We also term 'transmission line' as 'link' in the following for brevity.

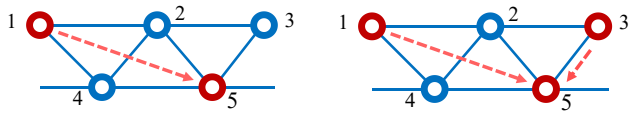


Fig. 1: Influence Model Toy Example

We will propose a way to quantify the threshold value later in Section III-D.

We stack $\hat{s}_i$, the predicted status of each link i, into a vector $\hat{s}[t]$ and refer to it as the prediction of the network state at time t. Then, the sequence $\hat{s}[t]$, denoted as $\hat{s}$, is a predicted failure cascade sequence for the system.

To apply the influence model in failure cascade analysis, the pairwise influence matrices $\{\mathbf{A}^{11}, \mathbf{A}^{01}\}$, the weighted influence matrix $\mathbf{D}$, and the bisection thresholds $\{\epsilon_i\}_{i=1}^{M}$ need to be specified.

III. Learning Influence Model Parameters

In this section, we discuss how to generate the training sample pool and use it to acquire $\mathbf{A}^{11}$, $\mathbf{A}^{01}$, $\mathbf{D}$ and $\{\epsilon_i\}_{i=1}^{M}$. Before that, we first introduce the failure cascade assessment oracle we apply in this work.

A. Cascade Failure Assessment Oracle

Many cascading outage and blackout assessment oracles have been proposed, for example, OPA [32], [33], improved OPA [34], Practice [35], CFS [17], etc. A comprehensive comparison of different oracles can be found in [36]. However, there is no standard oracle for assessing failure cascades at present [36]. In this paper, we follow the CFS oracle proposed in [17]. The CFS oracle is similar to the short-term OPA except that the line outages are treated deterministically, and the optimal re-dispatch is assumed not applicable during a fast failure cascade. Only urgent load shedding and generation curtailment are available if system-wide power mismatch occurs. This situation can happen in the real world. For example, on Sep 8th, 2011 the San Diego blackout experienced dozens of transmission lines triggered off within 11 minutes, leaving insufficient time for optimal re-dispatch [9]. Another practical consideration for following the CFS oracle is its relatively fast evaluation speed compared to optimization-based re-dispatch. in generating cascade sequences, including system failure check and proportional load shedding [5], [34]. Thus, in this paper we rely the CFS oracle to generate our training pool, and will consider more factors as important issues in future work, including the optimal flow redispatch [34], voltage and frequency instability [36], etc.

In CFS after each failure event, the DC/AC power flow problem is solved using the MATPOWER Toolbox [37]. A detailed oracle of generating synthetic failure cascades is presented in Appendix. By 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 $s^k := \{s^k[t]\}_{t=0}^{T_k}$ where T_k denotes the final time step that the k-th cascade terminates; $s^k[t]$ is the network state at time t; and s^k records the k-th cascade sequence.

The influence model can be directly learned from regression by a constrained optimization problem as follows.

$$\begin{aligned}\min_{\mathbf{D},\mathbf{A}^{11},\mathbf{A}^{01}} \quad & \frac{1}{K}\sum_{k=1}^{K}\sum_{t=1}^{T_k} f(s^k[t],\tilde{s}^k[t]) \\ s.t. \quad & \tilde{s}_i^k[t+1] = \sum_{j=1}^{M} d_{ij}\left(\mathbf{A}_{ji}^{11}s_j^k[t] + \mathbf{A}_{ji}^{01}(1-s_j^k[t])\right), \forall i,k; \\ & \sum_{j=1}^{M} d_{ij} = 1, \forall i; \quad d_{ij}, \mathbf{A}_{ji}^{11}, \mathbf{A}_{ji}^{01} \geq 0, \forall i,j,\end{aligned} \tag{5}$$

where $\tilde{s}^k[t]$ is the estimated value of the network state $s^k[t]$; $f(s^k[t],\tilde{s}^k[t])$ is the cost function that quantifies the distance between $s^k[t]$ and $\tilde{s}^k[t]$; M is the number of links, and K is the number of cascade sequences in $\mathcal{S}_{train}$.

The problem size of (5) is very large because for each link pair (i,j) there exist two independent pairwise influence values $\mathbf{A}_{ji}^{11}$, $\mathbf{A}_{ji}^{01}$ and a weight d_{ij}, thus the problem size is $O(M^2)$. In order to improve the computational efficiency, we train $\{\mathbf{A}^{11}, \mathbf{A}^{01}\}$ and $\mathbf{D}$ separately.

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

We use a Monte Carlo based method to learn $\mathbf{A}^{11}$ and $\mathbf{A}^{01}$. 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

$$\mathbf{A}_{ji}^{11} := \frac{\sum_{k=1}^{K} C_{ji}^{11}(s^k,\tau_i^k)}{\sum_{k=1}^{K} C_j^{1}(s^k,\tau_i^k)} \tag{6}$$

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

$$\mathbf{A}_{ji}^{01} := \frac{\sum_{k=1}^{K} C_{ji}^{01}(s^k,\tau_i^k)}{\sum_{k=1}^{K} C_j^{0}(s^k,\tau_i^k)} \tag{7}$$

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 [29].

We take the following toy example to gain a more intuitive view of (6) and (7). 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 (6) and (7),

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

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

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

Once $\{\mathbf{A}^{11}, \mathbf{A}^{01}\}$ have been obtained, their values can be substituted into (5) 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 the following convex quadratic programming.

$$\begin{aligned} \min_{\mathbf{D}} \quad & \frac{1}{K}\sum_{k=1}^{K}\sum_{t=1}^{T_k}\sum_{i=1}^{M}\Bigg(s_i^k[t+1] \\ & \qquad - \sum_{j=1}^{M} d_{ij}(\mathbf{A}_{ji}^{11} s_j^k[t] + \mathbf{A}_{ji}^{01}(1 - s_j^k[t]))\Bigg)^2 \qquad (9) \\ s.t. \quad & \sum_{j=1}^{M} d_{ij} = 1, \forall i; \qquad d_{ij} \geq 0, \forall i, j. \end{aligned}$$

Note that 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 can solve these small optimization problems in parallel in practice to further reduce the training time.

D. Learning Bisection Thresholds $\{\epsilon_i\}_{i=1}^M$

The threshold value ϵ_i should be carefully selected; otherwise it can easily induce large prediction errors. 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 (4) the state of link 2 will be assigned to 1 instead of 0, misidentifying the failure. To tackle this, we propose a scheme to estimate the thresholds value $\{\epsilon_i\}_{i=1}^M$ adaptively according to different initial contingencies. We summarize our adaptive threshold selection scheme for a specific link i as follows.

Step 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.

Step 2: Forming the threshold pool of link i from $\mathcal{S}_{train}$. 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.

Step 3: Selecting the 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[†]

$$k^* = \arg\min_{k=1,2,\cdots,K} ||s^{new}[0] - s^k[0]||_1 \qquad (10)$$

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

$$\hat{\epsilon}_i = \epsilon_i^{k^*}. \qquad (11)$$

Sometimes multiple solutions for k^* exist for (10). We choose $\hat{\epsilon}_i$ to be the median value among multiple options.

$$\hat{\epsilon}_i = \text{median}\{\epsilon_i^k\}_{k \in K^*} \qquad (12)$$

where K^* is the optimal solution set of (10).

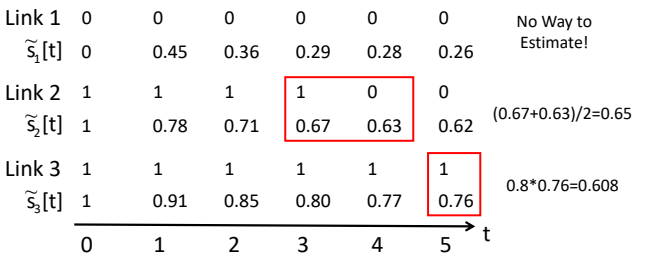


Fig. 2: This is an example of determining thresholds on all 3 link categories in a cascade sample. Each link representative has two rows of records: the first row denotes the real state value at each time step, while the second row denotes the iterative values of $\tilde{s}_i[t]$ for every link i based 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 $(0,1)$.

E. Overall Procedure

The overall procedure of learning the influence model and using it for failure cascade predictions is presented in Algorithm 1. Algorithm 1 consists of two parts: Learning Influence Model Parameters (Step 1-5); Failure Cascade Prediction (Step 6-11). In the learning part, step 1 uses the Monte-Carlo estimation (6) and (7) to obtain the pairwise influence matrices; step 2 learns the weighted matrix $\mathbf{D}$ by the convex quadratic optimization (9); step 3-4 follow the procedure in Section III-D to build the threshold pool from the training set and step 5 selects the threshold value for each link based on the given new initial contingency represented by $s^{new}[0]$. In the prediction part, step 7 to step 10 demonstrate that given the new initial contingency, the prediction iteratively calculates the intermediate state value $\tilde{s}$ based on (1) in step 8, predicts the state of any link based on the selected threshold value in step 9, and updates the predicted state and initiates

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

the prediction over the next generation in step 10, until no new links predicted to be in a failed state.

The time complexity of the training framework is dominated by the process of convex optimization (9) to learn $\mathbf{D}$. We explain this under a single training sample as follows. In the learning module, the Monte-Carlo learning of $\mathbf{A}^{11}$ and $\mathbf{A}^{01}$ has time complexity $O(M^2T)$ where M is the number of links and T denotes the largest length of failure sequence in the training sample pool. This is because both pairwise influence matrices contain M^2 elements, where each element represents the influence from one link to another, and estimating each element takes $O(T)$ as we need to traverse the state evolution of the involved link pair as the example in Section III-B shows. The convex optimization (9) can be solved by the conjugate gradient method with time complexity $O(M^4)$, since the size of variables $n = O(M^2)$ and thus the time complexity of each iteration is $O(M^2)$. In addition, the conjugate gradient method produces the exact solution after a finite number of iterations, which is not larger than the size $n = O(M^2)$ [38]. The threshold estimation has time complexity $O(MT)$ as it requires to traverse the state transition, with length $O(T)$, of one of the M links to determine its threshold each training sample. Fortunately, all these three operations can be implemented in parallel to enhance the time cost in practice.

The prediction module costs much less, since it only requires multiplication and addition manipulations. Such time efficiency in prediction compared with flow calculation will be numerically verified later in Section V.

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

Input: Training Sample Pool $\mathcal{S}_{train} = \{s^k[t]\}_{t=0,\dots,T_k}^{k=1,\dots,K}$; 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 (6) and (7) respectively;
Learn $\mathbf{D}$ from the quadratic optimization (9);
Build the threshold set Ω_i for each link i;
Find k^* based on equation (10) and form the set K^* containing all k^*s;
Obtain $\hat{\epsilon}_i$ by equation (12) 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 $\tilde{s}_i^{new}[t+1] < \hat{\epsilon}_i$ or $\hat{s}_i^{new}[t] = 0$;
  $\hat{s}_i^{new}[t+1] \leftarrow 1$ otherwise;
  $s^{new}[t+1] \leftarrow \tilde{s}^{new}[t+1]$, $t \leftarrow t+1$;
$T \leftarrow t-1$;
return $\mathbf{D}$, $\mathbf{A}^{11}$, $\mathbf{A}^{01}$, $\hat{s}^{new}$.

IV. Performance Metrics

To evaluate prediction performance via the proposed learning framework, we consider two sets of metrics: *sample-based metrics* and *link-based metrics*. Sample-based metrics capture the prediction performance for each cascade sequence samples in the test set, while link-based metrics focus on the prediction on any specific link we are interested in, for example the links connected to crucial infrastructures.

A. 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 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_{size}^k, $|A_{size}^k \cup C_{size}^k|$ is the real failure size while $|B_{size}^k \cup C_{size}^k|$ is the predicted failure size in s_{test}^k, and
$$l_{size}^k = \frac{\big||B_{size}^k \cup C_{size}^k| - |A_{size}^k \cup C_{size}^k|\big|}{|A_{size}^k \cup C_{size}^k|} \times 100\%.$$
- For l_f^k, $|A_f^k \cup B_f^k|$ is the number of effective links whose final states we mistakenly predict in s_{test}^k, while $|A_f^k \cup B_f^k \cup C_f^k \cup D_f^k|$ denotes the number of effective links, and
$$l_f^k = \frac{|A_f^k \cup B_f^k|}{|A_f^k \cup B_f^k \cup C_f^k \cup D_f^k|} \times 100\%.$$

For l_t^k, 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[¶].

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

B. Link-based Metrics

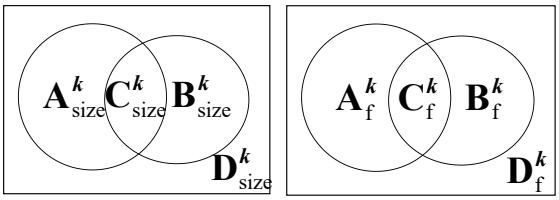
We consider four link-based metrics, and use an example to explain them. Consider the final state of link i in 8 different cascade sequences, say, $[0,1,1,0,0,1,0,1]$, where it fails initially for the 4th cascade sequence only. Assume that our prediction of its final states is $[1,1,0,0,0,0,1,0]$.

Link i Final State Prediction Error Rate $l_{f,i}$: $l_{f,i}$ denotes the ratio of incorrect final state prediction for link i among all test samples except for samples that link i trips initially. In the example, $l_{f,i} = 5/7 \approx 71\%$.

[‡]For more explanations of these metrics, please refer to [29].

[§]In this paper, the failure time is discretized as time steps. For example, a link fails at time step 3 while we predict it to be time step 5, then the failure time prediction error is $5-3=2$. The adjacent two time steps can represent some time interval length (e.g. 10 seconds or 5 minutes) in real systems.

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



	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

$$I_t^k = \frac{1}{6}\big(|4-2|+|3-5|+|5-5|+|4-1|+|2-5|+|1-3|\big) = 2$$

Fig. 3: An illustrative example of all metrics

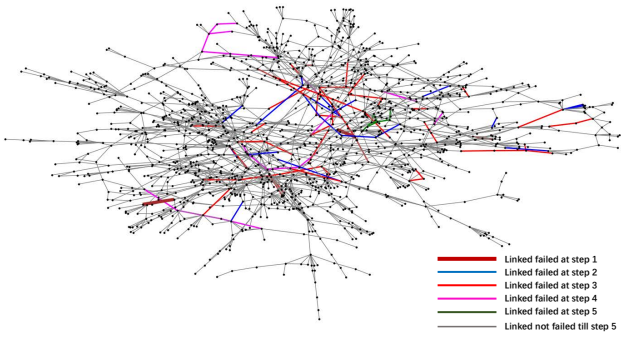


Fig. 4: A cascade sequence generated by Algorithm 1

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 samples that link i trips initially. It reflects the ratio of failure misidentification. In the example, $l_{md,i} = 2/3 \approx 67\%$.

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. In the example, $l_{fa,i} = 3/4 = 75\%$.

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 in 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 with different failure frequencies. The failure frequency of link i is defined as the ratio of link i experiencing failure during the cascade process among the training sample pool. For example, if link i fails during the cascade process in 200 cascade sequences among the training sample pool of 1000 cascade sequences, then the failure frequency of link i is $200/1000 = 20\%$. For the effect of failure frequency on prediction accuracy, it is intuitive that a link that rarely fails can induce a large $l_{md,i}$. Therefore, evaluating the prediction performance on links with different failure frequencies can deepen our understanding of the model and features in a very detailed manner.

V. Simulation Results

In this section, we conduct a numerical study of the proposed method for failure cascade prediction on both the DC flow and AC flow model, under the two sets of metrics.

A. Dataset Information

We apply the standard datasets in Matlab MATPOWER toolbox [37]. We choose three systems as examples: 1354-bus, 2383-bus, 3012-bus. We use the provided transmission line and generator values, and set capacity values which are not given to be large enough such that the corresponding transmission lines are free from cascading failure.

In our simulation we will vary the loading conditions, hence for convenience we define as the power generation and loading given by the dataset as the *default setting* for each system. Table I presents the information for cascade samples in 3 systems under DC and AC flow and their default settings, where *#Eff. Rate* is the portion of effective links, which have ever experienced failure induced by the cascade process in $\mathcal{S}_{train}$. The data is calculated from $50,000$ cascade samples from the total of $N-2$ initial contingencies.

In our experiment, we consider at most $50,000$ cascade sequence samples for training in each system. Note that the ratio of the training samples compared with the set of all possible $N-2$ initial contingencies is very low, for example in 1354-bus system, the ratio is $50,000/\binom{1710}{2} = 3.4\%$, and this ratio is even lower in larger systems.

TABLE I: Sample Information under DC and AC Flow

System Size	1354	2383	3012
Flow Model	**DC| AC**	**DC| AC**	**DC**
#Generators	260 | 260	327 | 327	297
#Links	1710 | 1710	2886 | 2886	3566
#Eff. Links	762 | 1547	2088 | 2815	2083
Eff. Rate	45% | 91%	72% | 98%	58%
Avg. Fail Size	179 | 80.3	598 | 390	263
Max Fail Size	314 | 1292	862 | 2606	792
Min Fail Size	2 | 2	110 | 2	11

A typical failure cascade sequence generated by Algorithm 1 fundamentally based on the influence model (1) is shown in Fig. 4, which represents the 1354-bus under the AC system with $1.5\times$ default loading. Two lines are initially tripped (bold brown lines) and the failure cascades propagate to a broader regime. These failed lines are generally with low capacity and high initial flow levels.

B. Performance Evaluation on DC Systems

In this part, we present the evaluation results under DC flow model. We only present the results on link-based metrics, where evaluations of sample-based metrics can be found in [29]. For the link-based metrics, based on different ranges of failure frequencies, effective links are grouped into 10 categories, ranging from $(0, 10\%]$ to $(90\%, 100\%]$.

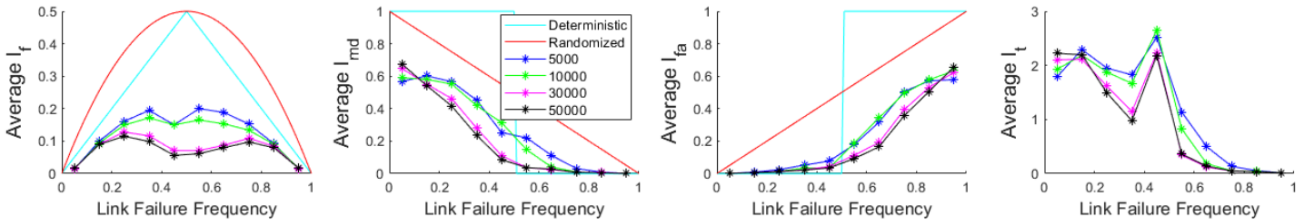


Fig. 5: Results on each link for 2383-bus DC system with $1.5\times$ default loading

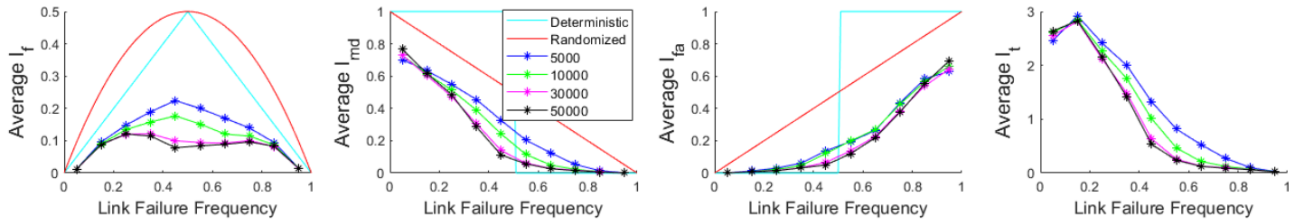


Fig. 6: Results on each link for 3012-bus DC system with $1.5\times$ default loading

Fig. 5 and 6 present the results on link-based metrics for the 2383-bus and 3012-bus systems under the DC flow model. We only present one loading condition for each system. Other loading conditions lead to results with similar pattern. For each system, the four subfigures represent the evaluation of $l_{f,i}$, $l_{md,i}$, $l_{fa,i}$ and $l_{t,i}$. For all the metrics, we calculate the average metric value among all the links in each failure frequency interval, marked as asterisks in each subfigure.

Different sizes of the training sample pools are considered, from 5,000 to 50,000, as shown in Fig. 5 by different colors. As the sample pool increases, the prediction accuracy is greatly improved. To demonstrate that our proposed framework scheme captures the failure cascade features, we compare it to two prediction methods that do not depend on the influence model: the deterministic prediction and the randomized prediction. The deterministic prediction is to 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%. The randomized prediction is to predict a link to be failed with probability equal to its failure frequency. These two methods are to some extent naive since they determine the final state of each link purely based on the link failure frequency. We now explain the results in Fig. 5 and Fig. 6.

For $l_{f,i}$, we highlight two observations. First, the prediction error based on the trained influence model is lower than the two commonly used methods among all the failure frequency intervals. Second, with increasing number of training samples, the average prediction error in each frequency interval decreases, and it decreases most among links with failure frequency around 50%, which are predicted with maximum deviation by the two naive methods.

For $l_{md,i}$ and $l_{fa,i}$, they are "conjugate" metrics. By comparing the two common methods with our prediction mechanism, we can observe that the prediction based on the influence model induces lower miss detection rate than the randomized method. Meanwhile, links with higher failure frequency leads to generally lower miss detection rate. This is because such links have more failure records in the training set, which instructs the model to predict these links to be failed. For $l_{fa,i}$, we have similar analysis but opposite results, where links fail with high frequency generally lead to large false alarm rate. Note that the deterministic prediction performs 0% miss detection rate for links with failure frequency $[0.5, 1]$. However, for such links the deterministic prediction performs 100% false alarm rate, as it always predicts links with failure frequency in $[0.5, 1]$ to be failed in the final state. Therefore, we cannot say that the deterministic prediction performs well for links with failure frequency $[0.5, 1]$. This tells that we need to combine the miss detection rate l_{md} and the false alarm rate l_{fa} together to judge and compare the performance.

For $l_{t,i}$, we can find that the prediction error for links with higher failure frequency is generally lower. The reason is that links fail with high frequency feed more information about its failure to the model. In addition, the prediction error for links with lower failure frequency is large, as they tend to be predicted to be normal, and whenever it happens, the time step we predict is the termination time, and thus the error may be very large if the truth is that this links fails very early.

Generally, the influence model trained in our proposed scheme can overall achieve a better prediction performance for each category of the links than the common prediction methods, especially for those link categories that the common methods will result in the maximum error.

C. Performance Evaluation on AC Systems

The evaluation over AC systems is similar to that in DC systems. We consider both sample-based and link-based metrics. We present the result over two systems: 1354-bus and 2383-bus systems, and in each system we consider two loading conditions: $1.5\times$ and $2\times$ default loading for 1354-bus system; $1\times$ and $1.25\times$ default loading for 2383-bus system.

1) Sample-based metrics: Fig. 7 and 8 show the result of the three sample-based metrics in the 1354-bus, and 2383-bus system. The left column in each figure illustrates the metric

values with increasing training samples. In the 1354-bus system, we only consider at most 5,000 training samples because the cascade pattern under the AC flow model in this system is not complex so that 5,000 samples are enough to reach a high quality prediction performance. In the 2383-bus system, 30,000 samples are adequate to yield a high quality prediction performance. In both systems, we can observe that with more training samples, the decreasing trend is generally over all the metrics, and the failure size prediction error rate can be reduced below 4%, the final state prediction error rate can be down to 10%, and the time prediction error is lower than 1.25 time step units.

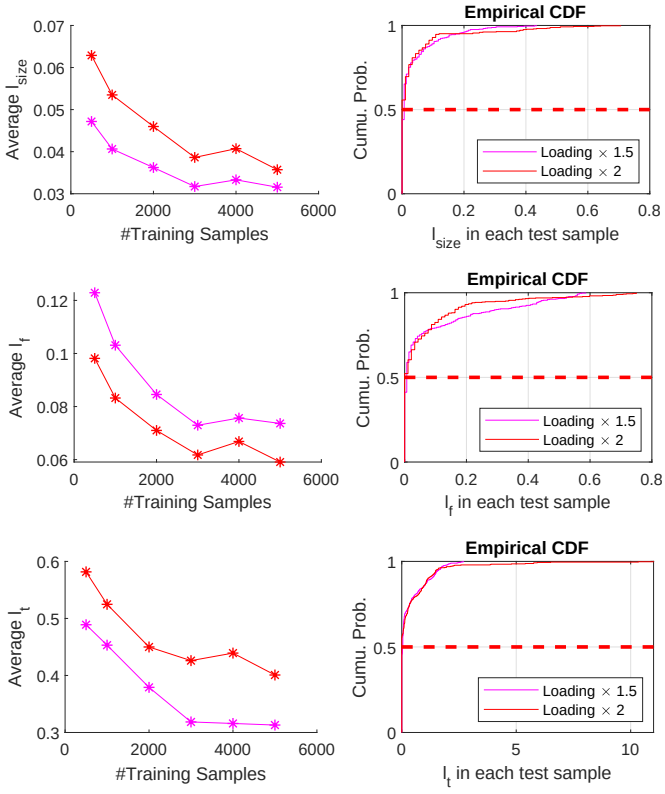


Fig. 7: Results on l_{size}, l_f, l_t in order for 1354-bus AC system

The right column in each figure shows the cumulative distribution function (CDF) of each metric when the largest number of training samples are used in each setting. The result informs us that most of the test samples can be predicted within reasonable range. For the 1354-bus system, more than 90% of the test samples can have $< 10\%$ failure size error rate; more than 70% can have $< 5\%$ final state error rate; and more than 80% can have time prediction error within 1 step. Such result is similar in the 2383-bus system.

2) Link-based metrics: Fig. 9 and 10 present the results over link-based metrics in the 1354-bus and 2383-bus systems. Similar to the trend in the DC flow cases, we find that prediction based on the trained influence model is more accurate than the deterministic and randomized methods, and more training samples will reduce the error rate among all these four metrics, especially for links whose failure frequency is around 50% generally.

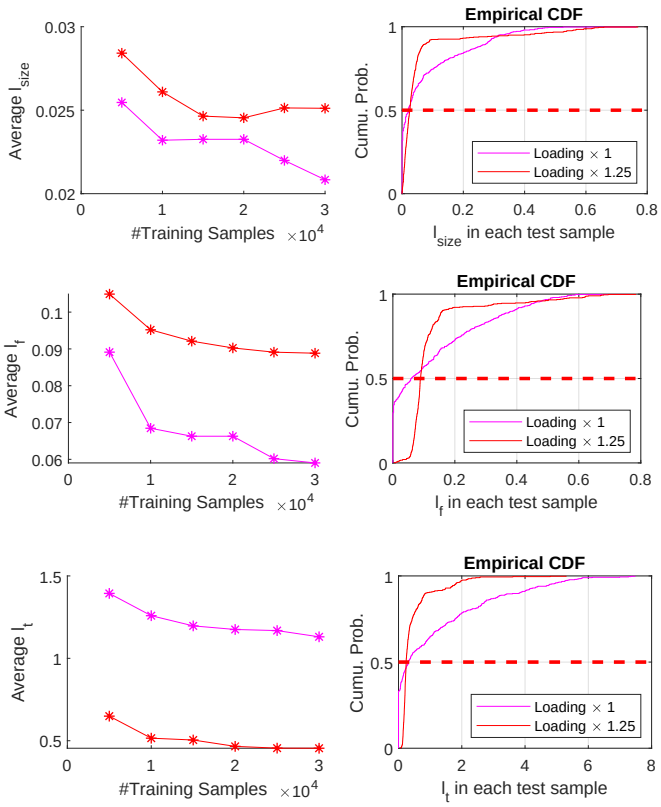


Fig. 8: Results on l_{size}, l_f, l_t in order for 2383-bus AC system

Two additional issues should be noted. (i) No links have failure frequency between 20% and 60% in 1354-bus system under AC flow. (ii) The miss detection rate for links that fail with low frequency may be even larger under more training samples, because for these links, more training samples may 'dilute' the cases of the cascading failure.

D. Comparison between Prediction Performance DC and AC

We have presented the prediction results over DC and AC flow models respectively. In this part we compare DC and AC under the same network settings. The results for 1354-bus and 2383-bus systems are summarized in Table II. We can observe that with enough training samples, generally the prediction performance in AC power systems slightly outperforms that in DC power systems. A few potential reasons may contribute to this difference. First, the nonlinearity of the AC model may render clustering of failure spots, leading to stronger patterns than that of the DC model. Second, load sheddings and generation curtailments in the AC model may be more aggressive than that in the DC model because the existence of AC power flow solutions is not guaranteed after an initial small load shedding. Third, the threshold values in determining link failures may be selected differently, causing different predicting results. However, the exact reason is still elusive, and can be one of the future research directions.

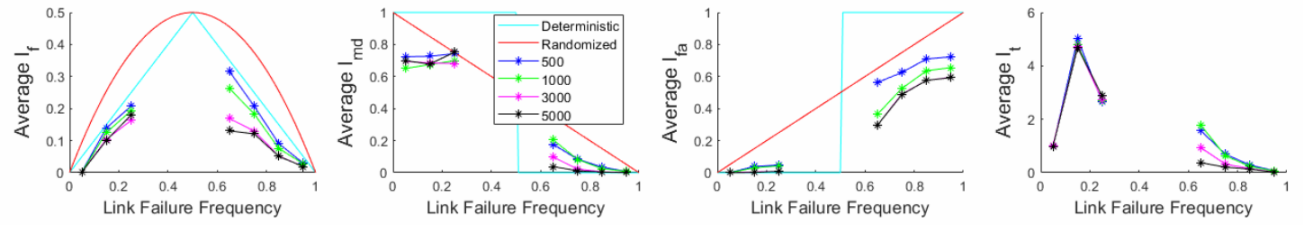


Fig. 9: Results on each link for 1354-bus AC system with $2\times$ default loading

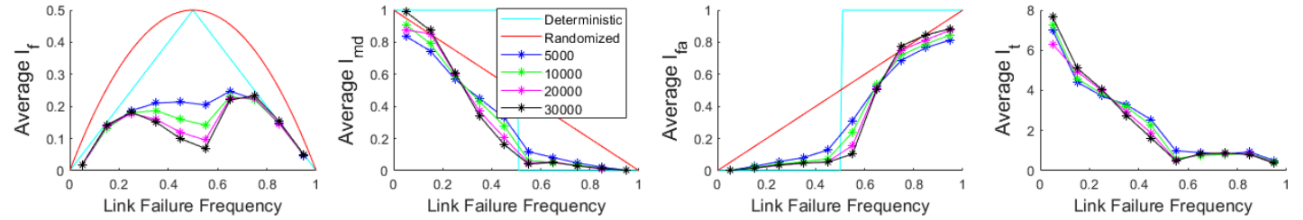


Fig. 10: Results on each link for 2383-bus AC system with $1\times$ default loading

TABLE II: Prediction Performance on DC and AC Systems

Metric	IEEE1354 (Loading $\times 1.5$)		IEEE2383 (Loading $\times 1$)	
	DC	AC	DC	AC
l_{size}	3.9%	3.2%	3.1%	2.1%
l_f	9.6%	7.7%	7.4%	6.0%
l_t	0.91	0.32	0.81	1.2

E. Computational Efficiency

Next, we demonstrate the superiority of the influence model based prediction framework to the power flow based contingency calculation by CFS in computational time cost reduction under both DC and AC flow models. We test both methods in MATLAB 2019a on Intel(R) Core(TM) i9-7920X CPU@2.90GHz Processor. In each setting, we run $1,000$ test samples.

We summarize the results in Table III, where T_{flow} denotes the computational time (in seconds) of the power flow based method, T_{IM} denotes the computational time of the influence model based prediction method, and $r = \frac{T_{flow}}{T_{IM}}$ is the improvement ratio. Results show that our method works better in larger systems and under higher loading conditions that tend to cause large failures. The result also suggests that the proposed method reaches a large improvement ratio on the AC systems because it avoids solving nonlinear equations.

TABLE III: Prediction Time Cost on 1,000 Samples

System	Low Load			High Load		
	T_{flow}	T_{IM}	r	T_{flow}	T_{IM}	r
2383DC	2597	43.3	**60**	3603	34.7	**104**
3012DC	3891	59.4	**66**	5864	46.9	**125**
1354AC	5280	24.9	**212**	4950	23.5	**211**
2383AC	14498	34.4	**421**	29197	33.5	**872**

F. Structures of System-Wide Influence

Two interesting system-wide influence structures are observed here. Firstly, the relative influence matrix $\mathbf{D}$ has a sparse structure. For example, Fig. 11 shows the structure of the $\mathbf{D}$ matrix for the 2383-bus system for both DC and AC models. We define an influence to be visible from link i to j if $d_{ji} > 0.01$. Then, the average numbers of visible links on each link is 4.7 and 3.1 under the DC model and the AC model, respectively. Such sparsity property of $\mathbf{D}$ promises lower time cost and higher scalability of learning $\mathbf{D}$ in Section III-B. Another observation in Fig. 11 is that in the DC model the matrix $\mathbf{D}$ has a rough symmetric structure, indicating a mutual influence between each pair of links. However, in the AC model, most visible links are structured in columns, suggesting a uni-directional influence from one link to other links.

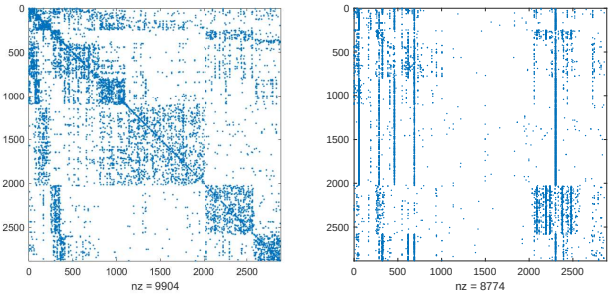


Fig. 11: Elements of $\mathbf{D}$ larger than 0.01 in 2383-bus system under DC (left) and AC (right) flow model

VI. Critical Initial Contingency Identification

In this section we propose a scheme to identify the critical initial contingencies based on the influence model.

Intuitively based on (1), a link whose failure has enormous negative effects on the whole system should greatly reduce the intermediate state values $\tilde{s}[t]$ of other links during the cascade, as this will drag the value $\tilde{s}_i[t]$ down below the threshold values ϵ_i for each link i with higher probability. Therefore, a

natural idea to identify the links that may cause large failure size, is to find the link j^* such that

$$j^* = \arg\max_j \sum_{i=1}^{M} \tau_{ij} = \arg\max_j \sum_{i=1}^{M} d_{ij}(\mathbf{A}_{ji}^{11} - \mathbf{A}_{ji}^{01}). \quad (13)$$

where $\tau_{ij} \triangleq d_{ij}(\mathbf{A}_{ji}^{11} - \mathbf{A}_{ji}^{01})$ represents the integrated influence from link j to i based on (1), as it combines both pairwise influences and the weights. When $s_j[t]$ turns from 1 to 0, the value of $\tilde{s}_i[t+1]$ is decreased by τ_{ij}. We define the sum $\sum_{i=1}^{M} \tau_{ij}$ as the *total influence* of link j on the whole system. Intuitively, the link j^* with highest total influence value is the link whose failure is most prone to cause large failures. We can find the top-L links $\{j_l\}_{l=1}^{L}$ that maximize $\sum_{i=1}^{M} \tau_{ij}$. Any $N-k$ initial contingency among these L links has a better chance to lead to a large failure size.

Based on this idea, we examine $N-2$ initial contingencies on the 1354-bus and 2383-bus systems under both DC and AC power flow models. We sort the total influence value $\sum_{i=1}^{M} \tau_{ij}$ for each link j in an increasing order, as shown in Fig. 12 and choose the 10 links with largest total influence values, to serve as our critical link pool.

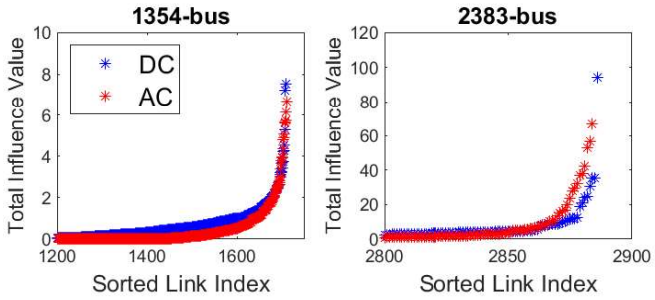


Fig. 12: Link indexes sorted based on total influence values for 1354-bus and 2383-bus systems

In each setting of both systems, we use the training pool $\mathcal{S}_{train}$ to obtain the empirical distribution of the failure size, which is denoted as p_0. In the critical link pool, we enumerate all the combinations of $N-2$ contingencies, and find the empirical failure size distribution p_{max}. It involves $\binom{10}{2}$ many initial contingencies. Comparisons of p_0 and p_{max} are presented in Fig. 13 to 15. Besides, we plot our prediction of failure size distribution in each figure as a supplement of the performance evaluation metrics in Section IV-A. Results show that our prediction can capture the pattern of the distribution accurately in general, while it takes a more concentrated form whose peak value is greater than the failure size distribution obtained by the flow calculation.

Fig. 13 presents the results for both systems under DC flow model. The blue bars represent p_0, while the orange bars represent p_{max}. We can observe that in both cases the distribution of p_{max} has a heavy tail which tends to concentrate on large failure size cases. p_{max} contains larger failure sizes with higher frequency, which indicates that the selection of the top-10 influential links by (13) tends to induce larger failure sizes. Failure cascades with small sizes are rarely observed from the contingencies selected in the top-10 critical links for the DC model.

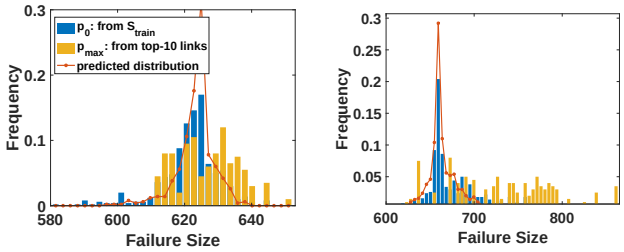


Fig. 13: $N-2$ contingency failure size distribution of $\mathcal{S}_{train}$ and top-10 influential links in DC systems (Left: 1354-bus, $2\times$ default loading; Right: 2383-bus $1.25\times$ default loading)

Fig. 14 and Fig. 15 present the results over AC flow case with different loading conditions for 1354-bus and 2383-bus systems respectively. In both systems, the $N-2$ contingencies selected from the top-10 critical links are more likely to induce larger failure sizes than the contingencies randomly selected from all the links.

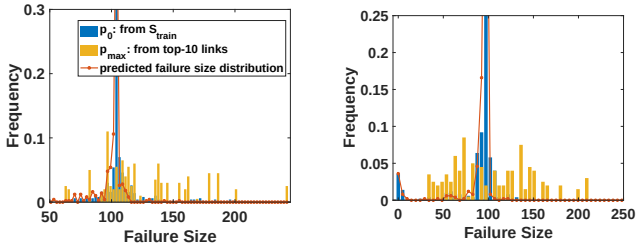


Fig. 14: $N-2$ contingency failure size distribution of $\mathcal{S}_{train}$ and top-10 influential links in 1354-bus AC system (Left: $1.5\times$ default loading; Right: $2\times$ default loading)

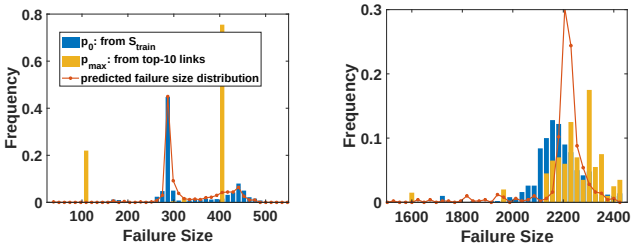


Fig. 15: $N-2$ contingency failure size distribution of $\mathcal{S}_{train}$ and top-10 influential links in 2383-bus AC system (Left: $1\times$ default loading; Right: $1.25\times$ default loading)

Table IV presents the results in a more quantified manner. For each of the six settings discussed above, we calculate the ratio of $N-2$ initial contingencies in the critical link pool whose failure size is no smaller than the median ($1/2$-quantile) value and $3/4$-quantile value of the failure size in $\mathcal{S}_{train}$. Such ratios are denoted as $\gamma_{1/2}$ and $\gamma_{3/4}$ respectively. The reference values are $\gamma_{1/2} = 50\%$ and $\gamma_{3/4} = 25\%$. We can observe that $\gamma_{1/2} > 50\%$ for all the settings, and for $\gamma_{3/4}$, it largely surpasses 25% in all the cases except '2383AC ($1\times$)'. In fact,

TABLE IV: $\gamma_{1/2}$ and $\gamma_{3/4}$ of all the six settings under our mechanism

Ratio	$\mathcal{S}_{train}$	Critical Link Pool					
	All Systems	1354DC (2×)	1354AC (1.5×)	1354AC (2×)	2383DC (1.25×)	2383AC (1×)	2383AC (1.25×)
$\gamma_{1/2}$	50%	68%	71.5%	55%	87.5%	78%	73.5%
$\gamma_{3/4}$	25%	57%	66%	55%	67.5%	0%	60.5%

for this setting, shown in Fig. 15, we can show that its $\gamma_{2/3} = 75.5\%$. The results verify that our mechanism works in broad settings as well. In addition, the identification of critical initial contingencies is purely based on the influence model trained by our framework. Its time complexity is $O(M^2)$ and can be implemented in parallel, which makes it favorable for large scale system screenings compared with the methods in [3], [26] which involves computing the matrix inverse with respect to the link flow, whose time complexity is at least $O(M^{2.373})$ by the most efficient algorithm till now [39].

VII. Conclusion

In this paper, we study the failure cascades in power grids by using the influence model. It is a flow-free approach that can explicitly characterize system-wide influences on each component. This influence information can be used for efficient predictions of failure cascades and assist in identifying the most critical initial contingencies.

To acquire the influence model of power grids, we proposed a hybrid learning method which combines the Monte Carlo method, a regression method, and an adaptive selection scheme. It has polynomial time complexity, and each module can be implemented separately in parallel. We applied this method to obtain influence models for both the DC and the AC large-scale power system examples. Four metrics of prediction performances in different levels of granularity have been proposed and tested on these examples. Numerical results showed that the influence model learned from our proposed method can achieve a good prediction accuracy in all the metrics for both the DC model and the AC model. A comparison between the influence model and an apposite flow-model-based approach revealed that the influence model can predict failure cascades in a much faster manner (at least two orders of magnitude faster) with acceptable small error rates. Thus, it is very suitable for screening severe contingencies in large quantities. A further investigation on the learned influence values revealed the critical information of severe contingencies, which further helped us identify the most critical initial link outages.

One of future research directions can explore the sparse structure of the weighted matrix $\mathbf{D}$. It can help further accelerate the learning procedure, and abstract important influence relationship in the system. Another important future research direction is to improve the training structure to better capture rare events. A possible approach could be using the importance sampling to enhance the weight of the events with lower occurrence frequencies. A broader application of the influence model in other network systems is also promised in the future, especially for interdependent systems such as power-communication systems, etc.

References

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

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

[3] 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.

[4] 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.

[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] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

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

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

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

[23] 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.

[24] 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.

[25] 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.

[26] K. Zhou, I. Dobson, P. D. Hines, and Z. Wang, "Can an influence graph driven by outage data determine transmission line upgrades that mitigate cascading blackouts?" in *2018 IEEE International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*. IEEE, 2018, pp. 1–6.

[27] K. Zhou, I. Dobson, Z. Wang, A. Roitershtein, and A. P. Ghosh, "A markovian influence graph formed from utility line outage data to mitigate large cascades," *IEEE Transactions on Power Systems*, vol. 35, no. 4, pp. 3224–3235, 2020.

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

[29] X. Wu, D. Wu, and E. Modiano, "An influence model approach to failure cascade prediction in large scale power systems," in *2020 American Control Conference (ACC)*. IEEE, 2020, pp. 4981–4988.

[30] J. Qi, K. Sun, and S. Mei, "An interaction model for simulation and mitigation of cascading failures," *IEEE Transactions on Power Systems*, vol. 30, no. 2, pp. 804–819, 2014.

[31] C. Chen, W. Ju, K. Sun, and S. Ma, "Mitigation of cascading outages using a dynamic interaction graph-based optimal power flow model," *IEEE Access*, vol. 7, pp. 168 637–168 648, 2019.

[32] I. Dobson, B. Carreras, V. Lynch, and D. Newman, "An initial model for complex dynamics in electric power system blackouts," in *Proceedings of the 34th annual Hawaii international conference on system sciences*, vol. 2. Citeseer, 2001, pp. 2017–2017.

[33] S. Mei, Y. Ni, G. Wang, and S. Wu, "A study of self-organized criticality of power system under cascading failures based on ac-opf with voltage stability margin," *IEEE Transactions on Power Systems*, vol. 23, no. 4, pp. 1719–1726, 2008.

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

[35] E. Ciapessoni, D. Cirio, and A. Pitto, "Cascading simulation techniques in europe: The practice experience," in *IEEE PES General Meeting*, 2013, pp. 20–25.

[36] P. Henneaux, E. Ciapessoni, D. Cirio, E. Cotilla-Sanchez, R. Diao, I. Dobson, A. Gaikwad, S. Miller, M. Papic, A. Pitto *et al.*, "Benchmarking quasi-steady state cascading outage analysis methodologies," in *2018 IEEE International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*. IEEE, 2018, pp. 1–6.

[37] R. D. Zimmerman, C. E. Murillo-Sánchez, and R. J. Thomas, "Matpower: Steady-state operations, planning, and analysis tools for power systems research and education," *IEEE Transactions on power systems*, vol. 26, no. 1, pp. 12–19, 2010.

[38] W. Sun and Y.-X. Yuan, *Optimization theory and methods: nonlinear programming*. Springer Science & Business Media, 2006, vol. 1.

[39] F. Le Gall, "Powers of tensors and fast matrix multiplication," in *Proceedings of the 39th international symposium on symbolic and algebraic computation*, 2014, pp. 296–303.

[40] C. Grigg, P. Wong, P. Albrecht, R. Allan, M. Bhavaraju, R. Billinton, Q. Chen, C. Fong, S. Haddad, S. Kuruganty *et al.*, "The ieee reliability test system-1996. a report prepared by the reliability test system task force of the application of probability methods subcommittee," *IEEE Transactions on power systems*, vol. 14, no. 3, pp. 1010–1020, 1999.

APPENDIX

A. Oracle of Cascade Sequence Generation

The CFS oracle [17] is summarized below.

1) Given the loading, compute the initial link flows.
2) Randomly initiate an $N-k$ contingency (k initially failed links) where $k=2$ in our setting.
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) For each island, specify a bus as the slack bus.
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 them and return to Step 3). Otherwise, terminate.

We use Newton-Raphson method to solve nonlinear equations in AC power model in each island during cascade. However, if it does not converge, we proportionally decrease the power injection and loading until convergence [5].

B. Test Results on the RTS-96 System

In [36], different oracles for failure cascade are compared under the standard RTS-96 3-area system model [40], consisting of 73 buses and 120 transmission lines. Here we validate the CFS oracle used in this paper to the existing results presented in [36]. Fig. 16 illustrates the log-log plot of the distribution of the number of outages among all the $N-2$ initial contingencies of transmission lines. It is closely matched to the DC OPA and AC OPA curves in Fig. 4 of [36].

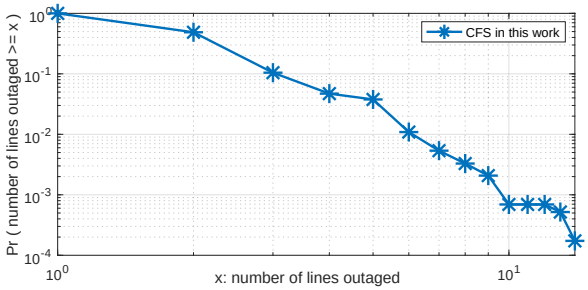
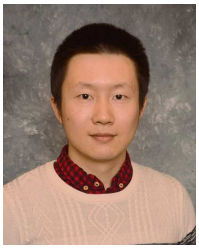


Fig. 16: Distribution of number of outaged lines of the RTS-96 system under the CFS oracle in this work (log-log scale)

C. Brief Summary of the Settings of Tested Systems

All the data of three systems come from the MATPOWER Toolbox [37]: (1) 1354-bus system (file: case1354pegase.m); (2) 2383-bus system (file: case2383wp.m); (3) 3012-bus system (file: case3012wp.m). The transmission line flow capacity is defined as the `rateA` term in the data files. The given generator capacities, power generation and load values are regarded as the default values in our setting.

Xinyu Wu (S'18) received the B.S. degree in electrical engineering from Shanghai Jiao Tong University, Shanghai, China, in 2018, and the M.S. degree from Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, in 2020. Currently he is pursuing the Ph.D. degree from the Department of Aeronautics and Astronautics at the Laboratory for Information and Decision Systems in MIT. His research includes power system analysis, communication networks, and interdependent complex systems.



Dan Wu (S'15, M'17) received the B.S. degree in electrical engineering and automation from the Huazhong University of Science and Technology, Wuhan, China, in 2012. He received the M.S. and Ph.D. degrees from the University of Wisconsin Madison, Madison, WI, USA, in 2014 and 2017, respectively. He was a Postdoctoral Associate at the Mechanical Engineering Department, Massachusetts Institute of Technology (MIT) from 2017 to 2019, and now is a Postdoctoral Associate at the Laboratory for Information and Decision Systems, MIT. His research includes nonlinear dynamics and optimization, electric power system analysis, natural gas networks, interdependent complex systems, algebraic and topological methods in engineering applications.

Eytan Modiano is Professor in the Department of Aeronautics and Astronautics and Interim Director of the Laboratory for Information and Decision Systems (LIDS) at MIT. Prior to Joining the faculty at MIT in 1999, he was a Naval Research Laboratory Fellow between 1987 and 1992, a National Research Council Post Doctoral Fellow during 1992-1993, and a member of the technical staff at MIT Lincoln Laboratory between 1993 and 1999. Eytan Modiano received his B.S. degree in Electrical Engineering and Computer Science from the University of Connecticut at Storrs in 1986 and his M.S. and PhD degrees, both in Electrical Engineering, from the University of Maryland, College Park, MD, in 1989 and 1992 respectively.

His research is on modeling, analysis and design of communication networks and protocols. He received the Infocom Achievement Award (2020) for contributions to the analysis and design of cross-layer resource allocation algorithms for wireless, optical, and satellite networks. He is the co-recipient of the Infocom 2018 Best paper award, the MobiHoc 2018 best paper award, the MobiHoc 2016 best paper award, the Wiopt 2013 best paper award, and the Sigmetrics 2006 best paper award. He was the Editor-in-Chief for IEEE/ACM Transactions on Networking (2017-2020), and served as Associate Editor for IEEE Transactions on Information Theory and IEEE/ACM Transactions on Networking. He was the Technical Program co-chair for IEEE Wiopt 2006, IEEE Infocom 2007, ACM MobiHoc 2007, and DRCN 2015. He had served on the IEEE Fellows committee in 2014 and 2015, and is a Fellow of the IEEE and an Associate Fellow of the AIAA.