This article has been accepted for publication in IEEE Transactions on Smart Grid. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSG.2023.3278702

Distribution Grid Topology and Parameter Estimation Using Deep-Shallow Neural Network with Physical Consistency

Haoran Li, *Student Member, IEEE,* Yang Weng, *Senior Member, IEEE,* Vijay Vittal, *Life Fellow, IEEE*, Erik Blasch, *Fellow, IEEE*

***Abstract*—To better monitor and control distribution grids, the exact knowledge of system topology and parameters is a fundamental requirement. However, topology information is usually incomplete due to limited sensors in the grid. Therefore, estimating the system parameters using partial data is a critical topic for distribution systems. Due to the high nonlinearity of unobservable system quantities and noises, Deep Neural Networks (DNNs) are widely utilized for accurate estimation. While traditional approaches either treat DNNs as a black box or embed little physical knowledge into DNNs, they cannot guarantee that the DNN model is consistent with physical equations and hence lack accuracy and interpretability. Therefore, we propose a Deep-Shallow neural Network (DSN) for distribution system estimation. The key is to create virtual nodes to represent nodes without sensors in the system, and denote a DNN to approximate missing quantities at virtual nodes. Then, the Power Flow (PF) equations can be estimated via a shallow neural network, achieving physical consistency. Isolated by virtual nodes, the whole system is decomposed into a set of reduced graphs with approximate PF equations. Likewise, we introduce a Reinforcement Learning-based search algorithm to connect the reduced grids into one connected system. Correspondingly, the DSN is fine-tuned to achieve consistency with the PF equations of the connected graph. Finally, the paper illustrates the superiority of the proposed DSN due to its physical consistency. Specifically, comprehensive experiments demonstrate the high performances of our DSN over other methods in distribution grids.**

***Index Terms*—System identification, distribution systems, partial observability, deep neural network, reinforcement learning.**

I. Introduction

Power distribution systems are incorporating more and more Distributed Energy Resources (DERs) to achieve clean and economic energy consumption. However, DERs bring issues like bi-directional power flows [1], instantaneous system power unbalance [2], and more voltage violations [3]. To enable better monitoring and control, significant work has been proposed based on the assumption of complete system information on topology and parameters [4].

However, such knowledge is often incomplete due to limited instrumentation and low investment interests [5], e.g., in the secondary distribution networks. Secondly, distribution grid topology can experience relatively frequent topological and parameter changes. The goal of these reconfigurations is typically for achieving better operation [6]–[8], regular maintenance [9], and quick restoration after system outages [10]. Thirdly, many plug-and-play components are currently owned by individual users and therefore, their connectivity and status are not always available to the distribution grid operators. Overall, it is desirable to accurately estimate the distribution system parameters and topology for better monitoring.

To achieve this goal, existing methods can be mainly categorized into three groups: statistical test-based methods, traditional Machine Learning (ML) methods, and Deep Learning (DL) methods. The statistical test methods compute certain statistics using electrical measurements to determine the connectivity of the system. For instance, [11], [12] calculate the voltage statistics (e.g., covariance) to recover the topology. While, [1], [13], [14] estimate the edge weights via voltage and current expectations. Then, graphical model-based methods are employed to learn the topology. References [15] and [16] compute the second-order statistics of voltage and power injections to estimate grid topology. However, these works mostly focus on topology recovery but do not estimate system parameters.

To estimate system parameters, traditional ML-based methods treat the system identification problem as a regression task. Then, by minimizing the error of the model output, one can learn the model parameters that represent the system parameters and topology. The linear regression model is the most common method to learn a linear form of PF equations with certain variable transformations [9], [17]–[24]. The model is linear with respect to edge parameters given transformed input/output physical quantities. These approaches include some unrealistic assumptions regarding the grid. For example, [17]–[19] require all nodes of the system to have micro Phasor Measurement Units (micro-PMUs). References [21], [23] assume nodes without micro-PMUs have zero power injections. Reference [9] relaxes the zero-injection assumption but still requires the unmetered nodes to have low power correlations with metered nodes. Finally, [20], [22] require special perturbation techniques to actively gather data. To summarize, the incomplete placement of micro-PMUs or other meters limits the capacity of linearly expressing the power flow equation, thus impacting the performance of the above referenced methods.

Such observations are further theoretically supported in

Haoran Li, Yang Weng, and Vijay Vittal are with the Department of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ, 85281, USA. E-mail: {lhaoran,yang.weng,Vijay.Vittal}@asu.edu. Erik Blasch is with Air Force Research Laboratory, Arlington, VA, 22203, USA. E-mail: erik.blasch@gmail.com.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Smart Grid. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSG.2023.3278702

[21], where Moffat, et al. prove that it is impossible to recover the complete topology and parameters given limited phasor measurements. At the same time, it is valuable to accurately learn the system equations within the metered nodes to construct the so-called reduced grids [25]. To achieve reduced grids, the key is to capture the complex nonlinear correlations induced by missing data, measurement noises, and some plug-and-play devices with user privacy considered. Therefore, some other researchers have tried to employ nonlinear ML models like Support Vector Regression (SVR) [26], [27]. However, the ability of SVR to capture a representation is limited for distribution grids. To enhance the ability to capture the grid representation, Neural Networks (NNs) for universal approximation are primarily considered for modeling the power flow equations [28]–[30]. Above referenced works treat NNs as black boxes to learn the mapping rule. However, the black box models lead to little physical interpretability and cannot guarantee the generalizability of the learned model to different loading conditions.

Thus, in this paper, we try to provide answers to the following question: can we maximize the recovery of distribution system information while deploying the universal approximation capability in other unrecoverable areas? To recover the power flow equations of the partially observable distribution systems and maximize the recovery of the physics, the key is to decompose a DNN model into a linear part with locally full observability and a nonlinear part with a high capacity to handle the randomness from hidden quantities.

We propose a Deep-Shallow neural Network (DSN) with a deep-shallow structure to obtain maximal physical consistency. DSNs can be shrunk to physical equations for sub-areas with full observability. Under full observability, only the shallow NN in the DSN activates. Further, the DSN creates some virtual nodes to represent an unobservable area. The deep NN in the DSN activates in this situation to approximate the variables of the virtual nodes. Then, physical quantities related to the virtual nodes are constrained into a safe range and the local topology of the virtual nodes is guaranteed via controlling the sparsity of the DSN. Additionally, virtual nodes will isolate the distribution system into multiple reduced grids. To reconnect these reduced systems, we propose to utilize Reinforcement Learning (RL) to search the connection and find the most suitable one, leading to a reduced and connected grid. Correspondingly, the DSN model is fine-tuned to accurately represent the final grid with the most physical consistency. To summarize, the following contributions have been made in this paper:

- Introducing the problem of maximizing topology and parameter recovery for interpretation while minimizing approximating error in the non-recoverable regions;
- Determining the solution of the DSN to the proposed problem with a structural design and physically-constrained initialization;
- Proposing to fine-tune the DSN and produce a connected and reduced graph using RL methods;
- Providing extensive experiments on diversified physical systems against many state-of-the-art models to demonstrate the superiority of our proposed methods.

The remainder of the paper is organized as follows: Section II defines the problem. Section III proposes the DSN model and the RL-based DSN fine-tuning. Section IV conducts experiments for the baseline and DNS models, and Section VI concludes the paper.

II. Problem formulation

A distribution system can be modeled as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the node set and $\mathcal{E}$ is the branch set. Further, the node set $\mathcal{V}$ can be categorized based on the availability of sensors, e.g., micro-PMUs. Specifically, we denote $\mathcal{O}$ as the observed node set with phasor data, and $\mathcal{H}$ as the unobserved node set without such data. Basically, $\mathcal{O} \cup \mathcal{H} = \mathcal{V}$ and $\mathcal{O} \cap \mathcal{H} = \emptyset$. Our goal is to build an accurate PF mapping of the power grid using measurements of $\mathcal{O}$.

Before elaborating on our method, we present the underlying PF model. The accurate PF model can be obtained with complete system parameters and state measurements. Mathematically, for the i^{th} node, we denote u_i and v_i as the real and imaginary components of a voltage phasor. Also, we denote p_i and q_i as the corresponding active and reactive power. These quantities can formalize the power flow equation. For simplification of later derivations, we utilize the PF model in rectangular coordinates as follows.

$$\begin{aligned} p_i &= \sum_{k=1}^{|\mathcal{V}|} G_{ik}(u_i u_k + v_i v_k) + B_{ik}(v_i u_k - u_i v_k), \\ q_i &= \sum_{k=1}^{|\mathcal{V}|} G_{ik}(v_i u_k - u_i v_k) - B_{ik}(v_i u_k - u_i v_k), \end{aligned} \tag{1}$$

where $|\cdot|$ for a set represents the operation to obtain the set cardinality. G_{ik} and B_{ik} represent the real and imaginary parts of the $(i,k)^{th}$ entry of the admittance matrix, respectively.

However, for distribution systems, the system parameters G_{ik} and B_{ik} are usually unknown and the measurements of u_i, v_i, p_i, and q_i are only available when $i \in \mathcal{O}$. More importantly, [21] theoretically proves that it's impossible to obtain complete system parameters using partial measurements. Thus, we cannot obtain the exact PF model and need an accurate approximation tool. Namely, we need to find a map f such that $\boldsymbol{y} \approx f(\boldsymbol{x})$, where $\boldsymbol{x} = [u_1, \cdots, u_{|\mathcal{O}|}, v_1, \cdots, v_{|\mathcal{O}|}]^T$ and $\boldsymbol{y} = [p_1, \cdots, p_{|\mathcal{O}|}, q_1, \cdots, q_{|\mathcal{O}|}]^T$ are the measurement variables from nodes in $\mathcal{O}$. T represents the transpose operation.

There are diversified ML models to represent the mapping function f. In this paper, we propose that the selected model should have two distinguishing properties. First, the model should have enough capacity to capture the nonlinear correlations of input voltages, including the explicit quadratic relationship of observed voltages and implicit approximation of hidden voltages. Second, the model should maintain the consistency of the PF equations. Specifically, the model parameters in f should retain the same values and sparsity as the PF model. Therefore, the learned f can be generalized to different system loading conditions. To summarize, we have the following problem definition.

- Problem: ML-based PF approximation with maximal physical consistency.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Smart Grid. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSG.2023.3278702

- Input: N measurements $\{\boldsymbol{x}_n\}_{n=1}^N$ and $\{\boldsymbol{y}_n\}_{n=1}^N$ from observed nodes, where $\boldsymbol{x}_n$ and $\boldsymbol{y}_n$ are the n^{th} samples of vector variables $\boldsymbol{x}$ and $\boldsymbol{y}$, respectively.
- Output: an accurate mapping f with high approximation capacity and maximal physical consistency.

We utilize voltage phasor for parameter estimation, which may have high correlations. Hence, it's demanded to verify the estimation accuracy. We show following treatments are effective in our paper. First, The nonlinear transformation can reduce the correlations of voltage phasor. For example, the nonlinear terms, i.e., u_iu_k, u_iv_k, and v_iv_k, are treated as the input to the linear layer of the following DSN model. Although the correlation coefficients ρ between Voltage Magnitude (VM) or Voltage Angle (VA) are large, we find the values of ρ reduce significantly for variables among u_iu_k, u_iv_k, and v_iv_k. Table I shows the absolute average ρ values calculated by measurements of the IEEE 18-bus system with real-world load profiles. Obviously, the nonlinear terms have a much smaller value, bringing a high performance for topology identification and parameter estimation.

Second, the correlation can be further decreased if we consider the incremental data [14]–[16]. Basically, we can calculate the incremental term $\Delta u_iu_k[t] = u_iu_k[t+1] - u_iu_k[t]$, where t is the index for the time slot. Table I shows that the correlation value for the incremental term is significantly reduced compared to that of the raw voltage phasor. In the following derivations, we assume the notations defined in this section represent the incremental data. Third, in our following Equation 2, the l_1 based regularization with the suitable tuning of λ_1 can lead to excellent recovery of the topology and line parameters [31]. Due to the above three treatments, we show in Table II in Experiment that our DSN with l_1 regularization can obtain accurate topology and line parameters.

TABLE I: The absolute average value for the correlation coefficient.

	VM	VA	NOLINEAR TERM	INCREMENTAL TERM
$\lvert\rho\rvert$	0.89	0.91	0.56	0.41

III. Proposed Model

A. Deep-Shallow Neural Network from Elementary Examples

In this subsection, we investigate the design of the neural networks to achieve consistency of the PF equations. First, we consider two elementary examples for the model design. **Case 1. All nodes are observable (metered) in the system.** Under this condition, Equation (1) implies that we can construct a 2-layer shallow NN to represent the PF equations. The top left of Fig. 1 illustrates a 4-node fully metered system. Then, the corresponding PF equations can be represented by the shallow neural network at the bottom left of Fig. 1, where the input goes through a multiplication layer and a linear summation layer to predict the output. The multiplication layer helps to compute the multiplicative terms like u_1^2 and u_1u_2 in (1) and the summation layer helps to enforce the weighted summation for multiplicative terms, which is consistent with the PF equations.

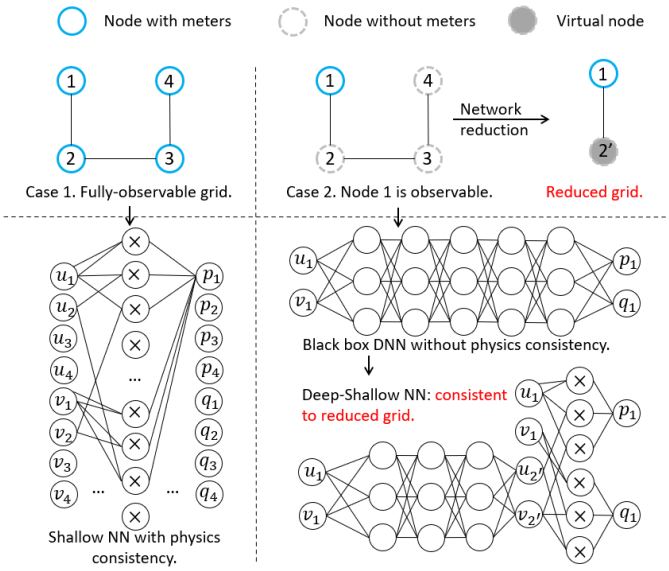


Fig. 1: The elementary examples to construct DSN.

Next, we consider the second case. **Case 2. Only one node is observable in the system.** As shown in the top right of Fig. 1, only node 1 has measurements. Then, we cannot directly utilize the shallow NN due to the missing quantities. Thus, a natural evolution of the solution is to directly deploy a black-box DNN to map from the inputs to the outputs, shown in the middle right of Fig. 1. Obviously, such a black-box model cannot maintain physical consistency and can easily encounter issues like overfitting. To handle this problem, we propose a structural design of the DNN model with a certain level of physical consistency and interpretability.

Specifically, we propose to create a virtual node in Case 2 and form a 2-node system with the virtual node and the observed node. As shown in the top right of Fig. 1, in a reduced grid, node 1 is observable and node $2^{'}$ is the virtual node that represents the aggregation of the unobservable nodes. Then, the voltage and power of the virtual node and the observed node can fit the shallow NN like Case 1. In particular, the shallow NN can take in measurements of $[u_1, v_1, u_{2'}, v_{2'}]^T$ and output $[p_1, q_1, p_{2'}, q_{2'}]^T$. Based on the power balance constraint, $p_{2'}$ and $q_{2'}$ can be completely determined by $v_1, v_{2'}, u_1, u_{2'}$, and the parameters of line 12. Thus, we only need to build the mapping from $[u_1, v_1, u_{2'}, v_{2'}]^T$ to $[p_1, q_1]^T$ and learn accurate line parameters. However, the pre-requisite is to obtain good estimates of $u_{2'}$ and $v_{2'}$ for node $2^{'}$.

To estimate $u_{2'}$ and $v_{2'}$, we make use of the universal approximation power of DNNs. Namely, we build another DNN model to capture the voltage measurement correlations and map from the input $[u_1, v_1]^T$ to $[u_{2'}, v_{2'}]^T$. Subsequently, the neurons representing $[u_{2'}, v_{2'}]^T$ can be treated as input to the shallow NN, formulating a Deep-Shallow neural Network (DSN). The bottom right section of Fig. 1 represents the DSN structure for Case 2. Notably, there are no measurements of $u_{2'}$ and $v_{2'}$, and we can't directly guarantee the intermediate neurons in the DSN to have exact values of $u_{2'}$ and $v_{2'}$. To mitigate the problem, we can enforce physical constraints with clear definitions of voltage and line parameters in the

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Smart Grid. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSG.2023.3278702

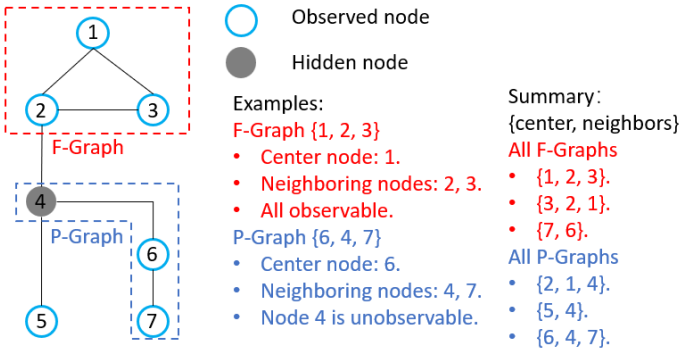


Fig. 2: Distribution system graph model and unit graph decomposition.

DSN, i.e., the intermediate neurons and the weights of the shallow layer in the DSN. More specifically, we can calculate other power system quantities with the data of voltage and line parameters, which enables the implementation of different physical constraints. More details are shown in Section III-B. In general, the deep-shallow hierarchy and physical constraints make the DSN consistent with underlying PF equations.

B. General DSN Case to Create Reduced Graphs

This subsection extends the DSN model to the general case where a subset of nodes in the distribution grid is observed. We specify the model equations and formally propose an efficient training algorithm. The right part of Fig. 2 illustrates a distribution grid with part of the nodes observable. To build a DSN for such a grid, the key is to identify how to place the virtual node and formulate the reduced grid.

For this goal, the focus is on small grid components as in Case 2 in Section III-A. Specifically, we decompose the whole graph $\mathcal{G}$ of the distribution grid into $|\mathcal{O}|$ unit-graphs $\{\mathcal{G}_i = \{\mathcal{V}_i, \mathcal{E}_i\}\}_{i=1}^{|\mathcal{O}|}$ with the graph center to be one observable node and radius to be 1 (the distance between every two connected nodes is defined to be 1). Then, for each center node i, denote $\text{Neigh}(i) = \{k | ik \in \mathcal{E}\}$. Thus, $\{i \cup \text{Neigh}(i)\}$ is the node set for $\mathcal{G}_i$. Obviously, based on the presence of hidden nodes in $\mathcal{G}_i$, the unit-graph may or may not require a virtual node. Therefore, we have the following categorizations.

F-Graph. We denote a fully-observable unit graph as a F-Graph, as shown in the middel part of Fig. 2. Namely, for a F-Graph $\mathcal{G}_i$, all nodes in $\mathcal{G}_i$ have micro-PMUs. For example, in Fig. 2, nodes $\{1, 2, 3\}$ form an F-Graph where node 1 is the center node and nodes 2 and 3 are neighboring nodes. Thus, $\mathcal{F} = \{i | i \text{ is a center of a F-Graph}\}$. Then, based on (1), $\forall i \in \mathcal{F}$ and all $k \in \text{Neigh}(i)$, the data of $[v_i, u_i, v_k, u_k]^T$ and $[p_i, q_i]^T$ can be the inputs and outputs of the 2-layer shallow NN in the DSN, respectively. Nevertheless, the unknown connectivity implies the missing knowledge of $\mathcal{F}$, i, and $\mathcal{G}_i$. The issue can be solved via a sparsity-based pre-training method. Basically, one can denote a mapping of the 2-layer shallow NN as $f_{S_1}(\boldsymbol{x}) = \boldsymbol{W}_1 \Phi_1(\boldsymbol{x})$, where $\Phi_1(\cdot) : \mathbb{R}^{2|\mathcal{O}|} \to \mathbb{R}^{4(|\mathcal{O}|+\binom{|\mathcal{O}|}{2})}$ represents the map to create the multiplicative terms like u_i^2 and $u_i u_k$ for input neurons u_i and u_k and $\binom{\cdot}{\cdot}$ represents the operation to calculate the combination number. $\boldsymbol{W}_1^{2|\mathcal{O}| \times 4(|\mathcal{O}|+\binom{|\mathcal{O}|}{2})}$ is the weight matrix for the linear layer. For the multiplication layer, we fix the neuron order such that the first $4|\mathcal{O}|$ neurons represent the square terms like u_i^2 and the following $4\binom{|\mathcal{O}|}{2}$ neurons represent the multi-multiplicative terms like $u_i u_k$. It should be noted that f_{S_1} is only used to tackle observed nodes and we have another shallow NN f_{S_2} to tackle the virtual nodes in the later descriptions. For f_{S_1}, our proposed pre-training tries to minimize

$$\frac{1}{N} \sum_{n=1}^{N} ||\boldsymbol{y}_n - f_{S_1}(\boldsymbol{x}_n)||_2^2 + \lambda_1 ||\boldsymbol{W}_1||_1, \tag{2}$$

where $||\cdot||_p$ is the l_p norm, and λ_1 is a positive constant. Namely, we add the l_1 norm of $\boldsymbol{W}_1$ as a regularization term to enforce sparsity that selects correct multipliers for p_i and q_i where $i \in \mathcal{F}$. Correspondingly, the sparse selection helps to identify the connection between nodes i and k and produces non-zero terms if $k \in \text{Neigh}(i)$, which has been verified in existing work [18], [32], [33]. Notably, for node $h \notin \mathcal{F}$, the sparse selection does not work and results in under-fitting and a large loss due to the missing data. Thus, to distinguish the node set $\mathcal{F}$, we create an evaluation standard based on the loss:

$$\frac{1}{N} \sum_{n=1}^{N} ||\boldsymbol{y}_n[i] - f_{S_1}(\boldsymbol{x}_n)||_2^2 \leq \epsilon, \tag{3}$$

where $\boldsymbol{y}_n[i]$ is the i^{th} element of $\boldsymbol{y}_n$ and ϵ is a positive threshold. If (3) holds, then $i \in \mathcal{F}$. Then, based on the non-zero terms in $\boldsymbol{W}_1$, the system can easily find the set $\text{Neigh}(i)$. Specifically, let $\psi(i, k)$ denote the vector of neuron indices in the multiplication layer of the shallow NN such that the elements in $\psi(i, k)$ represent the indices of multiplier $u_i u_k$, $u_i v_k$, $v_i u_k$, and $v_i v_k$, sequentially. Then, we denote $\text{Neigh}(i) = \{k | \exists k' \in \psi(i, k), \boldsymbol{W}_1[i, k'] \neq 0 \text{ or } \boldsymbol{W}_1[i + |\mathcal{O}|, k'] \neq 0\}$ and define the F-Graph $\mathcal{G}_i = \{\mathcal{V}_i, \mathcal{E}_i\}$ as:

$$\mathcal{V}_i = \{i \cup \text{Neigh}(i)\}, \mathcal{E}_i = \{ik, k \in \text{Neigh}(i)\} \tag{4}$$

Initial Guess and Constraints on DSN based on F-Graphs. To fully utilize the pre-trained results, the following initialization and constraints, denoted as *F-Graph initialization and constraints*, are

- **Initialization 1**. We note that the pre-training using Equation (2) can bring a good parameter estimate for lines in $\mathcal{G}_i, \forall i \in \mathcal{F}$. Thus, we keep parameters $\boldsymbol{W}_1[i, k']$ and $\boldsymbol{W}_1[i + |\mathcal{O}|, k']$ for all $i \in \mathcal{F}$, $k \in \text{Neigh}(i)$, and $k' \in \psi(i, k)$. In the meantime, we zero out the rest elements in $\boldsymbol{W}_1$ since these elements (i.e., line parameters out of $\mathcal{G}_i$) are inaccurately estimated.
- **Initialization 2**. The symmetry of the admittance matrix helps to re-assign estimated parameters to some entries in $\boldsymbol{W}_1$. More specifically, $\forall h \in \mathcal{O} \setminus \mathcal{F}$, if there exists $i \in \mathcal{F}$ such that $h \in \text{Neigh}(i)$, we have $\boldsymbol{W}_1[h, h'] = \boldsymbol{W}_1[i, h']$ and $\boldsymbol{W}_1[h + |\mathcal{O}|, h'] = \boldsymbol{W}_1[i + |\mathcal{O}|, h']$, where $h' \in \psi(i, h)$. In general, this re-assignment process enforces the coefficients of multipliers to be the same in different equations., e.g., the estimated coefficient of $u_i u_h$ is the same for equations to calculate p_i and p_h.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Smart Grid. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSG.2023.3278702

- **Constraint 1**. The known connectivity of F-Graphs requires a constraint for the following training process, i.e., $\forall i \in \mathcal{F}, h \notin \text{Neigh}(i), \boldsymbol{W}_1[i, \psi(i,h)] = \boldsymbol{W}_1[j, \psi(i,h)] = \boldsymbol{W}_1[i + |\mathcal{O}|, \psi(i,h)] = \boldsymbol{W}_1[h + |\mathcal{O}|, \psi(i,h)] = \boldsymbol{0}$.

The above initialization and constraint are necessary due to the following reasons. (1) They restrict the partial weights in the shallow NN in the DSN to be consistent with line parameters and connections within F-Graphs. (2) They give a good estimate of the weights from the pre-training. Therefore, the shallow NN in the DSN can completely represent the PF equations within F-Graphs. (3) They make full use of the symmetry of the admittance matrix to guarantee weights for node $h \in \mathcal{O} \setminus \mathcal{F}$.

Despite above efforts, there are hidden nodes in $\text{Neigh}(h)$. Thus, we need to investigate the impacts of these hidden nodes and how to represent the PF equations within the neighborhood of h. Specifically, we propose the following definition.

P-Graph. We denote a partially-observable unit graph as a P-Graph, as shown in the middle part of Fig. 2. Namely, for a P-Graph $\mathcal{G}_h$, there exists at least one node $k \in \text{Neigh}(h)$ such that node k does not have a micro-PMU. For example, in Fig. 2, nodes $\{6, 4, 7\}$ form a P-Graph where node 6 is the center node and nodes 4 and 7 are the 1-hop neighboring nodes. Next, node 4 is hidden. Then, we have $\mathcal{P} = \{h | h \text{ is a center of a P-Graph}\}$. As we have identified $\mathcal{F}$ using (3), we can compute $\mathcal{P} = \mathcal{O} \setminus \mathcal{F}$. Note that in the above F-Graph, as a result of the identification and parameter initialization process, the connectivity and line parameters of node i and h have been determined if $i \in \mathcal{F}$ and $h \in \mathcal{P}$. However, other connections can hardly be directly identified due to the hidden data. Thus, for each P-Graph, one can create a virtual node as in Case 2 in Section III-A. The virtual node is an aggregated node to represent the remainder of the grid for one P-Graph. Therefore, we can assume $\forall h \in \mathcal{P}$, each node in $\text{Neigh}(h)$ is either in $\mathcal{F}$ or a virtual node. Since all F-Graphs are known, we can identify the P-Graph $\mathcal{G}_h = \{\mathcal{V}_h, \mathcal{E}_h\}$ as follows.

$$\begin{aligned} \text{Neigh}(h) &= \{v\} \cup \{i | i \in \mathcal{F}, h \in \text{Neigh}(i)\}, \\ \mathcal{V}_i &= \{i \cup \text{Neigh}(i)\}, \mathcal{E}_i = \{ih, h \in \text{Neigh}(i)\}, \end{aligned} \tag{5}$$

where v is an index of a virtual node. The right part of Fig. 2 summarizes all the F-Graphs and P-Graphs for the presented system.

Virtual Nodes Creation and Complete DSN Structures. We introduce a DNN model f_D to map from $\boldsymbol{x}$ to the voltage of virtual nodes whose number is the same as $|\mathcal{P}|$. For the sake of convenience of later derivations, we denote the vector of voltage components for nodes in $\mathcal{F}$, $\mathcal{P}$, and virtual nodes as $\boldsymbol{x}_F$, $\boldsymbol{x}_P$, and $\boldsymbol{x}_V$, respectively, and $\boldsymbol{x}_V$ can be computed as $\boldsymbol{x}_V = f_D(\boldsymbol{x})$. Similarly, let $\boldsymbol{y}_F$ and $\boldsymbol{y}_P$ be the power data of nodes in $\mathcal{F}$ and $\mathcal{P}$, respectively. By default, set $\boldsymbol{x} = [\boldsymbol{x}_F; \boldsymbol{x}_P]$ and $\boldsymbol{y} = [\boldsymbol{y}_F; \boldsymbol{y}_P]$, where $[\cdot;\cdot]$ is the vector concatenation. According to the above designs, $\boldsymbol{y}_F$ has been accurately approximated by $f_{S_1}(\boldsymbol{x})$. Therefore, we need another shallow NN to approximate the contributions of virtual nodes to $\boldsymbol{y}_P$ and simultaneously maintain the physical consistency.

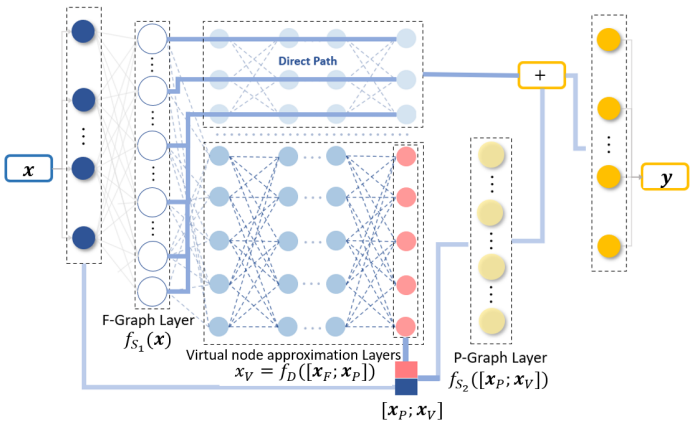


Fig. 3: The design of the DSN.

Subsequently, we introduce f_{S_2} such that $f_{S_2}([\boldsymbol{x}_P; \boldsymbol{x}_V]) = \boldsymbol{W}_2 \Phi_2([\boldsymbol{x}_P; \boldsymbol{x}_V])$, $\Phi_2(\cdot) : \mathbb{R}^{4|\mathcal{P}|} \to \mathbb{R}^{4(2|\mathcal{P}|+\binom{2|\mathcal{P}|}{2})}$ maps from voltage components to their multiplicative terms and $\boldsymbol{W}_2 \in \mathbb{R}^{2|\mathcal{P}| \times 4(2|\mathcal{P}|+\binom{2|\mathcal{P}|}{2})}$ is the linear weight matrix. With the defined f_{S_1}, f_{S_2}, and f_D, we can construct our DSN as shown in Fig. 3. The model formula is:

$$f([\boldsymbol{x}_F; \boldsymbol{x}_P]) = f_{S_1}([\boldsymbol{x}_F; \boldsymbol{x}_P]) + [\boldsymbol{0}; f_{S_2}([\boldsymbol{x}_P; f_D([\boldsymbol{x}_F; \boldsymbol{x}_P])])], \tag{6}$$

where a zero vector $\boldsymbol{0}$ is concatenated since virtual nodes bring zero contributions to $\boldsymbol{y}_F$. Generally, we enforce f_{S_1} to approximate $\boldsymbol{y}_F$ and f_{S_1} and f_{S_2} to approximate $\boldsymbol{y}_P$. Therefore, f_{S_1} and f_{S_2} can successfully represent the PF equations of the reduced grids with virtual nodes. Moreover, we can conveniently enforce physical constraints for virtual nodes, making the training of the DSN more efficient.

Constraints of DSN based on P-Graphs. The prior knowledge of distribution grids can enforce certain constraints for the P-Graphs. For example, we can enforce the following constraints that are common to all distribution grids, denoted as *P-Graph constraints*.

- The voltage components of virtual nodes $\boldsymbol{x}_V$ should be constrained in an interval $\mathcal{I} = [I_1, I_2]$ to maintain the voltage stability.
- The known connectivity of P-Graphs requires a constraint. Namely, $\forall h \in \mathcal{P}, k \notin \text{Neigh}(h)$, $\boldsymbol{W}_2[h, \psi(h,k)] = \boldsymbol{W}_2[h + |\mathcal{P}|, \psi(h,k)] = \boldsymbol{0}$.

While we numerically observe that the above two constraints can bring good results, some other domain knowledge can also be introduced, e.g., the range of line parameters and line power. More specifically, we can utilize the voltage values in the intermediate neurons $\boldsymbol{x}_V$ and the line parameter values in $\boldsymbol{W}_2$ to calculate the line power and introduce similar penalty terms for line parameters and line power. In general, the variable voltage of virtual nodes $\boldsymbol{x}_V$ and line parameters in $\boldsymbol{W}_2$ provide significant possibilities to enforce power system constraints and make DSN consistent to power flow equations, which never appear in previous work. During the training, the connectivity constraints on the F-Graph and G-Graph can be easily satisfied by restricting the corresponding 0s to be non-trainable all the time. For the interval constraint of

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Smart Grid. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSG.2023.3278702

$\boldsymbol{x}_V$, one can add penalty terms when the $\boldsymbol{x}_V$ falls outside of the interval. Specifically, we define the interval distance $d(\boldsymbol{x}_V, \mathcal{I}) = \max\big(\max(\boldsymbol{x}_V - I_2, 0), \max(I_1 - \boldsymbol{x}_V, 0)\big)$. Then, the DSN is trained via minimizing

$$\frac{1}{N}\sum_{n=1}^{N} ||\boldsymbol{y}_n - f(\boldsymbol{x}_n)||_2^2 + \lambda_2 d(\boldsymbol{x}_V, \mathcal{I}), \tag{7}$$

where λ_2 is a positive constant. We show the following two steps can make sure $\boldsymbol{x}_v$ can fall inside the interval.

- Initialization. For the DNN $f_D(\cdot)$ in the DSN, we set the last layer's bias vector to be $\mathbf{1}$, i.e., all entries are ones. For the rest weight matrices and bias vectors, we set their entries to be zeros. Consequently, the initial output of $f_D(\cdot)$ is $\mathbf{1}$.
- Output clipping. The clipping technique helps to restrict the quantity within a certain range [34]. For entries in $\boldsymbol{x}_V = f_D([\boldsymbol{x}_F; \boldsymbol{x}_P])$ that are smaller than I_1 (or larger than I_2), we can simply set the corresponding entries to be I_1 (or I_2).

Finally, since the above process provides the connections within F-Graphs and P-Graphs, these unit graphs can be connected together to obtain a set of reduced grids $\{\widetilde{\mathcal{G}}_k\}_k$. Namely, we check $\forall i \in \mathcal{O}$ and find the neighborhood nodes $\text{Neigh}(i)$ to build the connections. Further, the weights in $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ can determine the parameters in $\{\widetilde{\mathcal{G}}_k\}_k$. To summarize, the final training process of DSN can be seen in Algorithm 1.

Algorithm 1 Train-DSN($\{\boldsymbol{x}_n\}_{n=1}^N$, $\{\boldsymbol{y}_n\}_{n=1}^N$).

Input: Training data $\{\boldsymbol{x}_n\}_{n=1}^N$ and $\{\boldsymbol{y}_n\}_{n=1}^N$.

Output: the DSN model $f(\boldsymbol{x})$, the reduced grids $\{\widetilde{\mathcal{G}_k}\}_k$, and the corresponding parameters.

Hyper-parameter: penalty coefficient λ_1 and λ_2, threshold ϵ to identify $\mathcal{F}$, batch size M, and learning rate η.

1: Build $f_{S_1}(\boldsymbol{x})$. Then, train $f_{S_1}(\boldsymbol{x})$ under objective of (2) via Stochastic Gradient Descent (SGD) [35].
2: Identify the set $\mathcal{F}$ and $\mathcal{P}$ using (3).
3: Obtain F-Graphs and P-Graphs by (4) and (5), respectively.
4: Implement *F-Graph initialization and constraints* for $f_{S_1}(\cdot)$ and *P-Graph constraints* for $f_{S_2}(\cdot)$.
5: Utilize $\mathcal{F}$ and $\mathcal{P}$ to construct DSN via (6).
6: Train DSN under the objective of (7) via SGD.
7: Connect F-Graphs and P-Graphs to obtain a set of reduced grids $\{\widetilde{\mathcal{G}_k}\}_k$ and utilize the trained $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ to identify line parameters for $\{\widetilde{\mathcal{G}_k}\}_k$.

Remark: We provide the following explanations and summary of the recoverable and non-recoverable line parameters. First, authors in [21] prove that it's impossible to recover all the parameters and connectivity without full-rank measurement matrices, let alone partially-available phasor measurements. Thus, our goal in the DSN model is to recover as many parameters as possible while maintaining the functional form and constraints, bringing excellent performances under different operational conditions. Second, to achieve the goal, we decompose the system into F-Graphs and P-Graphs in Section III-B. We show in Equation (2) that the parameters of the F-Graph can be estimated via a 2-layer shallow NN with sparsity regularization, i.e., parameters of the line $i_1 i_2$ where $i_1, i_2 \in \mathcal{F}$ and the line $i_1 h_1$ where $i_1 \in \mathcal{F}$ and $h_1 \in \mathcal{P}$. These are all the recoverable parameters. Third, without additional information, it's impossible to accurately recover the parameters of the rest lines, including line $h_1 h_2$ where $h_1, h_2 \in \mathcal{P}$ and line $h_1 v$ where $h_1 \in \mathcal{P}$ and v is a neighboring virtual node. This is because there are infinite combinations of the voltage and line parameters related to virtual nodes to fit the observed measurements. However, assigning them physically feasible values can lead to physical consistency and high generalizability of the DSN model.

The above reduced grids $\{\widetilde{\mathcal{G}}_k\}_k$ are isolated by virtual nodes. In general, we make the DSN consistency to PF equations of $\{\widetilde{\mathcal{G}}_k\}_k$ via sparsity regularization and range/connectivity constraints in the *F-Graph constraints* and *P-Graph constraints*. This reduces overfitting and promotes generalizability. Nevertheless, the model for reduced grids may still be overly complicated due to too many virtual nodes. For example, some virtual nodes at the same location can be merged with the same voltage. Thus, we finally utilize an RL-based search algorithm to connect the reduced graphs, fine-tune DSN, reduce the intermediate neurons (i.e., the input to $f_{S_2}(\cdot)$), and find the best fit.

C. RL-based Search to Fine-tune DSN and Connect Reduced Graphs

For the $|\mathcal{P}|$ virtual nodes, there are at most $2^{\binom{|\mathcal{P}|}{2}}$ possible connections to be searched, which results in a high computational cost when $|\mathcal{P}|$ is large. To reduce the computational complexity, we enforce constraints to enable a valid search space by requiring the search result to be a sparse and connected graph.

Mathematically, we denote the virtual node set as $\mathcal{N} = \{n_1, n_2, \cdots, n_{|\mathcal{P}|}\}$ to make them distinguishable from previous node indices. Then, the search starts from node n_1 and chooses another node $n_i \in \mathcal{N} \setminus \{n_1\}$ to make a connection $n_1 n_i$. We denote the action of selecting n_i as a_1. Subsequently, we can conduct a_2 such that the next choice of node n_h comes from the set $n_h \in \mathcal{N} \setminus \{n_1, n_i\}$ to make the connection $n_i n_h$. After repeating the above process $|\mathcal{P}|-1$ times with sequential actions $\{a_1, a_2, \cdots, a_{|\mathcal{P}|-1}\}$, there is a tree structure among all the virtual nodes, resulting in a connected graph. For example, in Fig. 4, the search process add connections by actions a_1 and a_2 to obtain a tree structure.

However, there may exist loops within the connections of the virtual nodes to better represent the true grid. Thus, we enable extra $M+1$ actions $\{a_{|\mathcal{P}|}, \cdots, a_{|\mathcal{P}|+M}\}$ to enable but not necessarily create the loop structures. More specifically, for $1 \le m \le M+1$, we denote n_k as the current node. Then, the action $a_{|\mathcal{P}|+m}$ selects a node from the set $\mathcal{N}$. Next, if the new connection already exists, the graph remains unchanged after this action. For instance, in Fig. 4, the action a_3 is to build the connection between n_3 and n_2, which already exists. Otherwise, another connection is built for constructing

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Smart Grid. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSG.2023.3278702

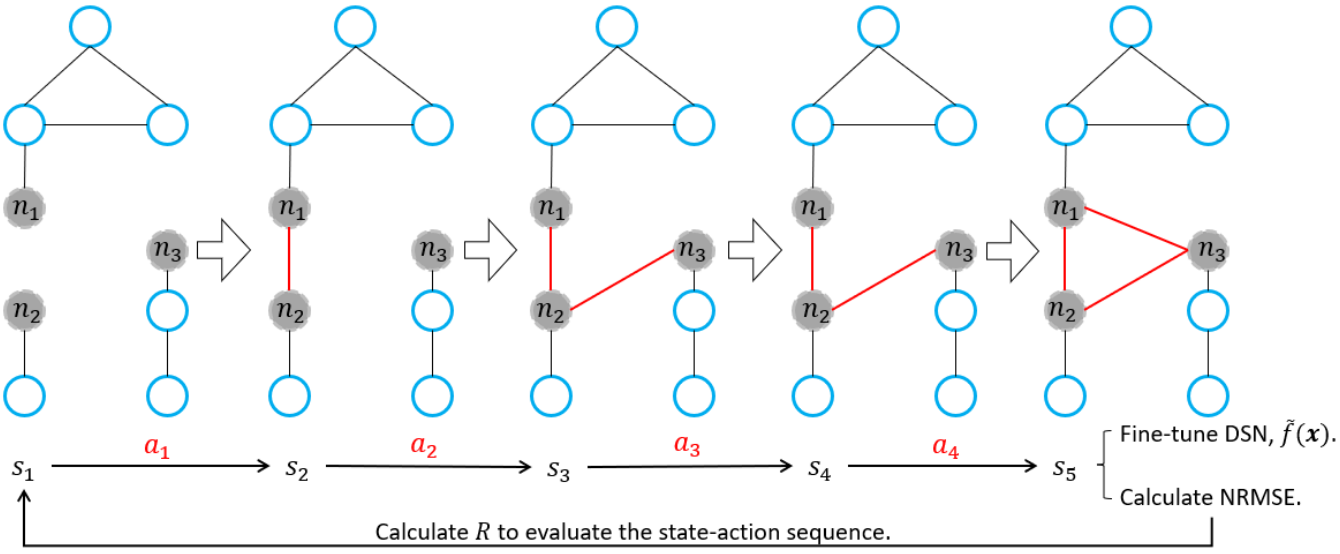


Fig. 4: An illustrative example of the RL-based search for the connected graph.

a loop. The action a_4 in Fig. 4 connects node n_3 and n_1 and formulates a triangle. Obviously, we only need to restrict M to a moderate number so that the sparsity is guaranteed. There are the following approaches to determine a moderate value. (1) We can directly assign a large M (e.g., M=$|\mathcal{P}|$). Since the action to build loops can repeat to build the same connection (e.g., a_3 and a_4 in Fig. 4), a search algorithm can still find the optimal structure with a large M and enough search trials. (2) We can assign a reasonable value based on some prior knowledge. For example, one may set $M = -1$ if the network is radial. (3) We can dynamically increase the value of M in the search process. This saves computational time and improves convergence. In general, we believe there is a large room to explore search-based topology identification as future work. In the numerical section, we set $M = 1$ since our test cases are radial distribution grids. Eventually, we find that there are no loops for the best-fit topology.

In general, the above $|\mathcal{P}| + M$ iterations generate a state sequence $\{s_1, s_2, \cdots, s_{|\mathcal{P}|+M+1}\}$ and an action sequence $\{a_1, a_2, \cdots, a_{|\mathcal{P}|+M}\}$, where each s_k represents the connectivity among virtual nodes at the k^{th} state. The state-action transition process essentially defines a Markov Decision Process (MDP) that can be conveniently optimized by utilizing Reinforcement Learning (RL) techniques. In this paper, we employ Q-learning to conduct the optimization due to its efficient convergence property [36].

After defining states and actions, we need a reward function to evaluate the action sequence and guide the training of the RL to generate the optimal action set. Intuitively, we prefer the selected actions (i.e., connections within $\mathcal{N}$) to better fit the training data. Thus, we propose to formulate the reward function based on the Mean Square Error (MSE) [37] of the DSN using the training dataset so that the search results can mostly fit the data. Next, the architecture of the DSN should be fine-tuned based on the search results according to the searched grid topology.

A simple and efficient fine-tuning process is adopted as follows. We can assume the line parameters are 0s between two connected virtual nodes. Namely, we can merge the virtual nodes n_i and n_k as long as line $n_i n_k$ is a searched line in $s_{|\mathcal{P}|+M+1}$. Then, the voltage estimates for nodes n_i and n_k should be equal. Therefore, in the DSN model, one only needs to merge the hidden neurons for the estimated voltage of n_i and n_k, leading to a new DSN structure. We denote the new mapping function as $\widetilde{f}(\boldsymbol{x})$.

Subsequently, we can train $\widetilde{f}(\boldsymbol{x})$ using the same objective as (7). The MSE error of the trained $\widetilde{f}(\boldsymbol{x})$ can help to construct the reward. Basically, the metric is the Normalized Root-Mean-Square Error (NRMSE) [38], [39] such that NRMSE $= \frac{1}{\sigma_y}\sqrt{\frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{y}_i - \widetilde{f}(\boldsymbol{x}_i))^2}$, where σ_y is the standard deviation of the outputs. Then, the reward can be calculated as $R = \frac{1}{1+\text{NRMSE}}$. R is a terminal reward for evaluating the performance of the state and action sequences after one search. Consequently, for each state transition (s_i, a_i, s_{i+1}), we also assign R as the reward (7). With defined states, actions, and rewards, our Q learning algorithm tries to iteratively update the Q function:

$$Q_{t+1}(s_k, a_k) = Q_t(s_k, a_k) + \alpha\big(R_t + \gamma \max_a Q_t(s_{k+1}, a) - Q_t(s_k, a_k)\big), \tag{8}$$

where $Q(s_k, a_k)$ is a function to evaluate the k^{th} state-action pair ($1 \le k \le |\mathcal{P}|+M$), and t as a subscript shows it's the t^{th} episode. α and γ are pre-defined learning rate and discount factor, respectively. The optimization $\max_a Q_t(s_{k+1}, a)$ is a lookup of the Q-table and determines the maximum Q value given s_{k+1}. Finally, we employ the greedy method to encourage some explorations of random actions in Q learning. We summarize the Q-learning algorithm in Algorithm 2.

Our model is not necessarily limited to handling radial networks. For the observed nodes, we categorize them into F-Graphs and P-Graphs. However, they can both be loopy. In particular, for F-Graphs, the loopy structure and line parameters can be easily estimated using the optimization in Equation

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Smart Grid. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSG.2023.3278702

Algorithm 2 Q learning to search $\widetilde{\mathcal{G}}$.

Input: Training data $\{\boldsymbol{x}_n\}_{n=1}^N$ and $\{\boldsymbol{y}_n\}_{n=1}^N$.

Output: the DSN model $\widetilde{f}(\boldsymbol{x})$, the connected reduced grids $\widetilde{\mathcal{G}}$, and the corresponding parameters.

Hyper-parameter: penalty coefficients λ_1 and λ_2, threshold ϵ to identify $\mathcal{F}$, batch size M, learning rate η, p for the greedy method to select the action in Q-learning, α and γ as learning rate and discount factor respectively, for the Q-function updating, the maximum number of episode T, and M as the extra action number.

$f(\boldsymbol{x})$, $\{\widetilde{\mathcal{G}_k}\}_k$ = Train-DSN($\{\boldsymbol{x}_n\}_{n=1}^N$, $\{\boldsymbol{y}_n\}_{n=1}^N$).
for $t = 1$ to T **do**
  for $k = 1$ to $M + |\mathcal{P}|$ **do**
    Conduct a random sample from the uniform distribution $p_0 \sim U(0, 1)$.
    if $p_0 \leq p$ **then**
      Choose a random action for a_k.
    else
      Choose $a_k = \arg\max_a Q_t(s_{k+1}, a)$.
    end if
    Conduct a_k and transit from s_k to s_{k+1}.
  end for
  Generate a state sequence $\{s_1, s_2, \cdots, s_{|\mathcal{P}|+M}\}$ and an action sequence $\{a_1, a_2, \cdots, a_{|\mathcal{P}|+M}\}$.
  Utilize connection in $s_{|\mathcal{P}|+M}$ to generate $\widetilde{\mathcal{G}}_t$ and the corresponding DSN model $\widetilde{f}_t(\boldsymbol{x})$.
  Train $\widetilde{f}_t(\boldsymbol{x})$ via SGD and calculate the reward R_t.
  Utilize (8) to update the Q function.
end for
Set $\widetilde{f}(\boldsymbol{x}) = \widetilde{f}_T(\boldsymbol{x})$ and $\widetilde{\mathcal{G}} = \widetilde{\mathcal{G}}_T$. Then, utilize the trained $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ to identify line parameters for $\widetilde{\mathcal{G}}$.

2 as long as all nodes are observable, e.g., the triangle structure in Fig. 2 and Fig. 4. For P-Graphs, the remaining unidentifiable connections and parameters are related to the virtual nodes. Although we add one singleton virtual node with one line for each P-Graph, our RL-based search algorithm provides possibilities for the construction of loops among different virtual nodes, as stated in this subsection and displayed in Fig. 4. Therefore, our model can tackle loopy structures for both observed and hidden regions.

It's remarkable that our DSN is a flexible model that can make the best use of input information to provide a good approximation of the PF equations. Basically, (1) The structural design of DSN is based on the identification of the F-Graphs and P-Graphs. Consequently, line parameters in the F-Graphs can be accurately estimated and stored in the weights of the DSN. (2) The functional form and the plug-and-play physical constraints for virtual nodes can further improve the consistency of DSN to the underlying PF equations of reduced grids. Then, the end-to-end training of DSN refines the model and guarantees high generalizability to diversified operational conditions. (3) The RL-based search algorithm can connect all reduced grids and fine-tune the DSN model. Therefore, the final DSN model is optimal to represent the PF equations of a connected graph and best fit the data.

With the above properties, the obtained DSN model can (i) reduce to the PF equation if the system is fully observable, (ii) estimate as many line parameters as possible for F-Graphs and some lines of P-Graphs, and (iii) conduct universal approximation with physical constraints if only singleton nodes are observable. In conclusion, under different observability levels, our DSN model can deliver a good solution for system modeling. Finally, our numerical results in Fig. 5 and Table II further support our claims. We find that even when the observability level $\gamma = 0.1$, the power prediction error and line parameter estimation error are still very limited compared to other methods.

IV. Experiment

A. Dataset Description

In our experiment, we utilize simulated data from standard IEEE distribution systems [40] and residential load profiles [1]. IEEE provides standard power system models, including the grid topology, parameters, and generation models. for accurate simulations. The model files and the simulation platform, MATPOWER [40], are based on MATLAB. In this experiment, we incorporate IEEE distribution networks of 18-, 33-, 115-, and 123-bus systems for testing. To conduct the simulation, the load files are required as the input to the systems. Specifically, we employ the Pacific Gas and Electric Company's (PG&E's) hourly residential load profile (8760 samples for one year) in North California [1]. With the above data, MATPOWER produces the system states of voltage and power in (1).

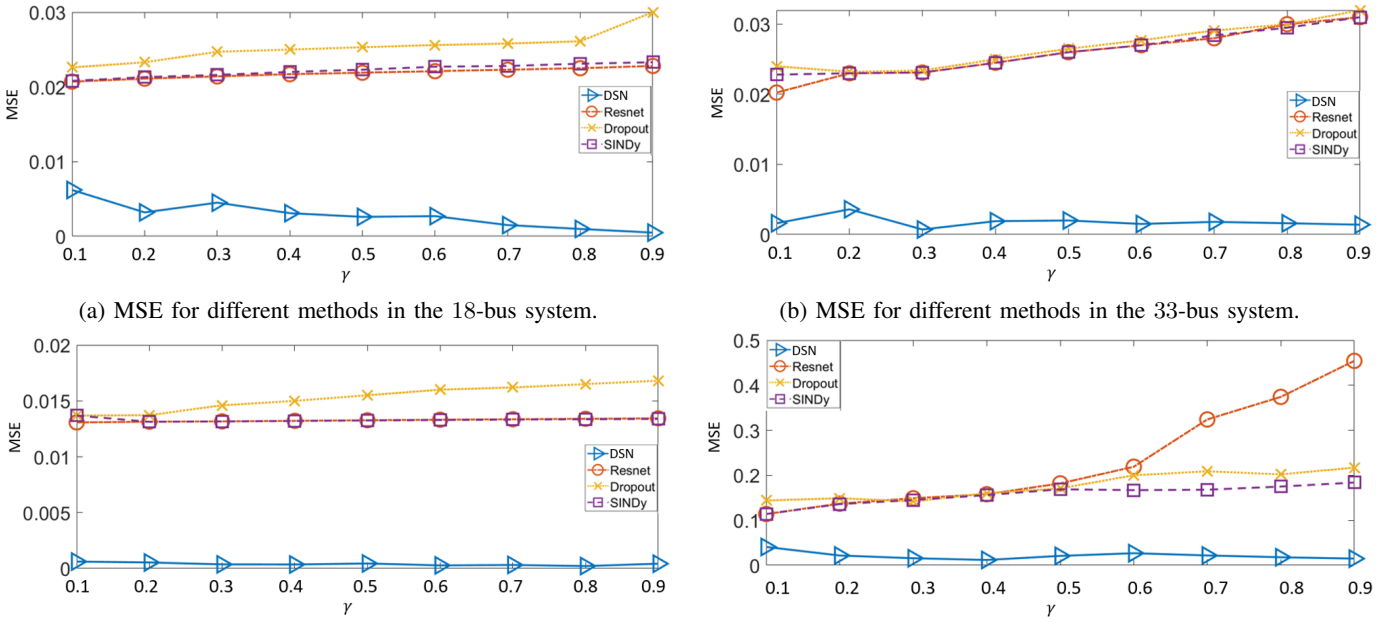
B. Benchmark Models

To fully investigate the strong interpretability and generalizability of the DSN, we compare our proposed DSN with other advanced DNN models. Specifically, we have the following benchmark models for comparison.

- **Resnet** [41]. Deep Residual Network creates a shortcut connection to pass the deep information directly to the shallow layers. Such a skip-connection effect not only helps to avoid gradient vanishing issues in the training phase, but also contributes to the model generalization ability since the low-complexity features are connected to the output, thus decreasing the model complexity.
- **SINDYs** [32], [42]. The sparse identification of nonlinear dynamics (SINDy) utilizes the sparse regression technique to recover the parameters of the physical systems, while the base of the regression can be selected via DNNs. In our experiments, we employ the quadratic term as the symbolic bases and eliminate the DNN part for simplicity.
- **DNNs with Dropout Method** [43]. Dropout method randomly disables neurons in training, thus preventing the neurons from over co-adapting and increasing the model generalizability.
- **Physics-conditioned GAN** [44]. Physics-conditioned Generative Adversarial Network (GAN) is a cutting-edge method to conduct distribution grid state estimation with pseudo-measurements of hidden nodes. In Section IV-F,

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Smart Grid. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSG.2023.3278702



(a) MSE for different methods in the 18-bus system.

(b) MSE for different methods in the 33-bus system.

(c) MSE for different methods in the 115-bus system.

(d) MSE for different methods in the 123-bus system.

Fig. 5: Results of MSE for different models and different distribution grids.

we demonstrate that our DSN can also conduct state estimation with pseudo-measurements for hidden nodes and achieve better performance.

C. Model Evaluation

We propose the following metrics to evaluate the generalizability and interpretability of DSN and benchmark models.

Generalizability. The generalizability of the ML model represents the capacity to take the place of the underlying PF equations when the system condition changes. To evaluate the generalizability, we conduct a 5-fold cross-validation. Mean Square Error (MSE) of the testing dataset is used as the measure for generalization errors.

Parameter estimation. SINDYs and our DSN method have the capacity to estimate the distribution system parameters. To evaluate the system identification performance, we consider the lines among node set $\mathcal{O}$, where both the line weight estimation error and the connectivity should be evaluated. Since the connectivity can be converted to the sparsity of the Laplacian matrix, we utilize the so-called normalized Total Vector Error ($nTVE$) [9] to evaluate the difference between the estimated $\hat{\boldsymbol{L}}$ and the true Laplacian matrix $\boldsymbol{L}$:

$$nTVE = 100 \times \frac{||\hat{\boldsymbol{L}} - \boldsymbol{L}||_2}{||\boldsymbol{L}||_2}.$$

D. Results for Model Generalizability

In our experiments, we test different systems with changing PMU penetration level $\gamma \in \{0.1, 0.2, \cdots, 0.9\}$ to comprehensively compare the model generalizability among different methods and 5-fold cross-validation is conducted. The results are shown in Fig. 5a to 5d. We find that for each trial, our DSN **always achieves the lowest MSE value** in the validation dataset. Further, the MSE of our DSN decreases as γ increases, while for other methods, the MSE increases.

The lowest generalization error comes from (1) the well-extracted local governing equations that are generalizable to different datasets and (2) the physical constraints that enable the physical variables to be within the physical range. Also, the increase of sensor penetration (γ) leads to more physical parameters being captured, thus decreasing the MSE further. However, for other methods without physical consistency, MSE will increase due to the growth of the output dimensionality.

E. Results for Network Parameter Estimation

TABLE II: $nTVE(\%)$ error of parameter estimation for DSN and SINDy methods.

	18-bus		33-bus		115-bus		123-bus	
γ	DSN	SINDy	DSN	SINDy	DSN	SINDy	DSN	SINDy
0.1	29	41	35	36	36	89	53	77
0.2	11	47	26	31	31	83	45	71
0.3	16	35	23	34	28	81	48	68
0.4	13	38	27	28	24	78	37	62
0.5	15	28	19	26	21	72	34	54
0.6	22	27	14	24	25	61	26	59
0.7	11	23	9.5	17	16	55	22	45
0.8	7.2	15	7.6	15	9.8	29	11	40
0.9	5.3	14	6.3	18	9.3	21	8.4	33

In this subsection, we show the results pertaining to the second function of our DSN, i.e., estimating the edge weights to construct the underlying physical equations. For the line parameters and connectivity among observed nodes $\mathcal{O}$, we calculate the $nTVE(\%)$ for evaluation. The comparison is between our DSN and the SINDy since other DNNs cannot estimate the physical equation parameters.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Smart Grid. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSG.2023.3278702

The results are shown in Table 1. Generally, our DSN far outperforms the SINDy method for all systems when $\gamma < 0.5$. Empirically, the DSN's $nTVE$ is around $10\% \sim 25\%$ of the SINDy's $nTVE$. When γ increases, the performance of the DSN and SINDy will become closer. However, DSN's $nTVE$ still only covers around $40\% \sim 60\%$ of SINDy's $nTVE$. The reasons are as follows. (1) DSN employs a testing criterion in (3) to decompose $\mathcal{O}$ into $\mathcal{F}$ and $\mathcal{P}$. Then, the initialization rule of the DSN can enable the shared weights between $\mathcal{F}$ and $\mathcal{P}$ to always be accurately estimated in the pre-training of the F-Graph. For the SINDy method, however, the shared weight estimation incurs errors due to hidden quantities. (2) when $\gamma < 0.5$, the hidden nodes are dominant so that the DSN model performs much better than SINDy. (3) when γ is increasing, the number of hidden nodes decreases so that the inaccurate estimation of the shared weights in SINDy decreases, forcing DSN and SINDy to have closer performance.

F. Results for State Estimation of Hidden Nodes

In this subsection, we demonstrate the additional function of our DSN to estimate the voltage of hidden nodes if we have extra pseudo-measurements. In particular, for partially-observable systems, pseudo-measurements of hidden regions are often needed to enable state estimation for all hidden states [44]. Namely, one needs to have access to historical data of hidden nodes, which may have low data quality and large errors. Existing work employs the Gaussian mixture model [45], 2-stage clustering [46], NN to process pseudo-measurements [47], [48], and Generative Adversarial Network (GAN) [44] for distribution grid state estimation. With pseudo-measurements, we find that despite the lack of system parameters, our DSN model can also bring accurate hidden state estimation. This is mainly because our DSN model can (1) utilize pseudo-measurements to pre-train the DNN in our DSN and approximate a deterministic map between observed states and hidden states without PF equations, (2) employ shallow NNs to learn the underlying PF equations and further regularize the approximated hidden states, and (3) enforce power system constraints for the approximated hidden states.

While the problem deserves a lot of further effort to investigate different system assumptions in future work, we present a preliminary study by assuming that we know the locations and low-quality pseudo-measurements of hidden nodes. Hence, we can use the measurements of observed states and pseudo-measurements of observed and hidden states to train the DNN in the DSN. Subsequently, the known locations of hidden nodes enable us to replace virtual nodes with hidden nodes to construct the shallow NNs in the DSN. Finally, we can jointly utilize the measurements of observed nodes and low-quality pseudo-measurements of hidden nodes to estimate the line parameters and hidden states.

To numerically support our claim, we compare our methods with the recent method Physics-conditioned GAN [44] for the MSE error of the hidden state estimation. Specifically, we employ 8760 voltage and power samples for observed nodes and 500 voltage pseudo-measurements for observed and hidden nodes. The former samples are used to conduct end-to-end training for the DSN model while the latter measurements help to pre-train the DNN model in the DSN.

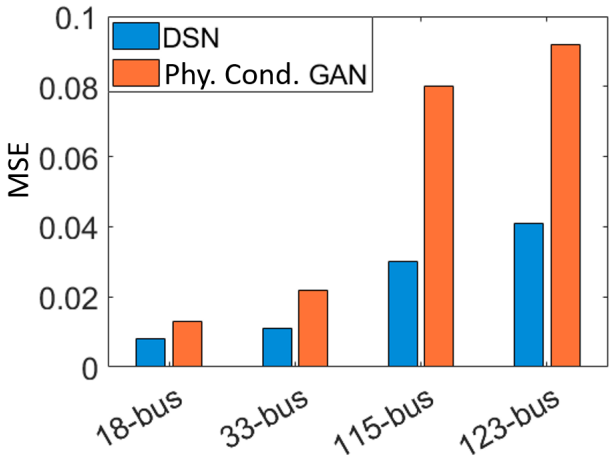


Fig. 6: The MSE for hidden state estimation.

Fig. 6 demonstrates the result. Specifically, our DSN achieves a $40\% \sim 60\%$ error decrease compared to physics-conditioned GAN. The key reason is that physics-conditioned GAN maps from noise vectors to the hidden data while our built-in DNN captures the relationship between the voltage of observed and hidden nodes. Obviously, voltage correlations are easier to learn. The noise vector-based transformation, however, highly depends on the data quality and sample numbers of the pseudo-measurements. Moreover, although physics-conditioned GAN assumes known system parameters to conduct state estimation, our DSN can learn the underlying parameters to facilitate the inference of hidden states. Therefore, the accurate estimation performs similarly to the physical law in the physics-conditioned GAN.

V. Discussion of the Utilization of DSN

We strongly believe that our DSN model can work as a foundation for steady-state system modeling and many other applications. Specifically, we present the following potential utilization.

- Parameter estimation and topology identification with partial observability. For distribution systems with sparse measurement locations, our DSN model can provide accurate parameter estimation for the identifiable portions (e.g., line parameters of F-Graphs). Specifically, the sparsity in the shallow NNs in the DSN implies the underlying topology, and the corresponding weight matrices can be simplified to obtain the line parameters. The identified topology and parameters lay the foundations for the distribution grid operation and optimization.
- Power prediction. Since our DSN is a mapping from observed voltage to power, the power prediction is a natural function given input voltage. The function may be useful when sensor communication fails or the power data are tampered. Therefore, our DSN with high generalizability can provide a validation of the power consumption.
- State estimation. While our DSN is a forward mapping from voltage to power, it can be treated as a

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Smart Grid. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSG.2023.3278702

building block for distribution system state estimation, an important function for system modeling in face of distributed generation and bi-directional power flow. First, since DSN is a neural network, we can embed DSN as the encoder part to an autoencoder [49]. Therefore, the decoder can transform from power to voltage, becoming an NN-based state estimator. Notably, the high physical consistency of our DSN can implicitly boost the performance of the decoder with physical constraints. Additionally, as stated in Section IV-F, our DSN can provide estimate states for hidden nodes given some pseudo-measurements.

- Outage detection. Renewable energy and bi-directional power flows may increase the chances of line outages for the distribution grid. Luckily, our DSN provides power prediction and line parameter estimation, which may be useful to detect and locate the line outage that brings significant load trips [50].
- Bad data or false data detection. The learned parameters in the DSN can help calculate the residual error for bad or false data detection [51]. Traditional detection methods rely on full system observability and complete knowledge of system parameters. Nevertheless, our DSN provides a doable approach with partial observability. Mathematically, the residual error can be calculated by the difference between the predicted and the measured power for the DSN.
- Renewable power and load profile generation. Power profile generation for residential loads or renewable energy is important for system stability. In particular, recent advances employ Generative Adversarial Networks (GANs) to generate profiles due to the high approximation power of GANs for data distributions [52]. While traditional GANs directly map from noise vectors to power data in their generators, we can create another generator that maps from normal system states to the power, i.e., our DSN model.

The above examples imply that when the underlying PF equations are missing, our DSN model is an important function to correlate system states, power, and parameters. Due to the physical consistency and the NN structure, we can easily combine our DSN with other NN-based functions for power system applications. Except for the listed points, there is still a large room to explore the utilization of our DSN. We will thoroughly discuss them in future work.

VI. Conclusion

We propose a Deep-Shallow Neural Network (DSN) to identify the PF equations of distribution systems under partial observability. DSN can be automatically shrunk to Power Flow (PF) equations for fully-observable areas and aggregate hidden nodes to virtual nodes to create a reduced grid. Then, DSN maintains the physical ranges and topology of the reduced grid with proper physical constraints. Finally, we provide a Reinforcement Learning (RL)-based search algorithm to connect the reduced grids to form a connected grid. In this process, the DSN enjoys fine-tuning to represent the PF equation of the connected grid. Extensive experiments are conducted over various distribution systems demonstrating reduced errors compared to contemporary methods.

Acknowledgement

Thanks to Erik Blasch for the concept development and document editing. The research was partially sponsored under AFOSR grant FA9550-22-1-0294, the Department of Energy under grants DE-AR00001858-1631 and DE-EE0009355, and the National Science Foundation (NSF) under the grants ECCS-1810537 and ECCS-2048288.

References

[1] Y. Weng, Y. Liao, and R. Rajagopal, "Distributed Energy Resources Topology Identification via Graphical Modeling," *IEEE Transactions on Power Systems*, vol. 32, no. 4, pp. 2682–2694, Jul 2017.

[2] D. Lew, M. Asano, J. Boemer, C. Ching, U. Focken, R. Hydzik, M. Lange, and A. Motley, "The Power of Small: The Effects of Distributed Energy Resources on System Reliability," *IEEE Power and Energy Magazine*, vol. 15, no. 6, pp. 50–60, Nov 2017.

[3] P. D. F. Ferreira, P. M. S. Carvalho, L. A. F. M. Ferreira, and M. D. Ilic, "Distributed Energy Resources Integration Challenges in Low-Voltage Networks: Voltage Control Limitations and Risk of Cascading," *IEEE Transactions on Sustainable Energy*, vol. 4, no. 1, pp. 82–88, Jan 2013.

[4] A. Yazdaninejadi, A. Hamidi, S. Golshannavaz, F. Aminifar, and S. Teimourzadeh, "Impact of Inverter-based DERs Integration on Protection, Control, Operation, and Planning of Electrical Distribution Grids," *The Electricity Journal*, vol. 32, no. 6, pp. 43–56, 2019.

[5] S. Bhela, V. Kekatos, and S. Veeramachaneni, "Enhancing Observability in Distribution Grids Using Smart Meter Data," *IEEE Transactions on Smart Grid*, vol. 9, no. 6, pp. 5953–5961, Nov 2018.

[6] C. Rudin, D. Waltz, R. N. Anderson, A. Boulanger, A. Salleb-Aouissi, M. Chow, H. Dutta, P. N. Gross, B. Huang, S. Ierome, D. F. Isaac, A. Kressner, R. J. Passonneau, A. Radeva, and L. Wu, "Machine Learning for the New York City Power Grid," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 2, pp. 328–345, Feb 2012.

[7] C. Rudin, S. Ertekin, R. Passonneau, A. Radeva, A. Tomar, B. Xie, S. Lewis, M. Riddle, D. Pangsrivinij, and T. McCormick, "Analytics for Power Grid Distribution Reliability in New York City," *Interfaces*, vol. 44, pp. 364–383, 08 2014.

[8] F. Darema, E. Blasch, S. Ravela, and A. Aved, "Dynamic Data Driven Applications Systems: Third International Conference," vol. 12312, 2020.

[9] H. Li, Y. Weng, Y. Liao, B. Keel, and K. E. Brown, "Distribution Grid Impedance & Topology Estimation with Limited or No Micro-PMUs," *International Journal of Electrical Power & Energy Systems*, vol. 129, p. 106794, 2021.

[10] A. E. Abu-Elanien, M. Salama, and K. B. Shaban, "Modern Network Reconfiguration Techniques for Service Restoration in Distribution Systems: A Step to a Smarter Grid," *Alexandria engineering journal*, vol. 57, no. 4, pp. 3959–3967, 2018.

[11] D. Deka, S. Backhaus, and M. Chertkov, "Estimating Distribution Grid Topologies: A Graphical Learning based Approach," vol. 23, no. 5, pp. 1–7, Jun 2016.

[12] G. Cavraro, V. Kekatos, and S. Veeramachaneni, "Voltage Analytics for Power Distribution Network Topology Verification," *IEEE Transactions on Smart Grid*, vol. 10, no. 1, pp. 1058–1067, Jan 2019.

[13] S. Bolognani, N. Bof, D. Michelotti, R. Muraro, and L. Schenato, "Identification of Power Distribution Network Topology via Voltage Correlation Analysis," vol. 20, no. 15, pp. 1659–1664, Dec 2013.

[14] Y. Liao, Y. Weng, G. Liu, and R. Rajagopal, "Urban MV and LV Distribution Grid Topology Estimation via Group Lasso," *IEEE Transactions on Power Systems*, vol. 34, no. 1, pp. 12–27, 2018.

[15] D. Deka, S. Backhaus, and M. Chertkov, "Structure Learning and Statistical Estimation in Distribution Networks-Part I," *arXiv preprint arXiv:1501.04131*, 2015.

[16] ——, "Structure Learning and Statistical Estimation in Distribution Networks-Part II," *arXiv preprint arXiv:1502.07820, 2015*, 2015.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Transactions on Smart Grid. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSG.2023.3278702

[17] J. Yu, Y. Weng, and R. Rajagopal, "PaToPa: A Data-Driven Parameter and Topology Joint Estimation Framework in Distribution Grids," *IEEE Transactions on Power Systems*, vol. 33, no. 4, pp. 4335–4347, July 2018.

[18] O. Ardakanian, V. W. S. Wong, R. Dobbe, S. H. Low, A. von Meier, C. J. Tomlin, and Y. Yuan, "On Identification of Distribution Grids," *IEEE Transactions on Control of Network Systems*, vol. 6, no. 3, pp. 950–960, 2019.

[19] J. Yu, Y. Weng, and R. Rajagopal, "PaToPaEM: A Data-Driven Parameter and Topology Joint Estimation Framework for Time-Varying System in Distribution Grids," *IEEE Transactions on Power Systems*, vol. 34, no. 3, pp. 1682–1692, May 2019.

[20] G. Cavraro and V. Kekatos, "Graph Algorithms for Topology Identification Using Power Grid Probing," *IEEE Control Systems Letters*, vol. 2, no. 4, pp. 689–694, Oct 2018.

[21] K. Moffat, M. Bariya, and A. Von Meier, "Unsupervised Impedance and Topology Estimation of Distribution Networks—Limitations and Tools," *IEEE Transactions on Smart Grid*, vol. 11, no. 1, pp. 846–856, Jan 2020.

[22] G. Cavraro and V. Kekatos, "Inverter Probing for Power Distribution Network Topology Processing," *IEEE Transactions on Control of Network Systems*, vol. 6, no. 3, pp. 980–992, 2019.

[23] Y. Yuan, S. H. Low, O. Ardakanian, and C. J. Tomlin, "Inverse Power Flow Problem," *IEEE Transactions on Control of Network Systems*, vol. 10, no. 1, pp. 261–273, 2023.

[24] Y. Liu, N. Zhang, Y. Wang, J. Yang, and C. Kang, "Data-Driven Power Flow Linearization: A Regression Approach," *IEEE Transactions on Smart Grid*, vol. 10, no. 3, pp. 2569–2580, 2019.

[25] H. Oh, "A New Network Reduction Methodology for Power System Planning Studies," *IEEE Transactions on Power Systems*, vol. 25, no. 2, pp. 677–684, 2009.

[26] J. Yu, Y. Weng, and R. Rajagopal, "Robust Mapping Rule Estimation for Power Flow Analysis in Distribution Grids," in *2017 North American Power Symposium (NAPS)*, 2017, pp. 1–6.

[27] J. Yuan and Y. Weng, "Support Matrix Regression for Learning Power Flow in Distribution Grid With Unobservability," *IEEE Transactions on Power Systems*, vol. 37, no. 2, pp. 1151–1161, 2022.

[28] J.-H. Menke, N. Bornhorst, and M. Braun, "Distribution System Monitoring for Smart Power Grids with Distributed Generation Using Artificial Neural Networks," *International Journal of Electrical Power & Energy Systems*, vol. 113, pp. 472–480, 2019.

[29] S. Lakshminarayana, S. Sthapit, and C. Maple, "A Comparison of Data-Driven Techniques for Power Grid Parameter Estimation," *arXiv preprint arXiv:2107.03762*, 2021.

[30] A. S. Zamzam and N. D. Sidiropoulos, "Physics-Aware Neural Networks for Distribution System State Estimation," *IEEE Transactions on Power Systems*, vol. 35, no. 6, pp. 4347–4356, 2020.

[31] M. Hebiri and J. Lederer, "How Correlations Influence Lasso Prediction," *IEEE Transactions on Information Theory*, vol. 59, no. 3, pp. 1846–1854, 2013.

[32] S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Discovering Governing Equations from Data by Sparse Identification of Nonlinear Dynamical Systems," *Proceedings of the national academy of sciences*, vol. 113, no. 15, pp. 3932–3937, 2016.

[33] H. Li and Y. Weng, "Physical Equation Discovery Using Physics-Consistent Neural Network (PCNN) Under Incomplete Observability," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 925–933.

[34] B. Zhang, J. Jin, C. Fang, and L. Wang, "Improved Analysis of Clipping Algorithms for Non-convex Optimization," *Advances in Neural Information Processing Systems*, vol. 33, pp. 15 511–15 521, 2020.

[35] S.-i. Amari, "Backpropagation and Stochastic Gradient Descent Method," *Neurocomputing*, vol. 5, no. 4-5, pp. 185–196, 1993.

[36] F. S. Melo, "Convergence of Q-learning: A Simple Proof," *Institute Of Systems and Robotics, Tech. Rep*, pp. 1–4, 2001.

[37] D. M. Allen, "Mean Square Error of Prediction as a Criterion for Selecting Variables," *Technometrics*, vol. 13, no. 3, pp. 469–475, 1971.

[38] B. K. Petersen, M. L. Larma, T. N. Mundhenk, C. P. Santiago, S. K. Kim, and J. T. Kim, "Deep Symbolic Regression: Recovering Mathematical Expressions from Data via Risk-Seeking Policy Gradients," in *International Conference on Learning Representations*, 2021.

[39] H. Li, Y. Weng, and H. Tong, "Console: Convex neural symbolic learning," in *Advances in Neural Information Processing Systems*, vol. 35. Curran Associates, Inc., 2022, pp. 5766–5774.

[40] MATPOWER community, "Index for Matpower6.0," 2020, https://matpower.org/docs/ref/matpower6.0/menu6.0.html.

[41] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[42] K. Champion, B. Lusch, J. N. Kutz, and S. L. Brunton, "Data-Driven Discovery of Coordinates and Governing Equations," *Proceedings of the National Academy of Sciences*, vol. 116, no. 45, pp. 22 445–22 451, 2019.

[43] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[44] M. Kamal, W. Li, D. Deka, and H. Mohsenian-Rad, "Physics-Conditioned Generative Adversarial Networks for State Estimation in Active Power Distribution Systems with Low Observability," in *2022 International Conference on Smart Grid Synchronized Measurements and Analytics (SGSMA)*. IEEE, 2022, pp. 1–6.

[45] R. Singh, B. Pal, and R. Jabr, "Distribution System State Estimation through Gaussian Mixture Model of the Load as Pseudo-measurement," *IET generation, transmission & distribution*, vol. 4, no. 1, pp. 50–59, 2010.

[46] Y. R. Gahrooei, A. Khodabakhshian, and R.-A. Hooshmand, "A New Pseudo Load Profile Determination Approach in Low Voltage Distribution Networks," *IEEE Transactions on Power Systems*, vol. 33, no. 1, pp. 463–472, 2017.

[47] E. Manitsas, R. Singh, B. C. Pal, and G. Strbac, "Distribution System State Estimation Using An Artificial Neural Network Approach for Pseudo Measurement Modeling," *IEEE Transactions on power systems*, vol. 27, no. 4, pp. 1888–1896, 2012.

[48] K. R. Mestav, J. Luengo-Rozas, and L. Tong, "Bayesian State Estimation for Unobservable Distribution Systems via Deep Learning," *IEEE Transactions on Power Systems*, vol. 34, no. 6, pp. 4910–4920, 2019.

[49] V. Miranda, J. Krstulovic, H. Keko, C. Moreira, and J. Pereira, "Reconstructing Missing Data in State Estimation with Autoencoders," *IEEE Transactions on power systems*, vol. 27, no. 2, pp. 604–611, 2011.

[50] A. N. Samudrala, M. H. Amini, S. Kar, and R. S. Blum, "Distributed Outage Detection in Power Distribution Networks," *IEEE Transactions on Smart Grid*, vol. 11, no. 6, pp. 5124–5137, 2020.

[51] Y. Gu, T. Liu, D. Wang, X. Guan, and Z. Xu, "Bad Data Detection Method for Smart Grids based on Distributed State Estimation," in *2013 IEEE International Conference on Communications (ICC)*. IEEE, 2013, pp. 4483–4487.

[52] Y. Gu, Q. Chen, K. Liu, L. Xie, and C. Kang, "GAN-based Model for Residential Load Generation Considering Typical Consumption Patterns," in *2019 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*. IEEE, 2019, pp. 1–5.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/