Hd-Deep-EM: Deep Expectation Maximization for Dynamic Hidden State Recovery Using Heterogeneous Data

Zhihao Ma, *Student Member, IEEE,* Haoran Li, *Student Member, IEEE,* Yang Weng, *Senior Member, IEEE,* Erik Blasch, *Fellow, IEEE,* Xiaodong Zheng, *Senior Member, IEEE,*

***Abstract*—Uncertain power generations and loads are continually integrated into the power system, causing high risks of dynamic events. To better monitor systems, advanced meters like Phasor Measurement Units (PMUs) can record high-resolution system dynamic states in real time. However, due to the high cost, the placement of PMUs is limited in a system. Thus, it's hard to obtain all the dynamic system states via PMUs. On the other hand, traditional sensors in a Supervisory Control and Data Acquisition (SCADA) system can broadly cover the system, though they only provide low-resolution measurements. In this paper, we propose to utilize PMU and SCADA sensor data to recover the missing dynamic states in the power generation system. The problem has the following challenges based on unique properties of data: (1) Spatially, PMU and SCADA sensors have different locations. Thus, it's a must to approximate the data correlations to estimate the missing data accurately. (2) Temporally, the dynamic transitions in SCADA samples are scarce, urging efficient utilization of the SCADA data to approximate the dynamics. For challenge (1), we employ Deep Neural Networks (DNNs) with high capacities to capture spatial-temporal information to predict dynamic states. For challenge (2), we develop a new mechanism to utilize SCADA data efficiently. Specifically, we iteratively reuse the predicted dynamic states in the SCADA data to retrain the DNN model, gradually increasing the performance. The effectiveness of the proposed training procedure is theoretically verified via the framework of Expectation-Maximization (EM). Thus, our model to fuse heterogeneous data is termed Heterogeneous data Deep EM (Hd-Deep-EM). Finally, we demonstrate the high performance of the Hd-Deep-EM in diversified synthetic and realistic power systems.**

***Index Terms*—Dynamic state estimation, power systems, limited PMUs, deep learning, expectation maximization**

I. Introduction

A power-generation system integrates stochastic components and loads to facilitate clean and low-cost production and consumption, such as Photovoltaic (PV) power and wind power that highly depend on uncertain weather conditions. These components make the power system vulnerable to dynamic environmental events. To better capture system dynamics, Phasor Measurement Units (PMUs) are continually integrated into the power system to provide synchronized phasor measurements with 30-120 samples per second [1]. Such a high resolution enables accurate monitoring of the dynamic states. However, these deployments of PMUs come with high costs, resulting in relatively low penetrations throughout the system.

Meanwhile, the conventional measuring system of power systems, i.e., Remote Terminal Unit (RTU) based SCADA system, has been largely deployed over the last decade [2]. Compared to PMUs, SCADA measurements have relatively low resolution and can't fully record system dynamics. For example, the SCADA system provides only 0.5-2 measurement samples per second [3]. Therefore, the dilemma exists of limited PMUs with large dynamic data and broad SCADA sensors with few dynamic records. To improve the knowledge of system dynamics, we combine the information of SCADA data and PMU measurements to accurately estimate the missing dynamic states between SCADA data.

To integrate PMU and SCADA data, there are many existing studies, categorized into model-based and model-free approaches. In the model-based approach, system dynamic models are utilized to identify the relationship of future states and historical states [4]–[6]. For example, a Kalman filter [7] predicts the future states using historical states of both PMU and SCADA data. In these methods, nonlinear differential equations of power systems are employed. However, these system equations are usually inaccurate [8], leading to non-negligible errors for the dynamic state estimation.

Therefore, model-free approaches are introduced to mine information in historical measurements for accurate prediction [9]–[11]. Model-free methods generally build a map to estimate hidden dynamic states using corresponding PMU measurements. For example, one can train a Linear Regression (LR) model [12] using synchronized PMU and SCADA measurements, and the hidden states can be predicted via inputting the intermediate PMU data at the same time. Though the LR-based method is simple and easy to understand, the model capacity is limited. To increase the capacity, improvements can be done by (1) increasing the single model nonlinearity for better approximation or (2) aggregating different models. For example, [13] examines the k-nearest neighbor (KNN) algorithm with non-linear mapping rules. [14] proposes a Bagged Averaging of Multiple Linear Regression model by integrating and bootstrapping several different LRs. However, it's still hard for these methods to capture the complex temporal and

Zhiaho Ma, Haoran Li, and Yang Weng are with the Department of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, AZ, 85281, USA. E-mail: {zhihaoma,lhaoran,yang.weng}@asu.edu.

Erik Blasch is with MOVEJ Analytics, Dayton, OH 45324, USA. E-mail: erik.blasch@gmail.com.

Xiaodong Zheng is with with the Key Laboratory of Control of Power Transmission and Conversion (Ministry of Education), Shanghai Jiao Tong University, Shanghai 200240, China. E-mail: xiaodongzheng@sjtu.edu.cn.

spatial correlations of data in PMUs and SCADA.

To identify hidden states effectively under a complex spatial-temporal correlation, deep learning methods are increasingly proposed. These methods utilize hierarchical feature learning to extract features spatially or temporally. To capture spatial correlations, methods like Convolutional Neural Networks (CNNs) [15]–[17] and Graph Neural Network (GNN) can employ square or graph-based kernels for convolution and extraction of local features. To obtain temporal correlations, Recursive Neural Networks (RNNs) [18], [19] and Long Short-Term Memory (LSTM) [20] employ gates to filter sequential information and learn useful patterns. In this paper, we make use of the capacity of DNN to analyze spatial-temporal data for dynamic state estimation. However, DNN-based methods require relatively large datasets with diversified information. DNNs may overfit when the information is limited to uncover the dynamic states. Notably, such a scenario often happens for SCADA data due to the low resolution. Thus, it's quite challenging to build a good DNN model to predict the dynamic states of SCADA systems.

This paper addresses the system problem by explicitly introducing "more" training data with dynamic information. Specifically, the new training data is derived from the predicted hidden states and the corresponding input PMU data, containing rich dynamic information. Subsequently, the DNN can be trained with all the data to further improve the prediction. These two steps form an iterative way to renew the NN model towards better performance. To elaborate on the effectiveness of the procedure, we show that the iterative process is essentially an Expectation-Maximization (EM) method with good convergence performance. Thus, our method is named as Heterogeneous data Deep EM (Hd-Deep-EM).

There are also other studies to combine deep learning and EM algorithms for different domains. Specifically, in signal processing, [21] develops a generalized expectation maximization to estimate channel states and detect signals. [22] detects signals by leveraging the deep unfolding to represent the iterative EM algorithm and improve the detection accuracy. In computer vision, [23] utilizes online EM algorithm to improve the inference in the deep Bayesian network, achieving good image classification performances. [23] reconstructs images via EM network and employs the power of EM algorithm to tackle noises. In general, those methods can hardly be applied to power system PMU and SCADA data due to different resolutions. They also don't provide solid theoretical guarantees. In this paper, we theoretically and experimentally clarify the effectiveness and efficiency of Hd-Deep-EM for dynamic hidden state estimation.

For the numerical verification, the Hd-Deep-EM method is tested extensively at various conditions on the synthetic 200- and 500-bus systems datasets, where the loading conditions and PMU/SCADA sensor penetrations are diversified for extensive testing. To show the power of Hd-Deep-EM in facing the spatial and temporal difference, we compare the proposed Hd-Deep-EM method with shallow EM algorithm and EM algorithm with a Vanilla neural network. To show the power of Hd-Deep-EM in tackling the temporal trend from the SCADA measurements, we compare Hd-Deep-EM with the traditional neural network. The benchmark methods include EM and deep learning approaches, and Mean Square Error (MSE) is used for evaluating interpolation model accuracy. The results show that our proposed Hd-Deep-EM method can efficiently interpolate missing data to the SCADA measurement.

The rest of the paper is organized as follows: Section II defines the problem. Section III proposes our Hd-Deep-EM. Section IV conduct experiments for baselines and Hd-Deep-EM and Section V concludes the paper.

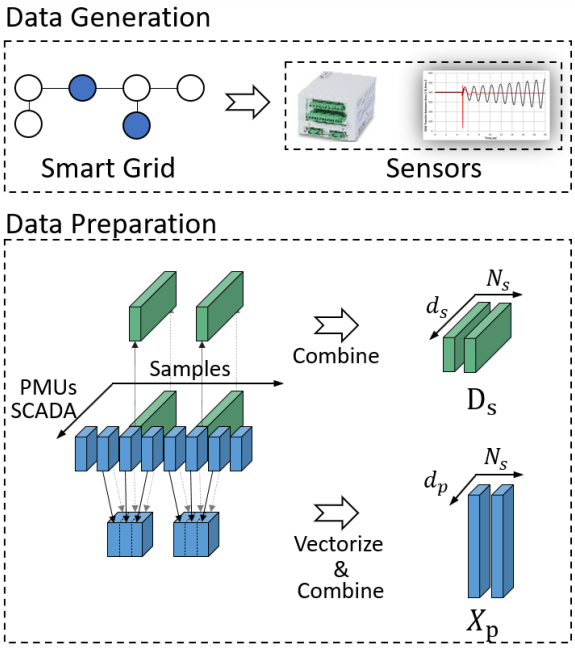

Fig. 1: A moving window-based segmentation procedure to process the PMU streams.

II. Problem Formulation

This section defines the problem with clear notations, which lays the foundations for the following derivations. We denote that N_p samples for d_p numbers of PMUs and N_s samples for d_s numbers of SCADA data. Then, let $\boldsymbol{D}_p \in \mathbb{R}^{N_p \times d_p}$ represent the PMU data matrix and $\boldsymbol{D}_s \in \mathbb{R}^{N_s \times d_s}$ for the SCADA data. Moreover, due to the difference in resolution among PMUs and SCADA data, we have $N_p \gg N_s$. Then, we denote the missing data matrix $\boldsymbol{Z} \in \mathbb{R}^{(N_p - N_s) \times d_s}$.

With defined PMU and SCADA data flows, we need to process the raw data and formalize the training data to estimate the dynamic states. While one can build a one-to-one mapping from the PMU measurements to the SCADA data simultaneously, the mapping may not be robust if data on either sides have outliers. Thus, to improve the robustness, we propose to utilize the PMU data of neighboring time slots as the input. Fig. 1 illustrates the process where we utilize a group of PMU data (i.e., the blue box) as input to estimate the SCADA sample (i.e., the green box). Naturally, if we input the intermediate PMU data that don't have the corresponding SCADA data, the prediction will be an estimate of the missing dynamic states of the SCADA data.

To predict the measurement states, mathematically, we employ a moving window to segment PMU streams based on the resolution of the SCADA data. As shown in Fig. 1, for each timestamp of SCADA data, the PMUs data will extract a $k \times d_p$ moving window that has a center of the target timestamp. Subsequently, each extracted moving window will be vectorized into an input vector. This procedure makes the processed PMUs and SCADA data have the same number of samples N_s. Therefore, we have $\boldsymbol{X}_p \in \mathbb{R}^{N_s \times (k \times d_p)}$ as the input PMU data matrix and $\boldsymbol{D}_s \in \mathbb{R}^{N_s \times d_s}$ as the output SCADA data. In order to recover the $\boldsymbol{Z}$ with function f, we also apply the moving window to the intermediate PMU streams to predict the missing states $\boldsymbol{Z}$. Specifically, we denote the PMU data matrix for recovering $\boldsymbol{Z}$ as $\tilde{\boldsymbol{X}}_p \in \mathbb{R}^{(N_p - N_s) \times (k \times d_p)}$. Finally, the strategy of this paper is to learn the mapping rule

$$f : \boldsymbol{X}_p \cup \tilde{\boldsymbol{X}}_p \rightarrow \boldsymbol{D}_s \cup \boldsymbol{Z}. \tag{1}$$

The complete problem definition can be summarized as follows.

- Problem: Deep expectation maximization for data interpolation.
- Given: a set of PMU-based samples with $\boldsymbol{D}_p \in \mathbb{R}^{N_p \times d_p}$ as the high resolution data and $\boldsymbol{D}_s \in \mathbb{R}^{N_s \times d_s}$ as the low resolution data, respectively.
- Output: a regression model to learn data mapping f between PMU data and SCADA data and interpolate the missing value of the SCADA data.

III. Proposed Model

The design of the above mapping f can be diversified. However, existing works can not sufficiently capture the spatial-temporal correlations of PMU and SCADA data. In this section, we first illustrate that a well-designed DNN for the function f can efficiently handle the spatial-temporal correlations and estimate the dynamic states. However, the training of the DNN may suffer overfitting due to the scarce dynamic information in SCADA data. Then, we show the key to resolve the issue is to completely make use of the hidden dynamic states to update the mapping. Such a hidden value estimation and parameter updating process lies in the domain of Expectation-Maximization (EM) algorithm. In particular, EM algorithms impact on numerous topics in power systems. They mostly target on probabilistic load or power source monitoring [24]–[26], where EM helps to estimate the underlying distributions with the maximal likelihood. For example, [24]–[26] develop a Gaussian Mixture Model (GMM) to model the density function of the load/source, and utilize EM algorithm to find a suboptimal solution. Some other applications appear in PMU-based cyber attack detection [27] and joint impedance and topology estimation [28]. For example, in [27], the objective becomes the log-likelihood of data with underlying cyber attack.

A. Expectation-Maximization Algorithm for Hidden State Estimation

EM algorithms are developed to tackle problems with latent variables, e.g., the hidden states in SCADA data streams. Specifically, the EM algorithm is an iterative process to separately estimate the statistics of latent variables and the corresponding parameters that determine the likelihood of the latent data. Thus, to apply EM algorithms to our problem, the first step is to identify what parameters should be used to determine the latent variables (i.e., the hidden states).

In general, one can categorize the parametric models into the generative and the discriminative models. The generative model estimates the joint distributions between the known and the unknown variables to generate new data. However, it's hard to choose an appropriate model to estimate hidden states. For example, GMM is usually used in most EM algorithms. However, the GMM model is not capable to represent the distribution of hidden states in power systems with complex spatial-temporal correlations.

Thus, in this paper, we propose to utilize a discriminative mapping f to directly map from PMU data to SCADA data. The advantage of a discriminative design is to maintain the spatial-temporal correlations in the PMU data, thus boosting the approximation accuracy of the hidden states. Mathematically, we can write the EM steps as follows.

- E step: Estimate the expected values of the hidden states in SCADA data streams. Namely, input the corresponding PMU data to f and output hidden states.
- M step: Update parameters of f for better estimation.

Obviously, the design of f should be carefully considered. In the following section, we display our design using deep learning techniques to achieve accurate dynamic state estimation.

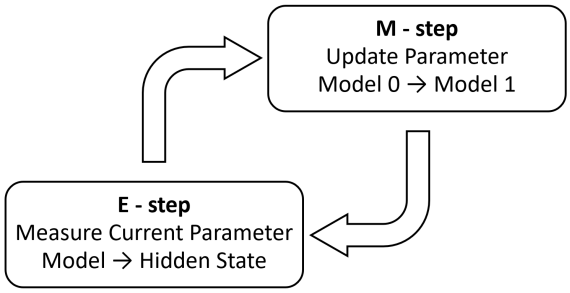

Fig. 2: An illustration of the EM algorithm.

Fig. 2 visualizes the process of the EM algorithms. To investigate the convergence of the EM algorithm, many studies are done and one can refer to [29] for more details.

B. Deep Neural Networks to Capture Spatial-Temporal Correlations

The core of our Hd-Deep-EM algorithm is the deep neural network (DNN) model, which is applied to approximate the data interpolation mapping between PMU to SCADA measurement. Intuitively, a DNN model can help capture the temporal-spatial correlations and learn useful features for the task. Specifically, the DNN model has multi-layer feed-forward neural network structure, which consists of typical three-level network architecture: one input layer, several hidden layers,

and one output layer. Since the input data comes from PMU data $\boldsymbol{X}_p$ and the output data comes from the SCADA data $\boldsymbol{D}_s$, we denote $\boldsymbol{x}$ as the variable for $\boldsymbol{X}_p$, $\boldsymbol{y}$ as the truth variable for $\boldsymbol{D}_s$, and $\hat{\boldsymbol{y}}$ as the predicted variable using the DNN. Then, the DNN function can be written as follows.

$$\begin{aligned} \boldsymbol{h}_0 &= \boldsymbol{x}, \\ \boldsymbol{h}_i &= \boldsymbol{g}_i\left(\boldsymbol{W}_i \boldsymbol{h}_{i-1} + b_{i-1}\right), \forall\ i = 1, \cdots, N_l, \\ \hat{\boldsymbol{y}} &= \boldsymbol{g}_{N_l+1}\left(\boldsymbol{W}_{N_l+1} \boldsymbol{h}_{N_l} + b_{N_l}\right), \end{aligned} \tag{2}$$

where N_l denotes the number of layers for the neural network, $\boldsymbol{h}_0$ denotes the input matrix of the network, $\boldsymbol{h}_i$ is the output matrix of the i-th hidden layer, b_i is the bias term, and $\boldsymbol{g}$ is the activation function for each hidden layer. We elaborate more on the model of the DNN based on the following perspectives.

1) Structure and Activations: DNN has a hierarchical structure to gradually extract non-linear features for the task. Generally, the i-th hidden layer models the interactions between the $(i-1)$-th features by introducing a connection weight matrix $\boldsymbol{W}_i$ and a bias vector b_i. Then, the result is activated via a non-linear activation function $g(\cdot)$. The choice of g can vary, and some common options include Rectified Linear Unit (ReLU), sigmoid function, and tangent function, etc. Notably, ReLU is the most commonly used because ReLU-based DNN can efficiently tackle the gradient vanishing problem and has a high efficiency to compute [30].

2) Loss function: After defining the structure and the inner activation functions, we need to train the DNN with a loss function. For this regression problem, the Mean Square Error (MSE) can smoothly measure the difference between the predicted output $\hat{\boldsymbol{y}}$ and the true output $\boldsymbol{y}$:

$$L(W, B, \boldsymbol{x}, \boldsymbol{y}) = \frac{1}{|N_s|} \sum_{i \in N_s} \left(\hat{\boldsymbol{y}} - \boldsymbol{y}\right)^2, \tag{3}$$

where $W = \{\boldsymbol{W}_i\}_i$ is the set of layer-wise weights of the DNN, and $B = \{b_i\}_i$ is the set of bias terms.

3) Training Process: With the defined loss function, training is conducted via minimizing the loss with given input/output data. The minimization can be written as:

$$\min_{W,B} L(W, B, \boldsymbol{x}, \boldsymbol{y}). \tag{4}$$

To solve the optimization, many algorithms can be utilized. For example, one can utilize either Stochastic Gradient Descent (SGD) [31] or the Adam [32] to train the model.

The defined DNN has the ability to learn the local features and determine the spatial-temporal correlations. To well-train the DNN model to approximate the dynamic states, we require the data to contain enough dynamic information. Nevertheless, the SCADA data has little dynamic information, decreasing the DNN model performance.

C. Deep Expectation Maximization to Boost the Training of the DNN

To resolve SCADA data scarcity, the key is to introduce more SCADA data with system dynamic information. As the dynamic systems states of the SCADA system are hidden, the problem can be decomposed into (1) estimating hidden values and (2) updating the model parameters as shown in Fig. 3. Mathematically, this iterative procedure is an Expectation-Maximization (EM) algorithm. This subsection derives the Hd-Deep-EM algorithm from training the DNN model, leading to the so-called Hd-Deep-EM model.

In general, the EM algorithm tries to solve an optimization with missing quantities. For the optimization to train the DNN in Equation (4), the hidden dynamic states of the SCADA data are introduced. Thus, the optimization can be rewritten as:

$$\min_{W,B} L(W, B, \boldsymbol{z}, \boldsymbol{x}, \boldsymbol{y}), \tag{5}$$

where $\boldsymbol{z}$ represents the variable of missing values in the SCADA data. Namely, $\boldsymbol{Z} = \{\boldsymbol{z}_i\}_{i=1}^{N_p - N_s}$. Clearly, $\boldsymbol{z}$ is also the output of the DNN f given the input PMU data $\boldsymbol{x}$ at the corresponding time. However, the unknown knowledge of $\boldsymbol{z}$ makes the direct optimization of Equation (5) impossible.

Therefore, the EM algorithm solves Equation (5) in an iterative manner. In the Expectation (E) step, the algorithm tries to approximate the expected values of $\boldsymbol{z}$ using the current model and data. Then, in the Maximization (M) step, the model tries to maximize the profit, i.e., minimize the loss, to obtain a better model. More specifically, for the k-th iteration, EM algorithm has the following formulations.

- E step: Estimate the missing values of SCADA data $\boldsymbol{z}^k$. Based on the definition of $\boldsymbol{z}$, the expected estimation of $\boldsymbol{z}^k$ can be written as:

$$\boldsymbol{z}^k = f(\boldsymbol{x}; W^k, B^k), \tag{6}$$

  where W^k and B^k represent the parameters of the DNN at the k-th iteration. The input data $\boldsymbol{x}$ comes from the intermediate PMU matrix $\tilde{\boldsymbol{X}}_p$ that corresponds to the hidden timestamps of the SCADA system. Then, the estimated values of $\boldsymbol{z}^k$ can be added to train the $(k+1)$-th DNN model, which improves the training with more accurate dynamic information in $\boldsymbol{z}^k$. Thus, the training process is the M step, shown as follows.
- M step: Retrain the DNN model to minimize the loss:

$$(W^{k+1}, B^{k+1}) = \arg \min_{W^{k+1}, B^{k+1}} L(W^k, B^k, \boldsymbol{z}^k, \boldsymbol{x}, \boldsymbol{y}). \tag{7}$$

  The obtained parameters W^{k+1} and B^{k+1} can formalize the $(k+1)$-th DNN model. Since $\boldsymbol{z}^k$ is known values, Equation (7) can be conveniently trained using SGD or Adam algorithms. All quantities of variable $\boldsymbol{z}^k$ can formulate the missing state matrix $\boldsymbol{Z}$. For different iterations, the values of $\boldsymbol{Z}$ can change. Thus, we also add the superscript and utilize $\boldsymbol{Z}^k$ to represent the predicted missing values in the k^{th} iteration, as shown in Fig. 3.

Finally, we summarize the complete algorithm for the Hd-Deep-EM model in Algorithm 1.

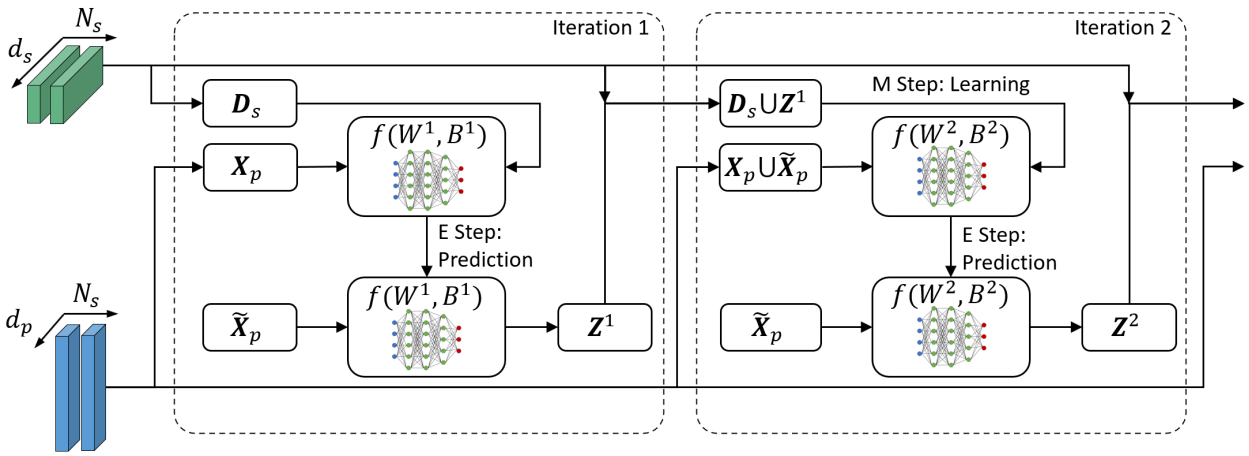

Fig. 3: Illustration of the computations for proposed Hd-Deep-EM method.

Algorithm 1 Hd-Deep-EM algorithm.

Input: PMU data matrix $\boldsymbol{X}_p$, SCADA data matrix $\boldsymbol{D}_s$, PMU data matrix for recovering data $\tilde{\boldsymbol{X}}_p$.
Hyper-parameters: number of iterations K and DNN-related parameters like batch size and learning rate.
Output: Missing data matrix $\boldsymbol{Z}$ for the dynamic states.

1: Initial parameters of the DNN f.
2: **for** $k = 1$ to K **do**
3: E step: Use Equation (6) to estimate the k-th dynamic states.
4: M step: Optimize Equation (7) to update parameters of the k-th DNN.
5: **end for**

D. Double Expectation Maximization Algorithms for Noisy Data

For realistic datasets, noise may lower the performance of the state estimation. To tackle noise data, we modify our Hd-Deep-EM algorithm for better estimation. Specifically, we can assume the noises are independent of measurements and can be easily modeled via a distribution model like GMM. Then, a natural idea is to decompose the model for estimating hidden states and noises. Since both data can be estimated via EM algorithms, we modify our Hd-Deep-EM and propose a Double Hd-Deep-EM algorithms for noisy data.

Specifically, in each iteration, we utilize the DNN model f to represent the mapping from input PMU data to the output. In the meantime, we propose to leverage another Gaussian model to estimate the noises, formulating the double EM algorithms (Fig 3). Mathematically, the double EM steps can be written as follows:

- E1 step: Estimate the missing values of SCADA data $\boldsymbol{z}^k$ using Equation (6).
- E2 step: Compute the noise data

$$\boldsymbol{n}^k = \boldsymbol{y} - f(\boldsymbol{x}; W^k, B^k). \tag{8}$$

- M1 step: Update the parameters of Gaussian model using $\boldsymbol{n}^k$. Then, obtain $\boldsymbol{n}^{k+1}$ using the Gaussian model.
- M2 step: Update parameters of DNN model f via:

$$\min_{W^{k+1}, B^{k+1}} L(W^k, B^k, \boldsymbol{z}^k, \boldsymbol{x}, \boldsymbol{y} - \boldsymbol{n}^{k+1}). \tag{9}$$

Note that for known SCADA data, we utilize $\boldsymbol{y} - \boldsymbol{n}^{k+1}$ to take place of $\boldsymbol{y}$ to achieve the denoising.

In general, the algorithm can be summarized as Algorithm 2. We denote our algorithm as the Double Hd-Deep-EM algorithm.

Algorithm 2 Double Hd-Deep-EM algorithm for noisy data.

Input: PMU data matrix $\boldsymbol{X}_p$, SCADA data matrix $\boldsymbol{D}_s$, PMU data matrix for recovering data $\tilde{\boldsymbol{X}}_p$.
Hyper-parameters: number of iterations K and DNN-related parameters like batch size and learning rate.
Output: Missing data matrix $\boldsymbol{Z}$ without noise for the dynamic states.

1: **for** $k = 1$ to K **do**
2: Initial parameters of the DNN f and the Gaussian model.
3: E step: Use Equations (6) and (8) to estimate the k-th dynamic states and the noise.
4: M step: Utilize the noise data to update the Gaussian model. Then, optimize Equation (9) to update parameters of the k-th DNN.
5: **end for**

E. Theoretical Support of Double Hd-Deep-EM Algorithm for Denoising

In the Double Hd-Deep-EM algorithm, we estimate parameters for both the DNN model and the underlying distribution of noises, which requires certain guarantees for the convergence to the true noise distribution, where the EM algorithm can provide the convergence theorem. Such convergence states the decreasing distance between the true likelihood of the noise

distribution and the estimated likelihood using the data in Equation (8). Specifically, we modify the convergence theorem from [29] and propose the following theorem.

***Theorem** 1: Let θ denote the parameters of the noise distribution. Then, let $\{\theta^t\}$ denote the parameter sequence in the EM iteration and t represents the iteration index. If θ^* is a limit point of $\{\theta^t\}$, then (1) θ^* is a stationary point of the likelihood function of the noise data in Equation* (8)*, and* (2) *the sequence $\{l(\theta^t)\}$ is non-decreasing and converges to $l(\theta^*)$, where $l(\theta)$ represents the likelihood of the noise distribution.*

The proofs of Theorem 1 can be seen in [29]. Basically, Theorem 1 justifies the convergent dynamics of the EM algorithm. More specifically, Theorem 1 proves that the convergent point θ^* exists for the parameters of the true distribution of the noises. Then, Theorem 1 illustrates the convergence to θ^* using the EM algorithm.

The convergence is further supported in the numerical validations where we investigate the proposed Hd-Deep-EM algorithm and the Double Hd-Deep-EM for noiseless and noisy data, respectively. The results demonstrate that our method has excellent denoising capacities under different noise levels.

IV. Experiments

A. Dataset Description

In the experiment, we employ synthetic datasets from 200- and 500- bus systems [33], [34]. For data preparation, we employ a commercial-grade simulator, Positive Sequence Load Flow (PSLF) to simulate PMU data with 60 samples per second. To diversify datasets, we vary the loading conditions during the simulation, which leads to different system dynamics. Moreover, we introduce three different fault events to our data set to validate robustness of our algorithm. As shown in Fig. 4, we visualize the event using Voltage Magnitude (VM) data obtained from PSLF. Further, we assume PMUs only locate at partial buses, with a penetration $\eta = 0.05$. For the rest data, we assume they are SCADA data with a sampling process. Then, we can obtain the SCADA data with 0.5 samples per second.

Totally, we have over 40 files of 10-second simulated data for each test system. To obtain $\boldsymbol{X}_p$ from these time series, we utilize the moving window with the length to be 0.33 seconds (i.e., $k = 20$ samples) to reformalize the training data. Thus, we have $d_p = 200 \times \eta = 10$; $d_s = 200 \times (1-\eta) = 190$ for $200-$bus test system and $d_p = 500 \times \eta = 25$; $d_s = 500 \times (1-\eta) = 475$ for $500-$bus test system. In general, we have $\boldsymbol{D}_s \in \mathbb{R}^{20\times 190}$ and $\boldsymbol{X}_p \in \mathbb{R}^{20\times(20\times 10)}$ for the 200-bus system and $\boldsymbol{D}_s \in \mathbb{R}^{20\times 475}$ and $\boldsymbol{X}_p \in \mathbb{R}^{20\times(20\times 25)}$ for the 500-bus system. They are matrices of PMUs and SCADA measurements data for one window. Further, we have $\boldsymbol{Z} \in \mathbb{R}^{580\times 190}$ $\tilde{\boldsymbol{X}} \in \mathbb{R}^{580\times 200}$ for the 200-bus system and $\boldsymbol{Z} \in \mathbb{R}^{580\times 475}$ $\tilde{\boldsymbol{X}} \in \mathbb{R}^{580\times 500}$ for the 500-bus system. They are total extracted matrices for missing SCADA data and PMU data to recover $\boldsymbol{Z}$.

B. Benchmark Method

To demonstrate the effectiveness of proposed model, we employ three different methods as benchmarks. The details of these methods are as follows.

- EM algorithm + Linear Regression (LR): LR fits a linear model with coefficients to minimize the error [35]. However, the LR model doesn't consider the spatial differences between PMU and SCADA data.
- EM algorithm + Vanilla Deep Neural Network: For this method, we also consider a Hd-Deep-EM framework. However, we utilize a vanilla DNN without the consideration of temporal correlations. Specifically, we set $k = 1$ for the window-based segmentation and don't consider the neighboring data for training. To distinguish our model and this method, we name this method as Hd-Deep-EM0 and our noiseless model as Hd-Deep-EM1. Further, we still name the noise version of Double Hd-Deep-EM model.
- Deep Neural Network (DNN) [36]: We utilize the same DNN in the Double Hd-Deep-EM to directly train the mapping without EM algorithm. The proposed DNN can effectively make use of spatial and temporal correlations. However, the scarce dynamic information of SCADA data may prevent the good training of the DNN.

In general, by comparing the testing accuracy of the proposed model and the benchmark models, we can evaluate the effectiveness of Hd-Deep-EM. As claimed in the proposed model, our Hd-Deep-EM can (1) capture the complex spatial correlations between PMU and SCADA data, (2) incorporate temporal correlations for a robust estimator, and (3) use the EM framework to iteratively incorporate the temporal dynamics of the SCADA data for better DNN training. Especially, we compare our Double Hd-Deep-EM with the LR+ EM to illustrate the high representational power of the DNN to capture spatial correlations. Then, we compare Double Hd-Deep-EM with Hd-Deep-EM0 to understand to impacts of considering temporal correlations to estimate the hidden states. Finally, we compare Double Hd-Deep-EM with DNN to illustrate the design of the EM framework.

During the testing, the hyper-parameters for all models are fine-tuned to achieve the best accuracy. Especially, we report the detailed results with respect to Mean Square Error (MSE) between the truth and predict $\boldsymbol{Z}$ matrix (i.e., missing dynamic states of the SCADA system) for all methods.

C. Deep Model Is Better Than Linear Model

In this subsection, we evaluate the effectiveness of our Hd-Deep-EM algorithm in tackling the spatial difference by comparing our Hd-Deep-EM algorithm to LR + EM. The results show that our model performs better than benchmarks. Specifically, we report the prediction performance of simulated data as follows.

Fig. 5 and Fig. 6 demonstrates the performances for the two methods in two different test system. The y-axis is the MSE error, and the x-axis represents the number of iterations during training. From the above figures, we find that our Hd-Deep-EM algorithm is lower than the MSE error of 0.05 and

This article has been accepted for publication in IEEE Transactions on Power Systems. 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/TPWRS.2023.3288005

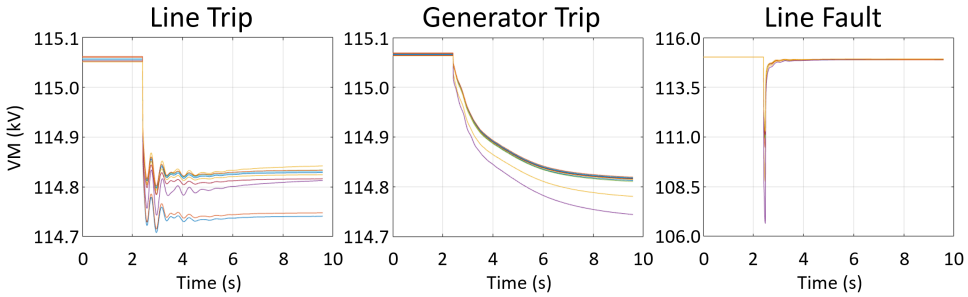

Fig. 4: The visualization of 10 PMUs' Voltage Magnitude (VM) measurements for three different events.

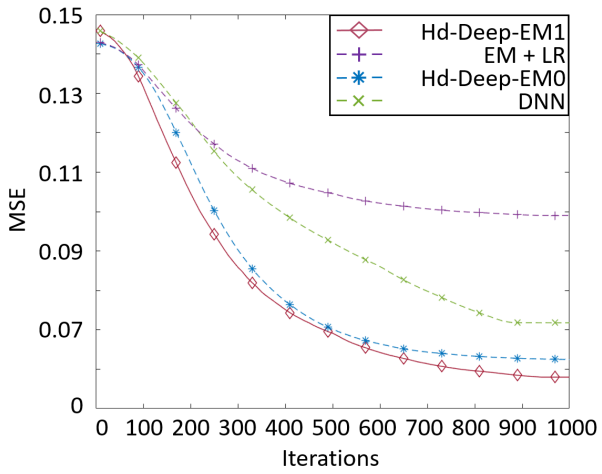

Fig. 5: Testing error for the 200-bus system.

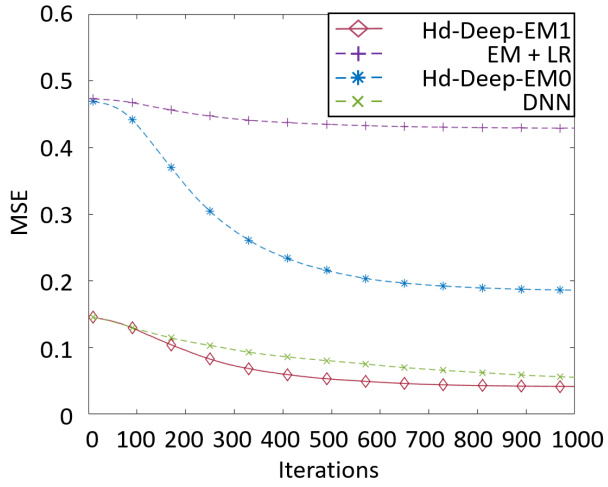

Fig. 6: Testing error for the 500-bus system.

0.4, compared to shallow EM+ LR in 200- and 500- bus test system after convergence. The better performance of the Hd-Deep-EM algorithm shows that the deep learning model can be better than the LR model in our complex scenario.

D. Consider Temporal Correlations Increase DNN Performance

In this subsection, we evaluate the effectiveness of our Hd-Deep-EM algorithm in integrating the temporal correlations. Specifically, we compare Hd-Deep-EM0 and Double Hd-Deep-EM. Due to the window-based segmentation, our Double Hd-Deep-EM can successfully integrate neighboring measurements in the temporal domain to estimate the dynamic states at one time slot. However, Hd-Deep-EM0 doesn't have this treatment. Then, we report the results of simulated data as follows.

Fig. 5 and Fig. 6 demonstrate the performances for the two methods. From the above figures, we find that our Double Hd-Deep-EM has an average of 0.05 and 0.06 testing MSE error for both test systems after convergence. In comparison, Hd-Deep-EM0 has an error of 0.07 and 0.2. The performance of these two methods can be explained as follow. Window segmentation can capture the data before and after the SCADA time stamp, which can benefit from finding the temporal trending of the PMU data and avoiding the negative impacts of the outliers. The better performance of the Hd-Deep-EM algorithm with window segmented data shows that our proposed deep learning model can easily capture the temporal trend for PMU data in our complex scenario. Finally, we observe that in Fig. 6, DNN model can perform better than Hd-Deep-EM0 that doesn't consider the temporal correlations. This verifies our claims that considering proper temporal correlations in the DNN and Hd-Deep-EM1 (noiseless) can improve the performance of estimating dynamic hidden states.

E. EM Procedure Provides Better Training of the DNN

In this subsection, we evaluate the effectiveness of introducing the EM procedure by comparing Hd-Deep-EM1 and a DNN model. Specifically, we report the results of simulated data as follows. Fig. 5 and Fig. 6 demonstrate the performances for the two methods. We find that our Double Hd-Deep-EM has an average testing error reduction of 0.01 and 0.02, compared to the DNN method. The reasons, when training the DNN model, the output SCADA data has limited dynamic information. Thus, direct training can't guarantee the DNN model performance to predict dynamic states. On the other hand, our Hd-Deep-EM can iterative predict the dynamic states and reuse the predicted results for training, leading to a better training procedure to incorporate dynamic information.

F. Double Hd-Deep-EM Provides Better Performance in Tackling Noise

In this subsection, we evaluate the effectiveness of interpolating noisy data by comparing Double Hd-Deep-EM among

This article has been accepted for publication in IEEE Transactions on Power Systems. 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/TPWRS.2023.3288005

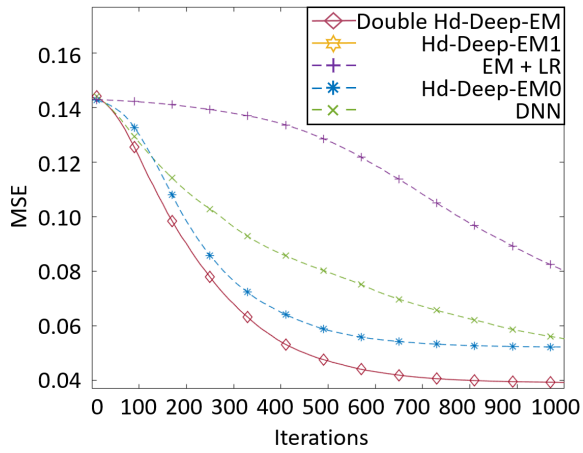

Fig. 7: Testing error for the 200-bus system with noisy data.

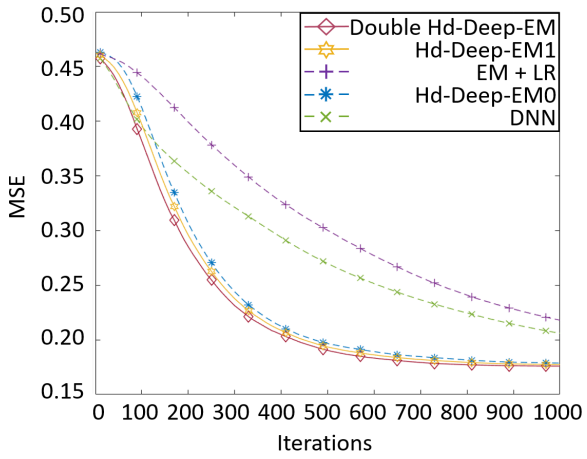

Fig. 8: Testing error for the 500-bus system with noisy data.

all three benchmark models and our Hd-Deep-EM1 method. In the experiment, we introduce noise into our dataset to validate the effectiveness in tackling noisy data. Specifically, we report the results of simulated data as follows. As shown in Fig. 7 and Fig. 8, we find that our Double Hd-Deep-EM has an average testing error reduction of 0.04, 0.02, 0.02 and 0.01, compared to EM+LR, Hd-Deep-EM0, DNN method and our proposed Hd-Deep-EM1 for the 200 bus system. Furthermore, Double Hd-Deep-EM has an average testing error reduction of 0.01, 0.02, 0.03 and 0.005, compared to bench mark methods and Hd-Deep-EM1 for the 500 bus system. The results show that our Double Hd-Deep-EM has the best performance of robustness to noisy data.

G. Sensitivity Analysis with respect to Noise Levels

Moreover, we further evaluate Double Hd-Deep-EM performance with other benchmarks models in 3 different level of noise injection to the simulated data. To quantify the level of the noise, we utilize the Signal-to-Noise Ratio (SNR) which compares the level of a desired signal to the level of noise. Signal-to-Noise Ratio can be defined as the ratio of signal power to noise power. The result of MSE error is shown in the tables below.

TABLE I: The MSE for different data interpolation methods in different noise level for Illinois 200-bus systems.

SNR Ratio (dB)	Double Hd-Deep-EM	Hd-Deep-EM1	EM+ LR	Deep EM0	DNN
25	**0.03914**	0.04874	0.08002	0.05214	0.05513
15	**0.03923**	0.05045	0.08204	0.05426	0.05525
10	**0.04035**	0.05765	0.08234	0.05987	0.05947

TABLE II: The MSE for different data interpolation methods in different noise level for South Carolina 500-bus systems.

SNR Ratio (dB)	Double Hd-Deep-EM	Deep EM1	EM+ LR	Deep EM0	DNN
25	**0.17596**	0.17723	0.21794	0.17873	0.20601
15	**0.17601**	0.17983	0.21819	0.18073	0.20730
10	**0.17616**	0.18177	0.22056	0.18395	0.21049

According to the performance in Table I and Table II, we find that our Double Hd-Deep-EM has an average testing error reduction over different noise level of 0.013, 0.017, 0.014 and 0.042, compared to Hd-Deep-EM1, EM+LR, Hd-Deep-EM0 and DNN method for 200 bus system. And Double Hd-Deep-EM has an average testing error reduction over different noise level of 0.004, 0.031, 0.005 and 0.043, compared to bench mark methods for 500 bus system.

H. Event Identification Accuracy Comparison with Different Dataset

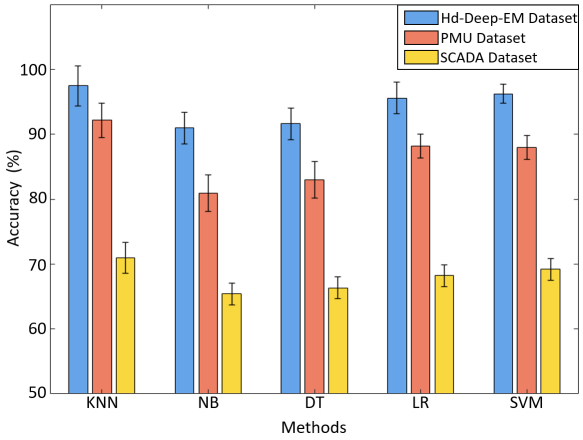

Fig. 9: Event identification for different datasets.

To evaluate the effectiveness of interpolation, we propose to compare the interpolated SCADA data, original SCADA data and PMU data for the data-driven task: identifying system events. Specifically, the same event identification algorithm is performed among the three datasets. We utilize the 5 different popular Supervised Learning methods to evaluate the results. The selected methods are K Nearest Neighbor (KNN), Naive Bayes (NB), Decision Tree (DT), Logistic Regression (LR),

and Support Vector Machine (SVM). As shown in Fig. 9, we find that our interpolated data can best improve the different learning methods, with KNN the best. Averagely, our merged dataset has average increase of 7.92%, and 36.35%, compared to the original PMU and SCADA datasets.

V. Conclusion

The increasing integration of renewable energy brings risks of system dynamic events. To achieve better monitoring of system dynamics, Phasor Measurement Units (PMUs) are placed to yield high-resolution data files. However, PMU units are limited in a system due to the high cost. For regions without PMUs, we propose to estimate the dynamic states using PMU and SCADA data. The problem is challenging due to the spatial differences between PMU and SCADA data and the temporal scarcity of the SCADA measurements. To solve these issues, we employ Deep Neural Networks (DNNs) to capture the spatial-temporal correlations efficiently. The DNN can be trained to map from PMU data to SCADA data, bringing a reasonable estimate of the missing measurements of SCADA data. However, the lack of dynamic information in the SCADA data disables a well-trained DNN.

Thus, we propose to utilize the estimated data to retrain the DNN model and build the complete algorithm in an iterative approach of estimation and retraining. Such a process can be explained under the framework of Expectation-Maximization (EM) algorithm. Thus, we name our method Hd-Deep-EM. Subsequently, we improve our Hd-Deep-EM in realistic applications by considering the noises. Specifically, we propose to utilize another EM algorithm to estimate the distribution of the noise data, bringing a Double Hd-Deep-EM algorithm. To validate the Hd-Deep-EM model, we utilize diversified synthetic and real-world datasets. All results show that Hd-Deep-EM and Double Hd-Deep-EM perform better than other existing methods.

References

[1] "IEEE Standard for Synchrophasor Measurements for Power Systems," *IEEE Std C37.118.1-2011 (Revision of IEEE Std C37.118-2005)*, pp. 1–61, 2011.

[2] A. Simões Costa, A. Albuquerque, and D. Bez, "An estimation fusion method for including phasor measurements into power system real-time modeling," *IEEE Transactions on Power Systems*, vol. 28, no. 2, pp. 1910–1920, 2013.

[3] M. Ghosal and V. Rao, "Fusion of pmu and scada data for dynamic state estimation of power system," in *IEEE North American Power Symposium*, 2015, pp. 1–6.

[4] A. S. Costa, A. Albuquerque, and D. Bez, "An estimation fusion method for including phasor measurements into power system real-time modeling," *IEEE Transactions on Power Systems*, vol. 28, no. 2, pp. 1910–1920, 2013.

[5] A. Albuquerque, D. Bez, and A. Simões, "Multistage strategies to incorporate phasor measurements into power system state estimation," in *IREP Symposium Bulk Power System Dynamics and Control - IX Optimization, Security and Control of the Emerging Power Grid*, 2013, pp. 1–11.

[6] K. Das, J. Hazra, D. P. Seetharam, R. K. Reddi, and A. K. Sinha, "Real-time hybrid state estimation incorporating scada and pmu measurements," in *IEEE PES Innovative Smart Grid Technologies Europe*, 2012, pp. 1–8.

[7] D. Simon, *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. John Wiley & Sons, 2006.

[8] J. Zhao, A. Gómez-Expósito, M. Netto, L. Mili, A. Abur, V. Terzija, I. Kamwa, B. Pal, A. K. Singh, J. Qi *et al.*, "Power system dynamic state estimation: Motivations, definitions, methodologies, and future work," *IEEE Transactions on Power Systems*, vol. 34, no. 4, pp. 3188–3198, 2019.

[9] Y. Weng, R. Negi, C. Faloutsos, and M. D. Ilić, "Robust data-driven state estimation for smart grid," *IEEE Transactions on Smart Grid*, vol. 8, no. 4, pp. 1956–1967, 2016.

[10] Y. Weng, R. Negi, and M. D. Ilić, "Historical data-driven state estimation for electric power systems," in *IEEE International Conference on Smart Grid Communications*, 2013, pp. 97–102.

[11] ——, "Probabilistic joint state estimation for operational planning," *Transactions on Smart Grid*, vol. 10, no. 1, pp. 601–612, 2017.

[12] Q. Song and M. Shepperd, "Missing data imputation techniques," *International journal of business intelligence and data mining*, vol. 2, no. 3, p. 261–291, 2007.

[13] G. E. Batista and M. C. Monard, "An analysis of four missing data treatment methods for supervised learning," *Applied artificial intelligence*, vol. 17, no. 5-6, pp. 519–533, 2003.

[14] N. T. Le and W. Benjapolakul, "A data imputation model in phasor measurement units based on bagged averaging of multiple linear regression," *IEEE Access*, vol. 6, pp. 39 324–39 333, 2018.

[15] S. J. Plathottam, H. Salehfar, and P. Ranganathan, "Convolutional Neural Networks (CNNs) for power system big data analysis," in *IEEE North American Power Symposium*, 2017, pp. 1–6.

[16] A. Gupta, G. Gurrala, and P. Sastry, "An online power system stability monitoring system using convolutional neural networks," *IEEE Transactions on Power Systems*, vol. 34, no. 2, pp. 864–872, 2018.

[17] C. Kim, K. Kim, P. Balaprakash, and M. Anitescu, "Graph convolutional neural networks for optimal load shedding under line contingency," in *IEEE Power & Energy Society General Meeting*, 2019, pp. 1–5.

[18] J. Vermaak and E. Botha, "Recurrent neural networks for short-term load forecasting," *IEEE Transactions on Power Systems*, vol. 13, no. 1, pp. 126–132, 1998.

[19] A. Ayad, H. E. Farag, A. Youssef, and E. F. El-Saadany, "Detection of false data injection attacks in smart grids using recurrent neural networks," in *IEEE Power & Energy Society Innovative Smart Grid Technologies Conference*, 2018, pp. 1–5.

[20] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for unconstrained handwriting recognition," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 31, no. 5, pp. 855–868, 2008.

[21] Y. Zhang, J. Sun, J. Xue, G. Y. Li, and Z. Xu, "Deep expectation-maximization for joint mimo channel estimation and signal detection," *IEEE Transactions on Signal Processing*, vol. 70, pp. 4483–4497, 2022.

[22] M. Shao, W.-K. Ma, J. Liu, and Z. Huang, "Accelerated and deep expectation maximization for one-bit mimo-ofdm detection," *arXiv preprint arXiv:2210.03888*, 2023.

[23] M. Jiang, Y. Ding, B. Goertzel, Z. Huang, C. Zhou, and F. Chao, "Improving machine vision via incorporating expectation-maximization into deep spatio-temporal learning," in *International Joint Conference on Neural Networks*, 2014, pp. 1804–1811.

[24] A. V. Pankratov, N. L. Batseva, E. S. Polyakova, A. S. Tavlintsev, I. L. Lapatin, and I. Y. Lipnitskiy, "Application of Expectation Maximization Algorithm for Measurement-based Power System Load Modeling," in *International Siberian Conference on Control and Communications*, 2019, pp. 1–5.

[25] Y. Xiang, T. Wang, and Z. Wang, "Improved gaussian mixture model based probabilistic power flow of wind integrated power system," in *IEEE Power & Energy Society General Meeting*, 2019, pp. 1–5.

[26] A. Ganjavi, E. Christopher, C. M. Johnson, and J. Clare, "A study on probability of distribution loads based on expectation maximization algorithm," in *IEEE Power Energy Society Innovative Smart Grid Technologies Conference*, 2017, pp. 1–5.

[27] D. Lee and D. Kundur, "Cyber attack detection in pmu measurements via the expectation-maximization algorithm," in *IEEE Global Conference on Signal and Information Processing*, 2014, pp. 223–227.

[28] 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, 2018.

[29] C. J. Wu, "On the convergence properties of the EM algorithm," *The Annals of statistics*, pp. 95–103, 1983.

[30] A. F. Agarap, "Deep Learning Using Rectified Linear Units (ReLU)," *arXiv preprint arXiv:1803.08375*, 2023.

[31] L. Bottou, "Stochastic gradient descent tricks," *Neural Networks: Tricks of the Trade: Second Edition*, pp. 421–436, 2012.

[32] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2023.

[33] Engineering Texas A&M University, "Illinois 200-bus system: ACTIVSg200," 2016. [Online]. Available: https://electricgrids.engr.tamu.edu/electric-grid-test-cases/activsg200/

[34] ——, "SouthCarolina 500-bus system: ACTIVSg500," 2016. [Online]. Available: https://electricgrids.engr.tamu.edu/electric-grid-test-cases/activsg500/

[35] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[36] C.-Y. Cheng, W.-L. Tseng, C.-F. Chang, C.-H. Chang, and S. S.-F. Gau, "A deep learning approach for missing data imputation of rating scales assessing attention-deficit hyperactivity disorder," *Frontiers in psychiatry*, vol. 11, p. 673, 2020.