Summer 8-2020

Learning Deep Architectures for Power Systems Operation and Analysis

Mahdi Khodayar
mahdik@smu.edu

LEARNING DEEP ARCHITECTURES FOR POWER SYSTEMS OPERATION AND ANALYSIS

Approved by:

DocuSigned by:
Prof. Jianhui Wang
3EB51DA799FC423...

Dr. Jianhui Wang
Professor

DocuSigned by:
Prof. Behrouz Peikari
0B23CAE020EB426...

Dr. Behrouz Peikari
Professor

DocuSigned by:
Prof. Khaled Abdelghany
65246D663543421...

Dr. Khaled Abdelghany
Professor

DocuSigned by:
Prof. Harsha Gangammanavar
72434E71C69A407...

Dr. Harsha Gangammanavar
Assitant Professor

DocuSigned by:
Prof. Feng Chen
AE4904D5DA3E47E...

Dr. Feng Chen
Associate Professor

LEARNING DEEP ARCHITECTURES FOR POWER SYSTEMS OPERATION AND ANALYSIS

A Dissertation Presented to the Graduate Faculty of the

Bobby B. Lyle School of Engineering

Southern Methodist University

in

Partial Fulfillment of the Requirements

for the degree of

Doctor of Philosophy

with a

Major in Electrical Engineering

by

Mahdi Khodayar

Ph.D., Electrical Engineering, Southern Methodist University
M.Sc., Artificial Intelligence, Khajeh Nasir Toosi University of Technology
B.Sc., Software Engineering, Khajeh Nasir Toosi University of Technology

August 4, 2020

ACKNOWLEDGMENTS

First of all, I would like to express my deepest gratitude to my adviser, Prof. Jianhui Wang for his invaluable guidance during my project. His high standard way of thinking and doing research shaped me and made me grown. I would like to equally thank Dr. Khaled Abdelghany at the Civil and Environmental Engineering Department of SMU as well as Dr. Feng Chen at the Department of Computer Science in University of Texas at Dallas for their valuable inputs and guidance during my research on machine learning and deep neural architectures. I would also like to thank Dr. Behrouz Peikari and Dr. Harsha Gangammanavar for their effort on reviewing my work and giving valuable comments on the mathematical and experimental contents of this thesis.

Mahdi Khodayar

Mahdi Khodayar,

Ph.D., Electrical Engineering, Southern Methodist University
M.Sc., Artificial Intelligence, Khajeh Nasir Toosi University of Technology
B.Sc., Software Engineering, Khajeh Nasir Toosi University of Technology

Learning Deep Architectures for Power SystemsOperation and Analysis

Advisor: Dr. Jianhui Wang
Doctor of Philosophy degree conferred August 4, 2020
Dissertation completed May 5, 2020

With the rapid increase in size and computational complexities of power systems, the need for powerful computational models to capture strong patterns from energy datasets is emerged. In this thesis, we provide a comprehensive review on recent advances in deep neural architectures that lead to significant improvements in classification and regression problems in the area of power engineering. Furthermore, we introduce our novel deep learning methodologies proposed for a large variety of applications in this area. First, we present the interval deep probabilistic modeling for wind speed forecasting. Incorporating the Rough Set Theory into deep neural networks, we create an accurate interval model for point prediction of intermittent wind speed datasets. Then, we develop a graph convolutional neural network for the spatiotemporal prediction of wind speed values in multiple neighboring wind sites. Our provided numerical results show the great improvement of prediction accuracy compared to classic deep learning. Using the concept of graph convolutions, we also develop a new conditional graph variational autoencoder to learn the probability density of future solar irradiance given the historical solar irradiance of multiple photovoltaic energy sites. This study led to the state-of-the-art performance in probabilistic solar prediction in power systems domain. Moreover, we introduced a novel multimodal deep recurrent structure that makes use of both system-wide power and voltage measurements as well as load parameters for accurate real-time load modeling. The numerical results show the significant improvement of this method compared to classic deep learning in estimating dynamic load parameters of smart grids. Moreover, we develop deep dictionary learning as a new paradigm in machine learning for energy disaggregation and behind-the-meter net load decomposition. The presented work leads to the best accuracy in comparison with recent sparse coding and dictionary learning-based decomposition methods in the literature. Finally, a novel deep generative model is introduced to learn the probability density of the measurements on the nodes and edges of a power grid. Using this model, we take a large number of samples from the probability distribution of the structure of power systems, hence, generating synthetic power networks with the same topological and physical behaviors as the original power system. Our simulation results on real-world datasets show the great improvements of the proposed approach compared to the data-driven approaches in the recent literature.

TABLE OF CONTENTS

CHAPTER

List of Figures

List of Tables

Chapter 1

Introduction

With the rapid growth of power systems measurements in terms of size and complexity, discovering statistical patterns for a large variety of real-world applications such as renewable energy prediction, demand response, energy disaggregation, and state estimation is considered a crucial challenge. In recent years, deep learning has emerged as a novel class of machine learning algorithms that represents power systems data via a large hypothesis space that leads to the state-of-the-art performance compared to most recent data-driven algorithms. This study explores the theoretical advantages of deep representation learning in power systems research. We review deep learning methodologies presented and applied in a wide range of supervised, unsupervised, and semi-supervised applications as well as reinforcement learning tasks. We discuss various settings of problems solved by discriminative deep models including Stacked Autoencoders and Convolutional Neural Networks as well as generative deep architectures such as Deep Belief Networks and Vatriational Autoencoders. The theoretical and experimental analysis of deep neural networks in this study motivates long-term research on optimizing this cutting-edge class of models to achieve significant improvements in the future power systems research.

1.1. Literature Review

The reliability and accuracy of data-driven models in power systems operation and analysis closely rely on the selection of data representation (i.e., features extracted from the underlying data) [17]. As a result, most of the concerns regarding the application of classic data-driven models in power systems is focused on the design of preprocessing techniques using unsupervised dimensionality reduction algorithms including the principal component analysis (PCA) [26], linear discriminant analysis (LDA) [99], and t-distributed stochastic neighbor embedding (t-SNE) [252]. Such feature extraction techniques dramatically increase the time and memory complexity of data-driven algorithms and lead to insufficient accuracy as they mainly cannot capture highly nonlinear and highly varying patterns inside the ambient space of the data [17].

Recent machine learning studies on wind forecasting [111, 115, 144, 148], photovoltaic (PV) power prediction [86,112,200,246], state estimation [162,220], power grid synthesis [117], and energy disaggregation [74, 95, 118] show that developing data-driven models with less dependencies on explicit preprocessing methods (e.g., PCA) leads to dramatically better performance in terms of classification and regression accuracy. Instead of having an explicit preprocessing approach, the deep learning studies form a composition of multiple nonlinear latent layers in a multi-layer artificial neural network (ANN). The ANN parameters (i.e., weights and biases) are generally trained in a greedy unsupervised layer-by-layer fashion [184], where each layer performs a nonlinear feature extraction on the features computed by its previous layer.

Based on the theoretical aspects, deep learning algorithms proposed in power engineering applications are generally categorized into three major classes:

1) Discriminative deep ANNs aim to directly learn a highly nonlinear decision boundary between different classes and regression regions of the power system data [108]. In this category, the Rectified Linear Unit (ReLU) ANN [59] is presented for real-time reliability management response. Due to its high generalization capacity and low computational complexity, the ReLU ANN is also utilized for online small signal stability assessment [27], faulted line localization [141], and phasor measurement unit (PMU) based event classification [121]. Moreover, the Stacked Autoencoder (SAE) is developed as a highly nonlinear version of the PCA for unsupervised pattern recognition for wind energy prediction [32, 111, 239], PV power forecasting [67], fault diagnosis [238], and transient stability assessment [204]. In addition, the Long Short-term Memory (LSTM) ANN is presented as a supervised temporal feature extractor with a deep recurrent formulation to model the sequential behavior of the time-dependent power systems measurements [118, 264]. In this area, LSTM-based sequential models are proposed for wind and PV power forecasting [52, 202, 263, 264], load modeling using system-wide measurements [46], real-time power fluctuation identification [230], power demand forecasting [205], energy disaggregation [118], reneasble energy pridiction [34, 52], as well as fault detection [251].

Convolutional Neural Network (CNN) is another major class of discriminative models that are powerful to capture coherent structures in power system measurements due to their convolutional and pooling operations [189]. Learning statistical convolution filters, the CNN extracts strong correlations between data points in both space and time domains [192]. The mixture of convolutional and pooling layers in this type of deep neural networks incorporates the spatial charactristics of measurements into their temporal features to solve spatiotemporal tasks in the area of renewable energy forecasting [112, 160], transient stability analysis [192], harmonic components analysis [189], fault detection [30], and short-term voltage stability assessment [240].

2) Probabilistic deep ANNs consider feature learning as a procedure to find a parsimonious set of hidden variables that best describe the probability density function (PDF) of the data. The PDF is further mapped to the target class/value of the problem. In this group, the Deep Belief Network (DBN) is a well-known probabilistic graphical model that learns the PDF of the data given its conditionally independent latent features. The features are learned by Gibbs sampling in order to provide an accurate estimation of the probabilistic behavior of the input data for probabilistic applications that need to address large uncertainty factors in the data. DBN is mainly applied to wind and solar power prediction [221, 235], transient stability assessment [261], day-ahead and week-ahead load prediction [75], as well as probabilistic state estimation [90]. Moreover, in this category of models, the Generative Adversarial Network (GAN) is presented that takes samples from an estimated PDF and compares the generated samples with the actual data in the dataset to increase the accuracy of the learned PDF. As this model can efficiently learn the major characteristics of the PDF, it is recently introduced to important outlier and fault detection problems for small-sample wind turbines [146] and smart grid cyber attack detection [6]. Furthermore, since GANs can synthesize the data by taking samples from the estimated PDF, these models are recently employed for model-free renewable scenario generation problems [37]. In this line of research, the Variational Autoencoders (VAEs) are presented as a novel version of deep generative ANNs that learn the PDF of the data by learning a high dimensional latent variable which is mapped to the original data samples in the dataset. VAE is shown to estimate accurate synthetic samples for power grid synthetic [119], unsupervised anomaly detection in energy time series [175, 262], and

Electric Vehicle load generation [170].

3) Deep Reinforcement Learning (DRL) algorithms are a major class of machine learning approaches that seek to learn an optimal policy based on the feedback from the environment computed by a reward function. This function reflects how much the problem's objective is satisfied based on the current state of the system. In contrast to the conventional deep learning that merely estimates a discrete target function for classification and continuous target funtion for regression, DRL aims to decline a general error function defined by the experience in a fully observable or partially observable environment. Hence, this method solves more general classes of problems compared to the classic deep learning. Due to its feedback-based nature, DRL is widely employed for control problems including voltage control [58], adaptive emergency control [88], as well as self-learning control for energy efficient transportation [180]. Also, DRL is applied to optimization problems for learning the optimal bidding strategies in electricity markets [236, 244], demand response strategies for energy management [89, 212, 242], as well as finding the optimal wind and storage cooperative schedule to decrease the effect of the uncertainty in renewable generation in smart grids [181]. Moreover, this class of methodologies are recently introduced to cyber attack detection and recovery [227], dynamic power allocation [165], and power system data integrity defense [9].

This chapter reviews the three major categories of deep neural networks in the domain of power systems research. First, the deep discriminative appraoch is introduced in Section 1.2. Various variations of this machine learning class of models is explained, and compared both mathematically and experimentally using several real world power systems datasets. Section 1.3 introduces probabilistic deep learning methods such as the classic DBN and its Gaussian variation as well as the recently proposed GANs and VAEs. The applications and theoretical advantages of these techniques are discussed in this section. Then, in Section 1.4, the chapter reviews DRL algorithms and their vast area of applications in power systems optimization and control. Finally, the conclusions are provided in Section 1.5.

1.2. Discriminative Deep Learning

Discriminative modeling is one of the major areas in machine learning that tends to estimate a function f_θ parameterized by $\theta \in \mathbb{R}^p$ that directly maps an input to the true output of the problem. Let us Consider a training dataset $D_{tr} = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$ that contains n training samples (x_i, y_i) with input x_i corresponding to the true output/label y_i, and a test dataset $D_{ts} = \{(x_{n+1}, y_{n+1}), (x_{n+2}, y_{n+2}), ..., (x_{n+m}, y_{n+m})\}$ with m unobserved test samples. The goal is to learn the optimal parameter θ^* where the average distance between $f_{\theta^*}(x)$ and y is the lowest for all samples $(x, y) \in D_{tr}$. The test error is the average error between the trained $f_{\theta^*}(x)$ and y for all $(x, y) \in D_{ts}$.

To obtain a nonlinear mapping between the inputs and outputs, the classic Multilayer Perceptron (MLP) defines an input layer $h^0 \in \mathbb{R}^{d_0}$ and L computational layers $\{h^1, h^2, ..., h^L\}$ where each layer $h^i \in \mathbb{R}^{d_i}$ $(i \in [1, L])$ is a nonlinear function of previous layer defined by $h^i = g^i(W^i h^{i-1} + b^i)$ where g^i is a nonlinear transformation function usually computed by a sigmoid or hyperbolic tangent function, $W^i \in \mathbb{R}^{d_i} \times \mathbb{R}^{d_{i-1}}$ is the weight matrix and $b^i \in \mathbb{R}^{d_i}$ is the bias of the activation function in layer h^i. Using the hidden layers, the MLP provides a nonlinear transformation between the input $h^0 = x$ and output $h^L = y$ in the dataset.

To train each layer h^i, the gradient descent (GD) method moves parameters W^i and b^i in the opposite direction of the gradient of the training error with respect to W^i and b^i, respectively. As the gradients dramatically decline with the increase in L, there is a trade-off between the number of computational layers L and the strength of GD to update the model. As L becomes larger to address more complex problems, GD becomes ineffective due to the vanishing gradients. Hence, the classic MLP does not provide sufficient generalization capability to accurately solve complex real-world problems. As a result, discriminative deep learning is proposed to efficiently train deep ANNs with $L > 1$ in order to have a high capacity mapping f_θ while providing an effective training procedure to update the parameters.

1.2.1. Rectified Linear Unit ANN

ReLU ANN defines a rectified linear unit activation function $ReLU(x) = max(0, x)$ at the computational layers of MLP rather than using the classic nonlinear activation functions. Since the gradient of $ReLU(x)$ with respect to a positive input x is always 1 regardless of x, this function solves the vanishing gradient problem of the MLP. Hence, this model is applied to power systems applications that require highly nonlinear feature extraction.

Table 1.1 summarizes the applications of discriminative modeling in the power systems area. As shown in this table, a ReLU ANN is implemented in [59] to estimate the cost of real-time resource allocations decisions in operation planning of the modified IEEE-RTS96 single area network [72]. Also, in [27], various ReLU ANN architectures are trained to learn the small signal stability assessment of the classic 16-machine 68-bus test system [195]. As shown in [27], when the number of layers increase from 2 to 6, the assessment accuracy is significantly increased since the ReLU ANN's hypothesis space becomes largers. In addition, the ReLU ANN is applied to real-time faulted line localization in IEEE 39-bus and 68-bus power systems which resulted in 98% and 93% location accuracy rate for line to ground and double line to ground faults, respectively. Furthermore, in [121], ReLU ANNs are shown to yield 98.17% accuracy for the classification of 6 events including generation loss, load loss, as well as line to ground faults in the IEEE 68-bus system.

1.2.2. Stacked Autoencoder

To train a deep ANN with input h^0 and L computational layers h^i $(i = 1, 2, ..., L)$, the SAE trains L AEs $\{AE^i\}_{i=1}^{L}$. Each AE^i is a MLP ANN with one hidden layer with an encoding activation function f_{enc} where a high-dimensional input $h^{i-1} \in R^{d_{i-1}}$ is encoded into a lower dimensional latent feature vector $h^i = f_{enc}(h^{i-1}) \in R^{d_i}$ which is further mapped back (decoded) to the original input h^{i-1} in the output layer $o^i = f_{dec}(h^i)$ using the decoding function f_{dec}. Hence, the GD error of AE^i is computed by $||o^i - h^{i-1}||_2^2$ to train the weight W_{enc}^i and bias b_{enc}^i of its encoding layer as well as the weight W_{dec}^i and bias b_{dec}^i of its decoder. To update the parameters of the SAE, starting from $i = 1$, each AE^i is trained and the trained encoder parameters W_{enc}^i and b_{enc}^i are used to initialize W^i and b^i of the layer i, respectively. Finally, the whole SAE ANN is trained using GD on the training data D_{tr}.

Due to the unsupervised feature learning at each AE, the SAE model is suitable for situations where the training data is limited or contains remarkable uncertainty and noise factors. Hence, this method respectively outperforms the MLP, Nonlinear Autoregressive Exogenous (NARX) ANN, and Time Delay ANN (TDANN) by 23.66%, 21.54%, and 14.81% in terms of the Mean Absolute Percentage Error (MAPE) for short-term wind speed prediction [32, 111, 239]. Moreoever, as shown in Table 1.1, the SAE outperforms ReLU in both classification tasks (e.g., stability assessment [27] and PMU event classification [121]) as well as regression tasks with large data variations (e.g., wind and PV power prediction [67, 239] and load forecasting [205]). Furthermore, due to its powerful greedy layer-wise training process, the SAE yields an average transformer fault diagnosis accuracy of 95.4% in the IEC 60599 and IEC TC 10 databases [60]. In addition, SAE improves the transient stability analysis accuracy of extreme learning machines (ELMs) by 6.59% in the IEEE 39-bus system [204].

1.2.3. Long Short-Term Memory Network

LSTM is a widely used deep recurrent ANN that extracts powerful temporal features from a time series $x_1, x_2, ..., x_T$. At each time step, $0 \leq t \leq T$, LSTM observes a sample x_t and updates its temporal memory C^t that describes the state of the time series at t, and produces a temporal feature vector h^t that summarizes LSTM's temporal information after the observation x_t. The recursive structure of LSTM features is defined by:

$$\begin{aligned}
i_t &= \sigma(W_i.[h_{t-1}, x_t] + b_i) \\
f_t &= \sigma(W_f.[h_{t-1}, x_t] + b_f) \\
o_t &= \sigma(W_o.[h_{t-1}, x_t] + b_o) \\
\tilde{C}_t &= tanh(W_C.[h_{t-1}, x_t] + b_C) \\
C_t &= f_t * C_{t-1} + i_t * \tilde{C}_t \\
h_t &= o_t * tanh(C_t)
\end{aligned} \tag{1.1}$$

where i_t is the input gate that decides the magnitude of information flow into the time-dependent memory C_t using the sigmoid activation σ with weight W_i and bias b_i. f_t is the forget gate that determines how much information needs to be removed from C_t using weight W_f and bias b_f. o_t is the LSTM's output at time t using weight W_o and bias b_o while h_t is the extracted tempoal feature at time t. At each time step t, the memory is updated by $\tilde{C}_t$ as a nonlinear function parameterized by W_C and b_C.

In contrast to the classic recurrent MLPs, the LSTM does not encounter the vanishing gradient problem; hence, can be efficiently trained using GD. As a result, as shown in Table 1.1, this method is applied to a large variety of time-dependent applications such as wind, PV, and load prediction [52, 67, 75, 239] as well as load modeling [46] and power fluctuation identification [230]. As Table 1.1 shows, the LSTM generally outperforms both ReLU and SAE in the domain of time-dependent applications due to its recurrent structure and powerful temporal memory. In [264], a novel attention mechanism-based LSTM is developed to improve the hourly solar energy prediction of MLP by 6.17% and 0.27 in terms of MAPE and Root Mean Squared Error (RMSE), respectively. Also, the LSTMs in [263] and [202] have shown the state-of-the-art performance in wind prediction tasks. Moreover, in [46], a LSTM is defined in a multimodal neural architecture to simultaneously capture the temporal characteristics of dynamic load parameters as well as the voltage and power changes in the IEEE 68-bus test system [195]. It is shown that the LSTM captures real-time dynamic behaviors of load parameters with 38.42% and 25.64% better RMSE and MAPE, respectively, compared to the TDNN method due to its larger hypothesis space and overcoming the overfitting problem. Similar accuracy improvements are recently reported in other time-dependent applications including power fluctuation identification [230], data-based line trip fault prediction [251], and industrial load forecasting [205].

1.2.4. Convolutional Neural Network

CNNs contain a two dimensional input layer I, a set of hidden convolution and pooling layers, and a fully connected output layer. Each neuron in the convolution layer is a nonlinear kernel that divides the input into small slices called receptive fields. The output of convolution operation at the k-th kernel in the l-th convolution layer is computed by:

$$f_l^k(p,q) = \sum_c \sum_{x,y} i_c(x,y).e_l^k(u,v) \tag{1.2}$$

where $i_c(x,y)$ is the (x,y) element of the c-th channel of input I, and $e_l^k(u,v)$ is the (u,v) element of the k-th kernel of layer l. The pooling layer sweeps an average or maximum function over small patches of the convolution output in (1.2) to further reduce the dimension of the extracted features which enhances the sparsity of the kernel parameters and avoids overfitting on the training set. Finally, the fully connected layer maps the extracted features to the target label of the underlying classification or regression task.

As the convolution and pooling layers process their local input patches simultaneously, the CNN yields the state-of-the-art performance in tasks where the local spatial and temporal correlations of the data play a crucial role. Therefore, this model outperforms ReLU ANNs as well as SAE and LSTM in applications where the data has a strong spatiotemporal structure such as the wind and PV power prediction [67, 239] as well as PMU event classification [121]. In [112], this model is applied to 6-hr ahead spatiotemporal solar irradiance prediction which obtains 21.62% and 16.78% better RMSE and MAPE, respectively, compared to the LSTM due to modeling the correlation between the radiation at neighboring solar sites by the convolution operation in (1.2). In addition, in [192], CNN is applied to the transient stability assessment of the IEEE 39-bus system. In a short period of time after a disturbance, the bus voltage phasors sampled from PMUs from various points of the system are given to the CNN to judge if the system is stable, aperiodic unstable or oscillatory unstable. CNN's classification accuracy is 98.7% while recent variations of support vector machines and decision trees lead to 95.2% and 92.1% accuracies. Furthermore, CNN is shown to yield promising results in fault diagnosis [30], harmonic power grid analysis [189], and voltage stability assessment [240].

1.3. Probabilistic Deep Learning

In contrast to discriminative deep learning where an explicit function maps x to y where $(x, y) \in D_{tr}$, the objective of probabilistic deep neural architectures is to capture the PDF $P(x)$ for all samples in the dataset D_{tr}. Then, an explicit function is learned to map $P(x)$ to $P(y|x)$, hence learning the true output y for all samples $(x, y) \in D_{tr}$.

1.3.1. Deep Belief Network

The DBN is a deep MLP with input h^0 and L computational layers h^i $(i = 1, 2, ..., L)$. Each layer h^i is a Restricted Boltzmann Machine (RBM) RBM^i, a generative graphical model that encodes the PDF of its input layer h^{i-1} into its latent feature vector h^i. At each RBM^i $i = 1, 2, ..., L$, the conditional PDF of the j-th neurons in the visible layer h^{i-1} and hidden layer h^i is computed by:

$$\begin{aligned} P(h^i_j = 1|h^{i-1}) &= \sigma(\sum_k W^i_{kj}.h^{i-1}_k + b^i_j) \\ P(h^{i-1}_j = 1|h^i) &= \sigma(\sum_k W^i_{jk}.h^i_k + b^{i-1}_j) \end{aligned} \tag{1.3}$$

To train W^i, the Contrastive Divergence method [115] is employed that adds the gradient of $P(h^{i-1})$ with respect to W^i to increase the likelihood of observing the visible vector h^{i-1} given the latent vector h^i. Similar approach is used to train b^i and b^{i-1} in an unsupervised fashion. When the unsupervised training is done for all layers, a dense layer $o = h^{L+1}$ is added on top of the last layer h^L and the whole neural network is trained by the supervised GD simialr to the SAE.

Table 1.2 shows the large variety of DBN's applications in power systems area. As shown in this table, the DBN leads to accurate wind and PV power prediction results due to capturing uncertainties in the energy time series [115]. Moreover, DBN shows a promising performance in transient stability classification with 94.69% accuracy in the Central China Regional Power Grid [261]. Furthermore, in [90], this method is recently applied to the state estimation of the US PGE69 Distribution Network that led to a remarkably small MAPE of 0.091% which shows the large hypothesis space and low bias of this probabilistic model.

1.3.2. Generative Adverserial Network

Assuming a training set D_{tr}, GAN is an unsupervised deep ANN that learns $P(x)\ s.t.\ x \in D_{tr}$ using a generator ANN $G(z)$ that observes some input noise $z \sim P(z)$ and outputs a sample x' drawn from the generators PDF P_g. The produced sample x' as well as the training samples $x \in D_{tr}$ are given to a discriminator ANN D, a binary classifier which decides if the generated sample x' comes from the true PDF $P(x)$ or the PDF of generated samples P_g. Training the generator and discriminator simultaneously, we improve the generator to create realistic samples by decreasing the distance between the real PDF $P(x)$ and the generated PDF P_g. To train the discriminator D, the following unsupervised objective is applied:

$$\max_D \ \mathbb{E}_{x \sim P(x)}[log\ D(x)] + \mathbb{E}_{x' \sim P_g}[log(1 - D(x'))] \tag{1.4}$$

Here, $D(x)$ is trained to differentiate between the samples generated from $G(z)$ and the true samples $x \sim P(x)$. Using (1.4), to simultaneously optimize ANNs $G(z)$ and D, the following min-max objective is optimized using the GD method:

$$\begin{aligned} \min\nolimits_G \max\nolimits_D J_{D,G} &= \mathbb{E}_{x \sim P(x)}[log\ D(x)] \\ &+ \mathbb{E}_{z \sim P(z)}[log(1 - D(G(z)))] \end{aligned} \tag{1.5}$$

To test the model on a testing set D_{ts}, the Kullback-Leibler (KL) divergence is used as a distance metric between the estimate PDF and the true PDF of samples $x \in D_{ts}$.

As shown in Table 1.2, GAN leads to a promising performance in a diverse set of complex classification problems including fault detection [65] and cyber attack classification [6], as well as regression problems such as scenario generation for the wind and solar power [37]. Compared to the classic DBN, GAN has a larger hypothesis space which leads to higher generalization capacity. Hence, as Table 1.2 shows, GAN outperforms DBN in both fault detetcion and cyber attack classification. Moreover, since GAN explicitly models the joint PDF of the data, it can be directly applied to realistic data synthesis problems such as power grid synthesis [119, 146] while DBN does not have such a capability.

1.3.3. Variational Autoencoder

Similar to GANs, the objective of VAE is to learn the PDF $P(x)\ s.t.\ x \in D_{tr}$ in an unsupervised fashion. The VAE consists of an encoder ANN $q_\theta(z|x)$ parametrized by θ and a decoder ANN $p_\phi(x|z)$ with parameters (weights and biases) ϕ. The encoder maps x into the latent representation z which has a Gaussian distribution estimated by $q_\theta(z|x)$. Then, to find the optimal z that is powerful enough to best reconstruct x, the decoder maps z into the actual input x. Hence, training the VAE consists of maximizing the likelihood of x as well as minimizing the KL divergence KL of the distribution of z (i.e. $q_\theta(z|x)$) and its actual distribution $N(0, I)$ where I is the identity matrix. Therefore, the loss function of the VAE is computed by:

$$J_{VAE} = \sum_{x \in D_{tr}} \Bigg[KL[q_\theta(z|x)||N(0,I)] - \\ \mathbb{E}_{q_\theta(z|x)}[log\ p_\phi(x|z)] \Bigg] \tag{1.6}$$

Training the VAE using GD, the decoder $p_\phi(x|z)$ provides an accurate estimation of the data PDF $P(x)$ when marginalized over all valid z.

As shown in Table 1.2, the VAE is applied to learn the conditional PDF of future wind speed/power given its previous measurements for short-term wind prediction [221]. Moreover, similar technique is applied in [117] and [235] to hourly and 6-hour ahead prediction of PV power with $2.07kW$ and $6.53kW$ better RMSE compared to the DBN, respectively. In addition to regression, VAE outperforms DBN in complex classification tasks with 3.45% accuracy improvement in transient stability assessment [261] and 5.74% better fault detection accuracy [146].) Moreover, VAE is utilized to learn the PDF of the physical and topological characteristics of power networks for power network synthesis. As shown in Table 1.2, VAE generates realistic power networks that accurately imitate not only the topological properties (e.g., diameter and density) but also the power flow statistics (maximum, minimum, and median flow) of the large-scale transmission network in CUSPG dataset [197].

1.4. Deep Reinforcement Learning

Besides classification and regression, deep ANNs are employed in Reinforcement Learning settings where the problem is modeled as a Markov Decision Process (MDP) (S, A, P_a, R_a) with the state set S, action domain A, and state transition probability $P_a(s, s') = P(s_{t+1} = s'|s_t = s, a_t = a)$ to model the likelihood of going from state s_t at time t to state s_{t+1} at time $t + 1$. This transition leads to observing the immediate reward $R_a(s_t = s, s_{t+1} = s')$ from the problem's environment. The goal is find the optimal policy $\pi^*(s_t)$ that determines action a_t to maximize the expected discounted reward sum $R_{avg} = \mathbb{E}\Big[\sum_{t=o}^{\infty} \gamma^t R_a(s_t, s_{t+1})\Big]$. The discounting factor $0 \leq \gamma \leq 1$ decides the contribution of the historical rewards to R_{avg}. The optimal policy $\pi^*(s)$ for a state $s \in S$ is computed by:

$$\pi^*(s) =_a Q(s, a) \tag{1.7}$$

where $Q(s, a)$ is the optimal state-action value function that estimates the reward of taking action a in state s.

1.4.1. Deep Q-network (DQN)

DQN [58] directly learns $Q(s, a)$ and employs (1.7) to find the optimal policy. To provide high generalization power and low estimation bias, the DQN implements $Q(s, a)$ by a deep neural network Q_{ANN} that observes an input $\langle s, a \rangle$ and outputs $Q(s, a)$. To train Q_{ANN}, the Temporal Difference (TD) error δ is defined as the difference between the current $Q(s, a)$ and the value function after the transition to s' computed by:

$$\delta = Q(s, a) - (R_a(s_t = s, s_{t+1} = s') + \gamma \max_a Q(s', a)) \tag{1.8}$$

To train the DQN (i.e., minimize δ), the Huber loss is computed by $J(\delta) = \frac{1}{2}\delta^2$ if $|\delta| \leq 1$ and $J(\delta) = |\delta| - \frac{1}{2}$ otherwise. Applying GD, one can minimize $J(\delta)$ with respect to the weights and biases of Q_{ANN}.

Table 1.3 shows the applications of DLR in the power engineering domain. As shown in this table, DQN is recently applied for optimal voltage control of a 200-bus system [58]. Moreover, this model shows a promising load shedding result of $26MW$ for optimal emergency control of the IEEE 39-bus system [88]. Furthermore, DQN is employed for power grid cost efficiency with transportation energy optimization, and showed 14.1% improvement compared to the classic binary control method [180]. The high generalization power of this method has encouraged the researcher to apply DQN for various real-world applications ranging from electricity marketing [244] and demand-response learning [89] to smart grid scheduling [181] and cyber attck detection [227].

1.4.2. Double DQN (DDQN)

To reduce the overestimation effect of the state-action value $Q(s, a)$ in (1.8), the DDQN uses a target deep ANN parameterized by θ' to compute the update value $\max_a Q(s', a)$ while the state-action $Q(s, a)$ is computed by a deep ANN with the original DQN parameters θ. As shown in Table 1.3, this method improves the classic DQN with 2.2% improvement in cost efficiency for transportation energy optimization [180] and $\pounds 43 * 10^3$ improvement in electricity market bidding profit [244].

1.4.3. Deep Deterministic Policy Gradient (DDPG)

DDPG is an actor-critic DRL algorithm. The actor $\mu(s)$ models the policy as a deep ANN that observes a states s and generates the corresponding continuous action a. The critic Q is a deep ANN that estimates $Q(s, a)$ for the state-action input $< s, a >$. To compute the state's value, the actor's output is given to the critic to calculate $Q(s, a)$. Similar to DQN, The critic's TD-error function J_Q is computed using the Bellman equation:

$$J_Q = \Big(Q(s, \mu(s)) - (R_a(s, s') + \gamma Q'(s', \mu'(s'))) \Big)^2 \tag{1.9}$$

where Q' and μ' are the target critic and actor deep ANNs, respectively. The target ANNs Q' and μ' are time delayed copies of Q and μ that slowly track the learned state-action values. The actor's loss function J_μ is computed by $Q(s, \mu(s))$ which is maximized to increase the DDPG's return while J_Q is minimized. To learn Q and μ using GD, the gradients of J_Q and J_μ with respect to their weights and biases are computed, respectively. Moreover, the target networks Q' and μ' are updated by respectively adding a small fraction of their corresponding parameters in the original networks Q and μ at each DRL episode. Table 1.3 shows the significant experimental advantage of DDPG compared to DQN-based methods. While DQN cannot handle high-dimensional action spaces, the DDPG learns policies in these conditions. Thus, DDPG is shown to generally provide better accuracy in both regression problems such as autonoumous voltage control [58], emergency control [88], strategic bidding [244] as well as classification tasks including cyber attack detection [227] and data integrity protection [9].

1.5. Conclusion

With the growing time and memory complexity of power system applications, the need for advanced statistical pattern recognition tools has lead to the use of deep learning methodologies. This novel class of methods can be mainly categorized into discriminative, generative, and reinforcement learning approaches. This review studies the deep discriminative models that provide an explicit method to map their complex input directly to the problem's solution. Due to their high generalization capacity, these models are widely applied to stability assessment, fault detection, as well as renewable generation prediction. Then, deep generative approaches are reviewed that provide a probabilistic approximation of data PDFs; hence, learning complex probabilistic structures for a wide range of power engineering applications including state estimation, renewable scenario generation, and power grid synthesis. Finally, deep reinforcement learning algorithms are discussed that seek to optimize an objective using the observed rewards captured from the problem's environment. The theoretical and experimental analysis of the employed method motivates future research in the area of deep learning to further extend the applications of this powerful class of models in new perspectives of power engineering.

Table 1.1. Discriminative Deep Learning in Power Systems Applications.

Applications	Dataset	Model	Performance Metric	Result
Reliability Management Response [59]	IEEE-RTS96	ReLU	Coefficient of determination (R^2 Score)	0.964
		SAE		0.951
Stability Assessment [27], [204], [189], [192], [240]	IEEE 39-bus	ReLU	Classification Accuracy	94.1%
		SAE		92.6%
		CNN		97.8%
Fault Detection [27], [238], [251], [30]	IEEE 39-bus	ReLU	Detection Accuracy, Location Accuracy Rate	93.20%, 91.12%
		SAE		94.18%, 91.71%
		CNN		96.09%, 94.31%
PMU Event Classification [121]	16-machine 68-bus Test System	ReLU	Classification Accuracy	94.11%
		SAE		95.07%
		LSTM		96.34%
		CNN		98.17%
Hourly Wind Power Prediction [111], [32], [239], [264], [202], [34]	Western Wind Dataset	ReLU	RMSE, MAPE	1,38%, 1.74%
		SAE		1.24%, 1.68%
		LSTM		1.13%, 1.53%
		CNN		1.07%, 1.26%
Hourly PV Power Prediction [112], [67], [264], [52], [160]	National Solar Radiation Database	ReLU	RMSE, MAPE	1.29%, 1.54%
		SAE		1.09%, 1.37%
		LSTM		0.97%, 1.10%
		CNN		0.85%, 0.92%
Load Modeling [46]	16-machine 68-bus Test System	ReLU	RMSE, MAPE	0.0435, 0.0120
		LSTM		0.008, 0.0071
Hourly Load Forecasting [205]	Industrial Power Demand Dataset	ReLU	Normalized RMSE	0.069
		SAE		0.051
		LSTM		0.032
Power Fluctuation Identification [230]	Market Trading Reports	ReLU	MAE, MAPE	0.042, 107.91%
		LSTM		0.038, 105.72%
Energy Disaggregation [74], [118]	Reference Energy Disaggregation Dataset	SAE	Precision, Recall, F-score	84.63%, 61.04%, 70.62%
		LSTM		89.83%, 65.72%, 75.93%

Table 1.2. Probabilistic Deep Learning in Power Systems Applications.

Applications	Dataset	Model	Performance Metric	Result
Wind Speed Prediction [221]	Shangchuan Island Wind Farm	DBN	RMSE, MAPE	0.5494, 6.39%
		VAE		0.4832, 4.81%
PV Power Prediction [112], [235]	North China Baoding Dataset	DBN	RMSE, MAPE	17.55 kW, 3.76%
		VAE		15.48 kW, 3.63%
Transient Stability Assessment [261]	Central China Regional Power Grid	DBN	Classification Accuracy	94.69%
		VAE		98.14%
Hourly Load Forecasting [75], [170]	Texas Urbanized Area Dataset	DBN	RMSE, MAPE	0.4851, 5.81%
		VAE		0.4032, 5.02%
State Estimation [90]	US PG&E69 Distribution Network	DBN	MAPE, Maximum Absolute Error	0.091, 0.073
		VAE		0.084, 0.069
Fault Detection [146], [175], [262], [225], [65]	Northern China Wind Farm (SCADA)	DBN	Classification Accuracy	79.11%
		VAE		84.85%
		GAN		87.32%
Cyber Attack Detection [6]	5-bus Smart Grid	GAN	Classification Accuracy	95.34%
		VAE		92.18%
Renewable Scenario Generation [37]	Wind & Solar Integration Dataset	GAN	Kullback–Leibler Divergence	0.61
		VAE		0.52
Power Grid Synthesis [119]	Columbia University Synthetic Power Grid (CUSPG)	GAN	Topological Distance, Power Flow Distance	0.678, 3.41 MW
		VAE		0.0512, 3.06 MW

Table 1.3. Deep Reinforcement Learning Applications in Power Systems.

Applications	Dataset	Model	Performance Metric	Result
Voltage Control [58]	Realistic 200-bus System (SCADA)	DQN	Average Control Reward	161.54
		DDPG		124.83
Emergency Control [88]	IEEE 39-bus	DQN	Load Shedding	26 MW
		DDPG		23 MW
Transportation Energy Optimization [180]	California Freeway Performance Measurement System (PeMS)	DQN	Cost Efficiency (compared to binary control)	14.1%
		DDQN		16.3%
Electricity Market [244], [236]	Synthetic Market Dataset	DQN	Profit(£)	5.2 * 10ˆ5
		DDQN		5.63 * 10ˆ5
		DDPG		5.86 * 10ˆ5
Demand-Response Strategy Learning [89]- [242]	Steel Powder Manufacturing Dataset	DQN	Operation Cost($)	161.93
		TD-based Actor-Critic DRL		134.85
Power Scheduling [181]	Shaanix Wind Farm Dataset	DQN	Average Income($)	$ 4268.17
		Improved DQN		$ 4730.21
Cyber Attack Detection [227], [9]	IEEE 9-bus System	DQN	Transient Energy	0.120 p.u.
		DDPG		0.056 p.u.

Chapter 2

Interval Deep Generative Neural Network for Wind Speed Forecasting

In recent years, wind speed forecasting is considered as a challenging task required for the prediction of wind energy resources. As a highly varying data, wind speed time series requires highly nonlinear temporal features for the prediction tasks. However, most forecasting approaches apply shallow supervised features extracted using architectures with few nonlinear hidden layers. Moreover, the exact features captured in such methodologies cannot decrease the wind data uncertainties. In this chapter, an interval probability distribution learning (IPDL) model is proposed based on Restricted Boltzmann Machines and Rough Set Theory to capture unsupervised temporal features from the wind speed data. The proposed model contains a set of interval latent variables tuned to capture the probability distribution of wind speed time series data using contrastive divergence with Gibbs sampling. A real-valued interval deep belief network (IDBN) is further designed employing a stack of IPDLs with a fuzzy type II inference system (FT2IS) for the supervised regression of future wind speed values. In order to automatically learn meaningful unsupervised features from the underlying wind speed data, real-valued input units are designed inside IDBN to better approximate the wind speed probability distribution function compared to classic DBNs. The high generalization capability of our unsupervised feature learning model incorporated with the robustness of IPDLs and FT2IS leads to accurate predictions. Simulation results on the Western Wind Dataset reveal significant performance improvement in 1-hr up to 24-hr ahead predictions compared to single-model approaches including both shallow and deep architectures, as well as recently proposed hybrid methodologies.

2.1. Introduction

In recent years, wind power has received a noticeable attention as a clean source of energy due to the environmental concerns. In the last decade, the global wind markets have grown by an average of 28 percent per year in terms of total installed capacity [179]. In many power systems, the stability and reliability of power generation and the reduction in emission of greenhouse gas are crucial issues to consider. The prediction of wind power which is generally considered as a highly varying time series, plays a key role in addressing such challenges. Since the wind power generated by a wind turbine is highly dependent on the atmosphere meteorology and wind speed, improving the accuracy of wind speed forecasting methods leads to the improvement of wind energy predictions [31]. Therefore, a large variety of time series forecasting methodologies is introduced in the recent literature in order to predict wind speed time series. The wind data has a stochastic and chaotic quality, thus, it is a very complex task to forecast the velocity of wind using linear approaches [57]. In addition, the length of the forecasting horizon has a negative correlation with the accuracy of forecasting methods. Ultra-short-term wind forecasting refers to wind data prediction in the range of a few minutes to one hour ahead [168]. This task is mainly applied for electricity market clearing, real-time grid operations, and regulation actions. Short-term forecasts are mainly for a period starting from one hour to several hours ahead. This type of prediction is generally for unit commitment and operational security in the electricity market. Medium-term and long-term forecasting refers to longer time horizons [24].

In the technical literature, wind forecasting methodologies are mainly classified into four categories: 1) Persistence model has a naïve smoothness assumption on the target function. In this approach, the future wind speed is considered to be equal to the wind speed in the forecasting time [260]. This method is the simplest and the most economical wind forecasting approach and is therefore widely employed by electrical utilities. The performance of Persistence model degrades rapidly when the forecasting time horizon is extended; hence, this model is only reliable for ultra-short-term purposes. 2) Physical methods are based on numerical weather prediction (NWP) using temperature, pressure, and obstacles as the weather parameters [135]. NWP outputs accurate estimations for long-term predictions mainly utilized for large-scale areas. The major drawback of numerical weather prediction models is the high time and memory complexity to produce results. This leads to serious issues when the model encounters unexpected errors during prediction. Hence, this methodology is not reliable for short forecasting horizons. 3) Statistical methods find the mathematical relationship between the online data of wind speed time series. Statistical models include auto regressive (AR), auto regressive moving average (ARMA), auto regressive integrated moving average (ARIMA), Bayesian approach, and gray predictions. [231] presents a hybrid AR approach using a K-nearest neighbor (KNN) regression model for short-term wind speed forecasting. Historical data samples are used to learn the coefficients of a KNN regression approach to capture variation patterns of the wind speed time series. Finding K nearest neighbors significantly increase the computational burden of the prediction method; hence, this approach has high computational complexity. Moreover, this method can suffer from the curse of dimensionality problem as the number of parameters grow exponentially with the growth in input size. The authors of [62] applied multiple variations of ARMA to forecast both wind speed and wind direction tuples. Although this model is applied for hourly wind data prediction, it cannot give accurate estimations for longer time horizons due to the linear assumptions in wind data patterns. In [97], the authors introduce a Bayesian forecasting approach based on structural break modeling that can incorporate domain knowledge about wind data. The model is applied for ultra-short-term wind prediction of utility-scale wind turbines. The linear charactristics of the presented structural break method restricts the ability of this model to address more challenging prediction problems with longer forecasting time horizons. 4) Artificial intelligence (AI) techniques including artificial neural networks (ANNs) [8,28,64,107,134,145,177,191,254], support vector regression (SVR) [83], and fuzzy methods [63,150] led to novel methodologies for wind prediction. ANNs are widely applied learning mathematical models that can capture the relationships between the input data and the forecasted wind speed values. In the relevant literature, ANNs are utilized for time series prediction of different weather variables in various time scales and yield satisfactory results when compared to traditional algorithms [177]. Feed-forward ANN [64,134,191] recurrent ANN [28], radial basis function (RBF) ANN [107,254], ridgelet ANN [8] and adaptive wavelet ANN [145] are recently proposed for wind speed and wind power forecasting. ANN-based approaches have been widely applied in the time series forecasting domain due to their capability to represent complex non-linear relationships between the input and output variables. Moreover, SVR [83] is introduced in the domain of short-term wind prediction as a kernel-based methodology that utilizes input features obtained from the generalized principal component analysis (GPCA). SVR implicitly learns features in a high dimensional space applying the kernel trick. The presented model in [83] extracts error-prone hand-engineered features captured by the GPCA for short-term forecasts. In recent years, fuzzy methods are also introduced in the domain of AI for wind prediction. The authors of [63] presented a two-stage adaptive neuro-fuzzy inference system that maps weather data collected from NWP to wind power values for short-term wind prediction. Obtaining NWP measurements is computationally complex; hence, this methodology cannot be applied for short-term purposes. In [150], a fuzzy version of support vector machines (SVMs) is developed for short-term wind speed forecasting. The proposed fuzzy manifold learning approach addresses the noise sensitivity issues in SVM; however, the accuracy improvement is not noticeable as the applied maximum margin SVM is more suitable for classification problems rather than regression tasks.

The AI methodologies introduced in recent literature can be viewed in two categories, shallow architectures and deep learning models; 1) Shallow models including feed-forward [64, 134, 191] and recurrent ANNs [28] and their different variations such as RBF [107, 254], adaptive wavelet ANN [145], and nonlinear autoregressive networks [12] are designed using single hidden layer to capture temporal features. In contrast to deep methodologies, such models are not capable of automatically learning unsupervised features from the data. Therefore, they require error-prone feature selection for the prediction model. 2) Deep learning architectures are able to train several layers of hidden computational units with high generalization capability. In very recent literature, [111] applied deep stacked auto-encoders (SAEs) for short-term wind forecasting. Denoising autoencoders are employed for the dimensionality reduction of wind speed time series. [248] applies a Bernoulli deep belief network (DBN) for the problem of short-term wind prediction. The deep architectures outperform conventional learning models including AR-based methods, ANNs, and SVR due to the following reasons: *a) Problem complexity* – When the target function is smooth enough, it can be estimated by applying shallow features with a low level of abstraction. However, in the case of wind prediction, the intermittent wind data is highly varying; thus, the smoothness assumption in the shallow models will lead to poor forecasting accuracy. *b) Sample complexity*

– In some regression problems, the size of the training dataset is small; thus, training shallow networks is preferred because using more complex deep networks with large parameter spaces will lead to the overfitting issues. However, in the case of wind prediction problems, the overfitting problem can be effectively avoided using ample wind data available for training. *c) Error-prone feature selection* – Most methodologies including ANN-based approaches [8, 28, 64, 107, 134, 145, 177, 191, 254], kernel-based models [83], and fuzzy methodologies [63, 150] need tediously hand engineered features. These models require sufficient prior knowledge about the specific domain in order to select reliable features from the wind data. However, DBN and SAE can leverage the unsupervised data to initialize model parameters; hence, they can be viewed as regularization techniques.

The prediction made by various regression methodologies have an irreducible uncertainty [83] that should be handled to increase accuracy for the applications of scheduling, maintenance, and resource planning in the wind energy generation [31]. However, deep networks introduced in the recent literature assume that the input data is clean. Moreover, these methods cannot model real-valued data as they assume Bernoulli distributions for the input variables. In this chapter, a novel interval probability distribution learning (IPDL) model is proposed for learning nonlinear temporal features from the time series data in order to address these issues. Our IPDL model is proposed as a graphical generative learning approach based on the Restricted Boltzmann Machines [77] and the Rough Set Theory [143, 173] to capture interval unsupervised features from the underlying input time series. IPDL consists of two subsets of visible (observable) and hidden nodes in a fully connected structure. Each visible unit is connected to the set of all hidden units and vice versa. The visible units contain input variables, that is, the exact noisy speed values collected from the wind data. The hidden units contain interval upper- and lower-bound values to extract inexact (rough) patterns from the input vector. An interval based energy function is defined on each configuration of binary values for all visible and hidden nodes. A probability distribution function is learned by decreasing the energy function while increasing the probability of observed input vectors in the wind speed time series dataset. It is shown that the conditional probability of visible and hidden layers can be easily decomposed to simple factors calculated with low computational burden.

Moreover, an unsupervised learning algorithm is presented based on contrastive divergence with Gibbs sampling to efficiently learn the IPDL parameters. In order to tune the parameters, IPDL's energy function is decreased using the gradient of an unsupervised loss function. Furthermore, an interval deep belief network (IDBN) with real-valued input vectors is proposed using a stack of IPDLs that capture probability distribution of wind data. A hybrid wind speed forecasting framework, DeepHybrid, is designed using IDBN wind features and a fuzzy type II inference system (FT2IS) for the supervised regression of future wind speed values. The supervised loss function is differentiable with respect to all the IDBN model parameters; hence, the whole forecasting framework can be tuned in an end-to-end fashion using gradient-based methodologies.

The contributions of the proposed architecture can be viewed in two areas: *a) Machine Learning:* The development of a novel interval probability distribution learning system and the incorporation of the Rough Set Theory with generative deep learning models to extract robust highly nonlinear features from the input data. *b) Wind forecasting:* The application of an unsupervised feature extraction model (rather than the hand-engineered or shallow features applied in previous methodologies), as well as fuzzy type II systems, in nonlinear manifold learning from wind data for supervised target function (future wind values) estimation.

The proposed deep learning framework has the following contributions:

1. A new probability distribution learning model, IPDL, is presented based on the Rough Set Theory and deep learning for the robust unsupervised feature extraction of time series data. The proposed generative model is proved to capture the joint distribution of input variables. Moreover, the inference and learning algorithms for the devised model are presented. To the best of our knowledge, our proposed IPDL is the first generative deep learning model that can capture interval knowledge from the data.

2. Real-valued input units are proposed for the interval DBN that can more accurately capture the wind speed patterns compared to previously applied DBN [248] in the literature. The classic DBN applied in the domain of time series prediction assumes that the input variables are sampled from a Bernoulli distribution while our novel IDBN model considers real-valued input variables; hence, the proposed architecture can more accurately model the temporal

data for real-world applications.

3. The proposed model can extract meaningful features from the input in an unsupervised manner. Thus, unlike other AI approaches including ANNs [8, 28, 64, 107, 134, 145, 191, 254], SVR [83], and fuzzy methodologies [63, 150] that are based on the supervised regression methods, no prior knowledge about the wind data is needed for the feature extraction.

4. In contrast to fuzzy networks in the literature [103, 140, 154] which are randomly initialized, the proposed framework finds the optimal initialization for the fuzzy system's membership functions using the deep learning-based unsupervised processing of the data in the generative IPDL stack of IDBN. The proposed IPDLs act as a generalization technique on the system's weights and biases; therefore, as shown in the simulation results, the proposed framework can more accurately address the uncertainties in the data. Moreover, as discussed in the simulation results, our model outperforms the fuzzy type-1 short-term prediction methodology [111] due to capturing input signal distributions as well as interval Type-2 rules.

The chapter is organized as follows: In section 2.2, the concept of wind speed data analysis is discussed. Section 2.3 describes the novel interval probability distribution learning model devised for unsupervised feature learning for deep neural architectures. The inference and learning algorithm of the proposed model is explained in this section. In section 2.4 the proposed time series forecasting approach, DeepHybrid, based on Deep Learning, Rough set theory and Fuzzy systems is introduced. Simulation results and comparison of the proposed approach with recent AI methodologies, including shallow ANNs and deep ANNs, are described in section 2.5. Finally, the conclusions and future works are provided in Section 2.6.

2.2. Wind Speed Data Analysis

Wind speed is a non-linear time series with many fluctuations; therefore, approaches based on the smoothness assumption such as Persistence method would not have an appropriate performance in order to be applied for the prediction tasks of long horizons. The proposed nonlinear method is a data-driven approach that captures statistical patterns from the input wind speed data. Identifying the optimal structure of data-driven models is a vital issue considered by in-

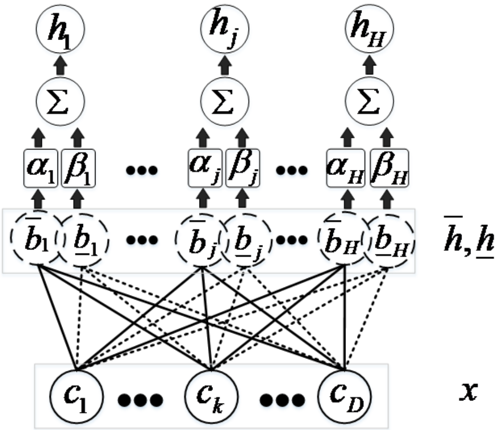


Figure 2.1. Structure of the interval probability distribution learning model with input x

put variable selection methods. In the recent literature, there are several papers such as [145] that have applied an autocorrelation function (ACF) to obtain the cross-correlation of wind speed time series at various time samples. Since only the linear dependence of a variable with itself can be computed by ACF and the wind speed data has a highly non-linear nature, Mutual Information (MI) is utilized as an effective approach to computing the non-linear correlations in the data as well as the linear correlations. Assuming two random variables$X$and$Y$, the entropy of Xdenoted by$H(X)$computes its uncertainty and$H(X, Y)$is the joint entropy of $X$and$Y$.The Conditional entropy computed by $H(Y|X) = H(X, Y) - H(X)$ shows the uncertainty of Ygiven that the variable X is observed. MI between two random variables is a non-linear function to measure the amount of information possessed about a variable when the other variable is observed. MI is calculated by$I(X, Y) = H(Y) - H(Y|X)$ and is the reduction in the uncertainty of variable$Y$given the observation of variable$X$. Considering$v(t)$as the wind speed value at time$t$, the MI between$v(t - l + 1)$and$v(t + 1)$is calculated considering$l >= 1$as the time-lag. In order to select input variables for the prediction method, the wind speed data corresponding to the time-lags with MI greater than the threshold$\tau > 0$ are selected as the input set for our algorithm to highlight the correlation in the wind speed time series.

2.3. Interval Probability Distribution Learning

In this section, first, the Rough Set Theory is explained. Then, the proposed interval probability distribution learning model is introduced based on deep learning and the Rough Set Theory. The proposed generative model is proven to capture probability distribution of its input data. Moreover, an inference approach and a learning algorithm is proposed to tune the interval upper- and lower-bound parameters of the presented model.

2.3.1. Rough Feature Extraction

The Rough set theory is a mathematical method introduced by Pawlak [143, 173] to deal with uncertain knowledge. An Information System$S$is defined by a 4-tuple $\langle U, A, V, f\rangle$. Here, the universe of primitive objects $U$is a finite non-empty set, and$A$is a finite non-empty set containing the attributes. Each attribute$a \in A$is associated with a domain set $V_a$and$V = \bigcup_{a\in A} V_a$.Sdefines a total information function$f : U \times A \to V$, and for every $a \in A$and$x \in U$,$f(x, a) \in V_a$. Suppose that$M \subseteq A$, then two objects$x, y \in U$ are indiscernible from each other in S by the set M, if and only if for every$a \in M$,$f(x, a) = f(y, a)$.$M \subseteq A$has a binary indiscernibility relation $IND(M)$on$U$which is called. The rough set theory defines two approximations for any concept set$X \subseteq U$and attribute set$M \subseteq A$. Using the knowledge of$M$,Xcan be approximated by the M-lower approximation$\underline{MX}$and M-upper approximation$\overline{MX}$:

$$\underline{MX} = \cup\{O \in U|M : \; O \subseteq X\} \tag{2.1}$$

$$\overline{MX} = \cup\{O \in U|M : \; O \cap X \neq \emptyset\} \tag{2.2}$$

and the M-boundary region of set$X$is defined by

$$BND_M(X) = \overline{MX} \; - \; \underline{MX} \tag{2.3}$$

Here,$\underline{MX}$is the set of all objects in$U$which can be certainly classified as members of $X$with respect to the set of attributes M.$\overline{MX}$is the set of objects in$U$which can possibly be classified as members of$X$with respect to the set of attributes M. The boundary region is the set of objects that cannot certainly be classified to$X$only by employing the set of attributes$M.BND_M(X)$describes the vagueness of$X$. If $BND_M(X) = \emptyset$then$X$is crisp (exact) with respect to M and if $BND_M(X) \neq \emptyset$ then$X$is called a rough (inexact) set.

2.3.2. Interval distribution learning model

A probabilistic generative model is introduced to learn the probability distribution of an input vector $x \in \mathbb{R}^D$ using a visible layer with observable units and a latent representation layer (hidden layer). The hidden layer $h \in \mathbb{R}^H$ reduces the dimensionality of the input data by capturing the most important characteristics of x inside h. Motivated by the Rough Set Theory, here, the latent units h are approximated by upper- and lower-bound estimations denoted by $\overline{h}$ and $\underline{h}$, respectively. Each feature h is a linear combination of $\overline{h}$ and $\underline{h}$; hence the activation of the j-th unit at the latent representation layer can be computed by $h_j = \alpha_j\overline{h}_j + \beta_j\underline{h}_j$ with α_j and β_j coefficients for each hidden unit j. As shown in Fig. 2.1, the proposed IPLD is an energy-based generative model with the following energy function:

$$
\begin{aligned}
E\left(x,\overline{h},\underline{h};\alpha,\beta\right) = &-\left(\alpha\overline{h}^T\overline{W}x + \beta\underline{h}^T\underline{W}x\right) \\
&- c^Tx - (\alpha\overline{b}^T\overline{h} + \beta\underline{b}^T\underline{h})
\end{aligned}
\tag{2.4}
$$

where $\overline{W}^{H\times D}$ and $\underline{W}^{H\times D}$ are the upper- and lower-bound weights, respectively. $c^{D\times 1}$ is the bias vector for the input vector, $\overline{b}^{H\times 1}$ is the upper-bound bias approximation for h while $\underline{b}^{H\times 1}$ is its lower-bound estimation. The coefficients $\alpha^{H\times 1}$ and $\beta^{H\times 1}$ decide the contribution of upper- and lower-bound hidden vectors on the total output h of this system. The energy function can be written as:

$$
\begin{aligned}
E\left(x,\overline{h},\underline{h};\alpha,\beta\right) = &-\left(\sum_j\sum_k \alpha_j\overline{W}_{j,k}\overline{h}_jx_k + \sum_j\sum_k \beta_j\underline{W}_{j,k}\underline{h}_jx_k\right) \\
&-\sum_k c_kx_k - \left(\sum_j \alpha_j\overline{b}_j\overline{h}_j + \sum_j \beta_j\underline{b}_j\underline{h}_j\right)
\end{aligned}
\tag{2.5}
$$

where $1 \leq j \leq H$ and $1 \leq k \leq D$ are the indices of hidden units and visible units, respectively. A joint probability distribution function is defined on the configuration of random variables x, $\overline{h}$, and $\underline{h}$:

$$P\left(x, \overline{h}, \underline{h}; \alpha, \beta\right) = \exp\left(-E\left(x, \overline{h}, \underline{h}; \alpha, \beta\right)\right)/Z \tag{2.6}$$

Here, Z is a partition function that normalizes the probability for all configurations of $(x, \overline{h}, \underline{h})$ to sum to 1. When the energy of a specific configuration is large, the probability of that configuration occurring in the system is small, while the probability associated with a low energy in (3.6) is large. When the energy of a configuration $(x, \overline{h}, \underline{h})$ is decreased, its likelihood in the model is increased. Therefore, the distribution of the input x is captured if the model learns to decrease the energy of observing x for the samples in the dataset by learning the values of $\overline{h}$ and $\underline{h}$. Our IPDL can be viewed as a Markov Network in which the joint probability distribution function of visible and hidden units can be factorized as computed by:

$$\begin{aligned} P\left(x, \overline{h}, \underline{h}; \alpha, \beta\right) &\propto \exp\left(\alpha \overline{h}^T \overline{W} x + \beta \underline{h}^T \underline{W} x + c^T x + \alpha \overline{b}^T \overline{h} + \beta \underline{b}^T \underline{h}\right) \\ &= \exp\left(\alpha \overline{h}^T \overline{W} x\right) \exp\left(\beta \underline{h}^T \underline{W} x\right) \exp\left(c^T x\right) \exp(\alpha \overline{b}^T \overline{h}) \quad \exp\left(\beta \underline{b}^T \underline{h}\right) \end{aligned} \tag{2.7}$$

As shown in (3.7), the joint probability density associated with the IPDL model is factorized into upper-bound factors i.e. $\exp\left(\alpha \overline{h}^T \overline{W} x\right)$ and $\exp(\alpha \overline{b}^T \overline{h})$, lower-bound factors i.e. $\exp\left(\beta \underline{h}^T \underline{W} x\right)$ and $\exp(\beta \underline{b}^T \underline{h})$, and the input configuration factor $\exp\left(c^T x\right)$. The upper- and lower-bound factors indicate how much the latent variables $\overline{h}$ and $\underline{h}$ are aligned with their corresponding bias variables $\overline{b}$ and $\underline{b}$, while the input configuration factor shows whether the input variables are aligned with the corresponding bias c or not. If large values are assigned to both x_k and c_k, the probability of configurations corresponding to that assignment grows, while the energy function is decreased. However, if x_k and c_k have opposite values (x_k has high/low values while c_k contains low/high values), the energy $E\left(x, \overline{h}, \underline{h}; \alpha, \beta\right)$ is increased leading to the decrease in the probability of the associated configurations $P\left(x, \overline{h}, \underline{h}; \alpha, \beta\right)$.

2.3.3. Inference in IPDL

In order to do inference in the proposed probabilistic network, the probability of latent vector h given input x is computed:

$$P\left(h \mid x\right) = \prod_j P\left(h_j \mid x\right) = \prod_j P(\alpha_j \overline{h}_j + \beta_j \underline{h}_j)$$

$$= \frac{\frac{\exp\left(\alpha \overline{h}^T \overline{W} x + \alpha \overline{b}^T \overline{h} + \beta \underline{h}^T \underline{W} x + \beta \underline{b}^T \underline{h} + c^T x\right)}{Z}}{\sum_{\hat{h} \in \{0,1\}^H} \frac{\exp\left(\alpha \overline{\hat{h}}^T \overline{W} x + \alpha \overline{b}^T \overline{\hat{h}} + \beta \underline{\hat{h}}^T \underline{W} x + \beta \underline{b}^T \underline{\hat{h}} + c^T x\right)}{Z}}$$

$$= \frac{\prod_j \exp\left(\sum_j \alpha_j \overline{W}_{j.} \overline{h}_j x + \alpha_j \bar{b}_j \overline{h}_j\right) \exp\left(\sum_j \beta_j \underline{W}_{j.} \underline{h}_j x + \beta_j \underline{b}_j \underline{h}_j\right)}{\sum_{\hat{h}_1 \in \{0,1\}} \cdots \sum_{\hat{h}_H \in \{0,1\}} \exp\left(\sum_j \alpha_j \overline{W}_{j.} \overline{\hat{h}}_j x + \alpha_j \bar{b}_j \overline{\hat{h}}_j\right) \exp\left(\sum_j \beta_j \underline{W}_{j.} \underline{\hat{h}}_j x + \beta_j \underline{b}_j \underline{\hat{h}}_j\right)} \tag{2.8}$$

Here, since the nodes inside the latent representation layer are mutually independent, the denominator in (3.8) can be written as a multiplication of individual expressions each corresponding to one specific hidden unit:

$$\prod_j \sum_{\hat{h}_j \in \{0,1\}} \exp\left(\sum_j \alpha_j \overline{W}_{j.} \overline{\hat{h}}_j x + \ \alpha_j \bar{b}_j \overline{\hat{h}}_j\right) \exp\left(\sum_j \beta_j \underline{W}_{j.} \underline{h}_j x + \beta_j \underline{b}_j \underline{\hat{h}}_j\right) \tag{2.9}$$

Therefore, the conditional probability of (3.8) can be computed as:

$$P\left(h \mid x\right) = \prod_j \left(\frac{\exp\left(\alpha_j \overline{W}_{j.} \overline{h}_j x + \alpha_j \bar{b}_j \overline{h}_j\right)}{1 + \exp\left(\alpha_j \overline{W}_{j.} \overline{h}_j x + \alpha_j \bar{b}_j \overline{h}_j\right)}\right) \prod_j \left(\frac{\exp\left(\alpha_j \underline{W}_{j.} \underline{h}_j x + \alpha_j \underline{b}_j \underline{h}_j\right)}{1 + \exp\left(\alpha_j \underline{W}_{j.} \underline{h}_j x + \alpha_j \underline{b}_j \underline{h}_j\right)}\right) \tag{2.10}$$

The conditional probability in (3.10) can be further written as the multiplication of conditional probability of the upper-bound hidden units $\overline{h}_j$ and lower-bound hidden units $\underline{h}_j$; hence, the conditional probability of latent representation given the input vector in (3.10) is expressed as:

$$P\left(h \mid x\right) = \prod_j P\left(\overline{h}_j | x; \alpha_j\right) P(\underline{h}_j | x; \beta_j) \tag{2.11}$$

where the conditional probability of upper-bound hidden representation given the input x and the upper-bound coefficient α_j is computed using:

$$P\left(\overline{h}_j|x;\alpha_j\right) = \frac{\exp\left(\sum_j \alpha_j \overline{W}_{j.}\overline{\hat{h}}_j x + \ \alpha_j \bar{b}_j \overline{\hat{h}}_j\right)}{1+\exp\left(\sum_j \alpha_j \overline{W}_{j.}\overline{\hat{h}}_j x + \ \alpha_j \bar{b}_j \overline{\hat{h}}_j\right)} = sigm\left(\alpha_j \overline{W}_{j.}x + \alpha_j \bar{b}_j\right) \tag{2.12}$$

Here, $sigm$ denotes the nonlinear sigmoid function. Similar to (3.12), the lower-bound conditional probability is written as $\left(\underline{h}_j \mid x;\beta_j\right) = sigm\left(\beta_j \underline{W}_{j.}x + \beta_j \underline{b}_j\right)$.

2.3.4. Learning Algorithm for IPDL

Assuming similar contribution for the upper-bound and lower-bound hidden units ($\alpha_j = 1 - \beta_j = 0.5$), the feed-forward computation of conditional probabilities $P(\overline{h}_j|x)$ and $P(\underline{h}_j|x)$ in (3.11) and (3.12) are the following:

$$P\left(\overline{h}_j|x\right) = \begin{cases} sigm\left(\overline{W}_j x + \bar{b}_j\right) & if \quad \overline{W}_j \geq \underline{W}_j \ \ and \ \ \bar{b}_j \geq \underline{b}_j \\ sigm\left(\overline{W}_j x + \underline{b}_j\right) & if \quad \overline{W}_j \geq \underline{W}_j \ \ and \ \ \underline{b}_j \geq \bar{b}_j \\ sigm\left(\underline{W}_j x + \bar{b}_j\right) & if \quad \underline{W}_j \geq \overline{W}_j \ \ and \ \ \bar{b}_j \geq \underline{b}_j \\ sigm\left(\underline{W}_j x + \underline{b}_j\right) & Otherwise \end{cases}$$

$$P\left(\underline{h}_j|x\right) = \begin{cases} sigm\left(\overline{W}_j x + \bar{b}_j\right) & if \quad \overline{W}_j \leq \underline{W}_j \ \ and \ \ \bar{b}_j \leq \underline{b}_j \\ sigm\left(\overline{W}_j x + \underline{b}_j\right) & if \quad \overline{W}_j \leq \underline{W}_j \ \ and \ \ \underline{b}_j \leq \bar{b}_j \\ sigm\left(\underline{W}_j x + \bar{b}_j\right) & if \quad \underline{W}_j \leq \overline{W}_j \ \ and \ \ \bar{b}_j \leq \underline{b}_j \\ sigm\left(\underline{W}_j x + \underline{b}_j\right) & Otherwise \end{cases} \tag{2.13}$$

As computed in (3.13), the probability of $\overline{h}_j$ being 1, is a function of upper-bound parameters $\overline{W}$ and $\bar{b}$ if the net value $\overline{W}_j x + \bar{b}_j$ fed to the upper-bound hidden unit is greater than the lower-bound net value $\underline{W}_j x + \underline{b}_j$; otherwise, the lower-bound parameters $\underline{W}$ and $\underline{b}$ contribute to the computation of the conditional probability $P\left(\overline{h}_j|x\right)$. Similar behavior is considered in (3.13) for the probability of lower-bound unit $\underline{h}_j$ being 1. If $\overline{W}$ and $\overline{b}$ lead to smaller net value compared to $\underline{W}$ and $\underline{b}$, then the upper-bound parameters are applied for the computation of $P\left(\underline{h}_j|x\right)$; otherwise the corresponding lower-bound parameters are employed for the feed-forward algorithm to obtain $\underline{h}$ having the input x.

In order to train the upper-bound and lower-bound parameters using T number of data samples inside the training set $D^{tr} = \left\{x^{(t)} \;\middle|\; 1 \leq t \leq T\right\}$, an unsupervised log-probability loss function is computed:

$$J\left(D^{tr}\right) = \frac{1}{T}\sum_{t=1}^{T} -\log(P(x^{(t)})) \tag{2.14}$$

Here, $J(D^{tr})$ is our supervised error function defined on the data D^{tr} that should be optimized tuning the parameters. In order to update any parameter θ, the stochastic gradient of J with respect to θ is computed by:

$$\begin{aligned} \frac{\partial - \log(P(x^{(t)}))}{\partial\theta} &= \mathbb{E}_{\overline{h}}\left[\frac{\partial E\left(x^{(t)}, \overline{h}, \underline{h}\right)}{\partial\theta} \;\middle|\; x^{(t)}\right] \\ &+ \mathbb{E}_{\underline{h}}\left[\frac{\partial E\left(x^{(t)}, \overline{h}, \underline{h}\right)}{\partial\theta} \;\middle|\; x^{(t)}\right] \\ &+ \mathbb{E}_{x,\overline{h},\underline{h}}\left[\frac{\partial E\left(x^{(t)}, \overline{h}, \underline{h}\right)}{\partial\theta}\right] \end{aligned} \tag{2.15}$$

where $\mathbb{E}$ is the expected value operation notation. The first two expectation terms, $\mathbb{E}_{\overline{h}}\left[\frac{\partial E\left(x^{(t)},\overline{h},\underline{h}\right)}{\partial\theta} \;\middle|\; x^{(t)}\right]$ and $\mathbb{E}_{\underline{h}}\left[\frac{\partial E\left(x^{(t)},\overline{h},\underline{h}\right)}{\partial\theta} \;\middle|\; x^{(t)}\right]$, on the right hand side of (3.15), can be efficiently computed; however, the third term, $\mathbb{E}_{x,\overline{h},\underline{h}}\left[\frac{\partial E\left(x^{(t)},\overline{h},\underline{h}\right)}{\partial\theta}\right]$, is computationally intractable as the number of input variables and hidden units grow. Hence, the third expectation operation in (3.15) is replaced by a point estimate at a single data point $\tilde{x}$. Fig. 2.2 shows the flowchart diagram of the proposed algorithm of training the IPDL model with maximum number of epochs $epoch_{max}$ and learning rate ?. Here, θ_t represents model parameters at time step t. In order to obtain $\tilde{x}$ for the time step t, first $\overline{h}$ and $\underline{h}$ are sampled feeding a data point from the training set $x^{(t)}$ to the IPDL and using the conditional probabilities defined in (3.13). Then, applying (3.11), sampling of h is done quiet efficiently to obtain a latent representation $\tilde{h}$ and the corresponding upper- and lower-bound estimations, $\overline{\tilde{h}}$ and $\underline{\tilde{h}}$. Finally, the hidden vector sample $\tilde{h}$ is applied to compute the input vector sample point $\tilde{x}$ using the following formulation:

$$P\left(\tilde{x}_k \,\middle|\, \overline{\tilde{h}}, \underline{\tilde{h}}\right) = sigm(\sum_j (\overline{\tilde{h}}_j^T \overline{W}_{j,k} + \underline{\tilde{h}}_j^T \underline{W}_{j,k}) + c_k) \tag{2.16}$$

Notice that (3.16) assumes $\overline{W}_{j,k} \geq \underline{W}_{j,k}$. If $\overline{W}_{j,k} < \underline{W}_{j,k}$, similar formulation can be used swapping $\overline{W}_{j,k}$ and $\underline{W}_{j,k}$ parameters. In order to tune IPDL using (3.15), the expectations operations of (3.15) are estimated by:

$$\mathbb{E}_h\left[\frac{\partial E\left(x^{(t)}, \overline{h}, \underline{h}\right)}{\partial\theta} \,\middle|\, x^{(t)}\right] \approx \frac{\partial E\left(x^{(t)}, \overline{\tilde{h}}, \underline{\tilde{h}}\right)}{\partial\theta}$$

$$\mathbb{E}_{x,h}\left[\frac{\partial E\left(x^{(t)}, \overline{h}, \underline{h}\right)}{\partial\theta}\right] \approx \frac{\partial E\left(\tilde{x}, \overline{\tilde{h}}, \underline{\tilde{h}}\right)}{\partial\theta} \tag{2.17}$$

where the upper-bound and lower-bound hidden samples are computed by:$\overline{\tilde{h}} \sim P\left(\overline{h} \mid x = \tilde{x}\right)$

$$\underline{\tilde{h}} \sim P\left(\underline{h} \mid x = \tilde{x}\right) \tag{2.18}$$

The proposed algorithm decreases the energy function E at the training observation $\left(x^{(t)}, \overline{\tilde{h}}, \underline{\tilde{h}}\right)$ while increasing it at the sample values $\left(\tilde{x}, \overline{\tilde{h}}, \underline{\tilde{h}}\right)$ obtained from the model. Hence, at each iteration t, the model distribution gets closer to the real distribution of the data.

Using (3.15), (3.16), and (3.17), the learning rule of the upper-bound weight parameter is computed by:

$$\frac{\partial E(x, \overline{h}, \underline{h})}{\partial \overline{W}_{j,k}} = \left[\frac{\partial E}{\partial \overline{W}_{j,k}}\left(-\sum_j \sum_k \alpha_j \overline{W}_{j.} \overline{h}_j x_k\right)\right]\left(\mathbb{I}\left(\overline{h}_j \geq \underline{h}_j\right)\right)$$

$$+\left[\frac{\partial E}{\partial \overline{W}_{j,k}}\left(-\sum_j \sum_k \alpha_j \overline{W}_{j.} \overline{h}_j x_k\right)\right]\left(\mathbb{I}\left(\overline{h}_j < \underline{h}_j\right)\right)$$

$$= \left(-\alpha_j \overline{h}_j x_k\right) I\left(\overline{h}_j \geq \underline{h}_j\right) + \left(-\beta_j \overline{h}_j x_k\right) I\left(\overline{h}_j \geq \underline{h}_j\right) \tag{2.19}$$

Here, $\mathbb{I}$ is the indicator function. The lower-bound weights as well as the bias vectors can be tuned similar to (3.19) using the gradient of the interval energy function defined in (3.5).

2.4. DeepHybrid Wind Forecasting Method

The proposed deep hybrid methodology consists of an interval deep belief network with rough pattern recognition and fuzzy type II inference system. Fig. 2.3 shows the structure of the proposed DeepHybrid model. First, as discussed in Section II, a feature selection algorithm based on Mutual Information is applied to the historical wind speed time series and the time lags more correlated to the future wind data are selected as the D-dimensional input variable vector $< x_1, x_2, \ldots, x_D >$. An interval deep belief network, using IPDL generative models with real-valued input variables is proposed in order to extract nonlinear features from the unlabeled wind speed distribution. The IDBN contains L number of IPDLs stacked together to extract temporal features. These features are learned by maximizing the log-likelihood of the IPDL models as an unsupervised approach to initialize the weights and biases of a multi-layer neural network. The initialization process can also be viewed as a regularization task, where the randomly-initialized parameters are moved to a good initial subspace. The resulting activations received from the IDBN for each data sample are fed to Gaussian membership functions with interval standard deviations to be utilized by a fuzzy type II takagi sugeno kang (TSK) inference system. The TSK is employed as a regression model to approximate the future time series values. The basic difference of the proposed TSK system compared to the Mamdani is the use of crisp sets in the consequent part. Thus, the calculation of the output signal is computationally simpler than Mamdani structures which require more time complexity due to the use of membership functions that are further deffuzied.

2.4.1. DeepHybrid Structure and Algorithm

The proposed AI methodology, DeepHybrid, consists of three stages:

a) Unsupervised Probability Distribution Learning– Fig. 2.3 depict the structure of the proposed DeepHybrid model. First, an interval DBN is designed using a stack of the proposed IPDL model and a TSK fuzzy system. In contrast to classic DBNs [248], here, real-valued input units are considered for the initial IPDL in the stack in order to more accurately learn the probability density of real-valued wind data. The IPDLs are trained consecutively with no supervision and using raw unlabeled wind speed time series with no preprocessing; hence, this step does not require any prior knowledge from the problem domain to extract features from the time series. Each IPDL is trained using (3.17), (3.18) and (3.19).

Considering L number of IPDL models in the stack of the IDBN, the network should train L generative models consecutively. The input to the first IPDL in the stack, $IPDL_1$, is the observed time series data $\boldsymbol{x} =< x_1, x_2, \dots, x_D >$. This model learns features h^1 from the input data $\boldsymbol{x}$. The i-th IPDL, $IPDL_i$, receives the features obtained from its previously trained IPDL, i.e. $IPDL_{i-1}$, denoted by h^{i-1}, and learns the target representation h^i that is fed to the following IPDL model.

b) Supervised IDBN tuning – The IDBN is fine-tuned applying linear regression after the L-th layer with the desired prediction output as the supervised signal. The initial membership function parameters are set by clustering the representation h^L obtained from IDBN, that is, the features obtained from the L-th IPDL model. The supervised squared error loss function is applied at this stage.

c) FT2IS learning and DeepHybrid fine-tuning – The hybrid predictor is fine-tuned applying stochastic gradient descent (SGD) method in an end-to-end manner. As the output of the proposed network is differentiable with respect to the IPDL models' upper-bound and lower-bound parameters, as well as the FT2IS model, the whole system can be trained efficiently in an end-to-end manner. The proposed learning procedure in Fig. 2.2 works as a regularization technique on the parameters and helps the IPDLs to find accurate initialization for DeepHybrid's interval weights and biases. In contrast to fuzzy networks in the literature [103, 140, 154] which are randomly initialized, the proposed framework finds the optimal initialization for the fuzzy system's membership functions applying the generative IPDL stack; hence, the proposed architecture can more accurately address the wind data uncertainties.

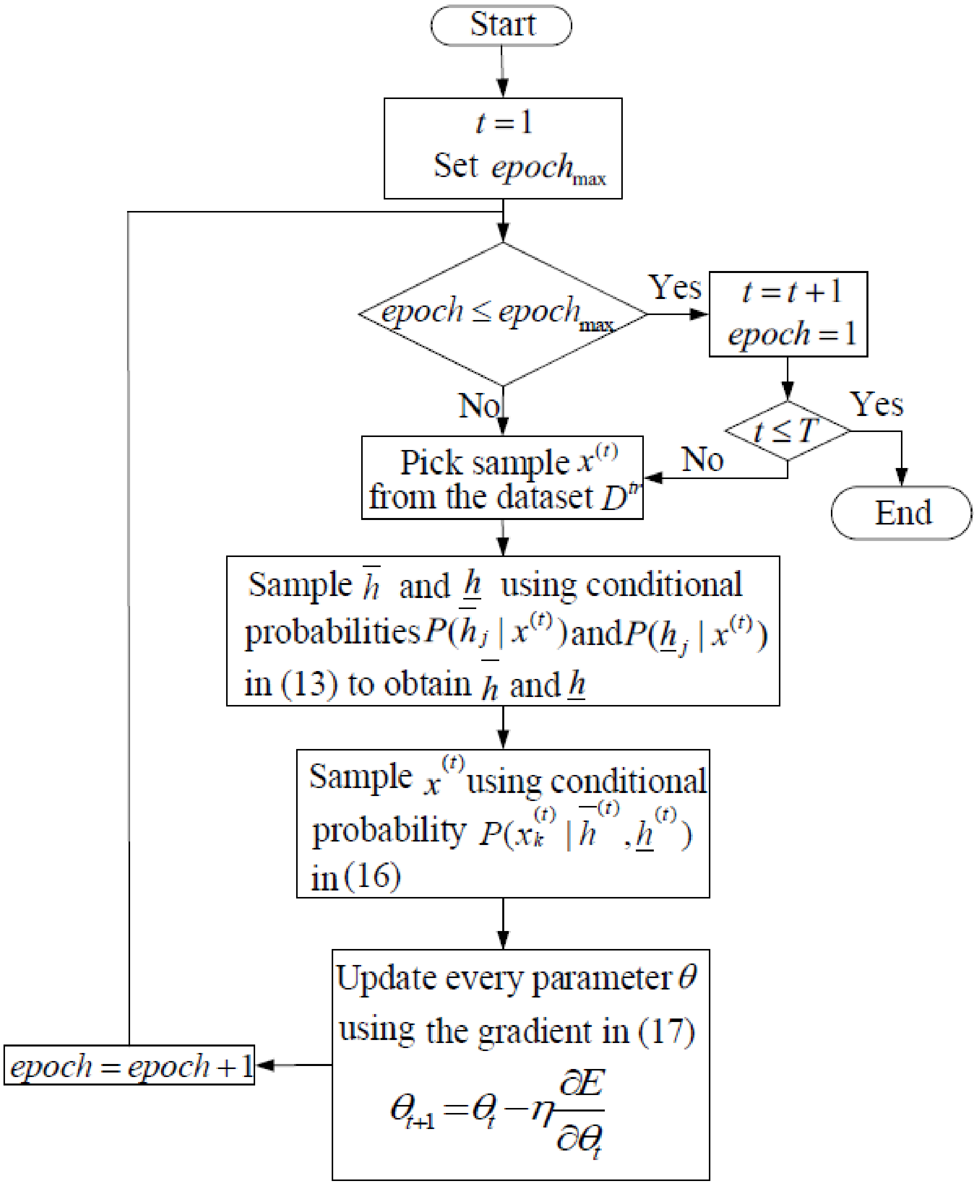

Start
$t = 1$
Set $epoch_{\max}$
$epoch \le epoch_{\max}$
Yes
$t = t + 1$
$epoch = 1$
Yes
$t \le T$
No
End
No
Pick sample $x^{(t)}$ from the dataset D^{tr}
Sample $\overline{h}$ and $\underline{h}$ using conditional probabilities $P(\overline{h}_j \mid x^{(t)})$ and $P(\underline{h}_j \mid x^{(t)})$ in (13) to obtain $\overline{h}$ and $\underline{h}$
Sample $x^{(t)}$ using conditional probability $P(x_k^{(t)} \mid \overline{h}^{(t)}, \underline{h}^{(t)})$ in (16)
Update every parameter θ using the gradient in (17)
$\theta_{t+1} = \theta_t - \eta \frac{\partial E}{\partial \theta_t}$
$epoch = epoch + 1$

Figure 2.2. Flowchart diagram of the training algorithm of IPDL.

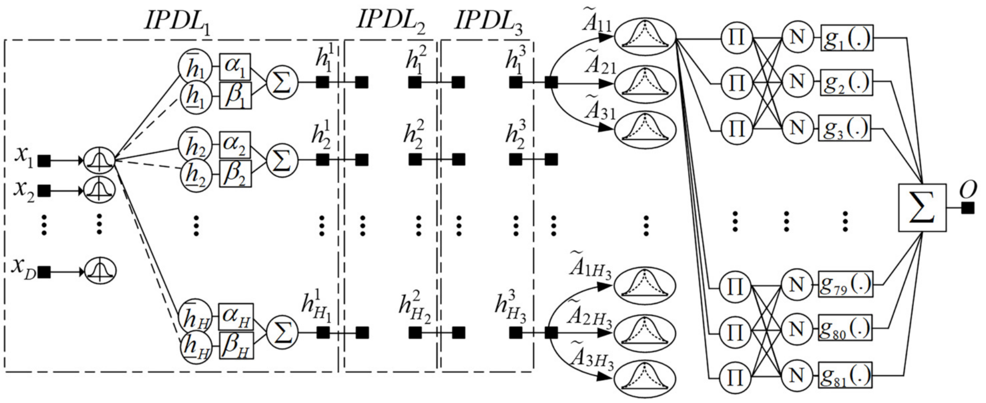


Figure 2.3. Structure of DeepHybrid with L=3.

2.4.2. Deep Belief Network for Real-valued data

As Fig. 2.3 depicts, h^L contains deep temporal features extracted by the IDBN. The input data (wind speed samples) is real-valued; therefore, the binary units in classic RBMs applied in [248] is not an effective choice due to assuming Bernoulli distribution for the time series signal. This motivates us to propose more complex real-valued input vector with Gaussian noise in order to model the wind speed distribution with higher estimation precision. As a result, the new energy function is defined by:

$$-\log P\left(x,\overline{h},\underline{h}\right) \propto E\left(x,\overline{h},\underline{h};\alpha,\beta\right) = -\left(\sum_{j=1}^{H}\sum_{k=1}^{D}\frac{x_j}{\sigma_j}\left(\alpha_j\overline{W}_{j,k}\overline{h}_j+\beta_j\underline{W}_{j,k}\underline{h}_j\right)\right) + \sum_{k=1}^{D}\frac{\left(x_k-c_k\right)^2}{2{\sigma_i}^2} - \left(\sum_{j=1}^{H}\alpha_j\overline{b}_j\overline{h}_j+\sum_{j=1}^{H}\beta_j\underline{b}_j\underline{h}_j\right) \tag{2.20}$$

where σ is the standard deviation vector of the Gaussian visible layer $\boldsymbol{x} =< x_1, x_2, \ldots, x_D >$. The conditional probability of the i-th visible unit, x_i, having value r given vectors $\overline{h}$ and $\underline{h}$ is computed by:

$$P\left(x_i=r \mid \overline{h},\underline{h}\right) = \frac{1}{\sqrt{2\pi}\sigma_i}\exp\left(-\frac{\left(r-c_i-\sigma_i(\sum_j\alpha_j\overline{h}_j\overline{W}_{j,i}+\beta_j\underline{W}_{j,i}\underline{h}_j)\right)^2}{2\sigma_i^2}\right) \tag{2.21}$$

In order to initialize the weight and bias parameters of the IDBN, log-likelihood of the model is maximized with $\sigma_i = 1$ to facilitate the training procedure applying Markov chain Monte Carlo to calculate expectations in (3.15).

2.4.3. Fuzzy Regression Method

In this study, four interval features, $h^L = \left[h_1^L, h_2^L, h_3^L, h_4^L\right]$ are extracted and utilized for the regression of the target function, i.e. the future time series values. The dimensionality reduction of DeepHybrid that compressed the $D-$dimensional input $\boldsymbol{x}$ into the 4-dimensional h^L, helps the FT2IS to avoid the curse of dimensionality which is a crucial issue for fuzzy systems. The extracted rough features are given to an FT2IS employed as a regression model to estimate future time series values. The proposed FT2IS is considered as a TSK system with type II membership functions in the premise part and crisp values in the consequent part. The differentiable property of Gaussian membership functions is the motivation for choosing such functions in the dissemination part. This characteristic helps the regression model to train the parameters of membership functions using updating algorithms that work based on the gradient of the loss function, such as the stochastic gradient descent applied in the last stage of training IDBN. Therefore, the whole model can be trained in an end-to-end fashion. For each feature h_k^L there are three Gaussian membership functions $\mu\widetilde{\mu}_{\tilde{A}_{jk}}(h_k^L)$ computed by:

$$\mu\widetilde{\mu}_{\tilde{A}_{jk}}\left(h_k^L\right) = e^{-\frac{1}{2}\left(\frac{h_k^L - c_{jk}}{\sigma\widetilde{\sigma}_{jk}}\right)^2} \tag{2.22}$$

Here, $\tilde{A}_{jk}$ represents the fuzzy type II sets of the j-th membership function for the k-th feature. Each membership function $\mu\widetilde{\mu}_{\tilde{A}_{jk}}$ is associated with the k-th representation unit h_k^L and the j-th rule, with an exact mean value c_{jk} and interval standard deviation $\sigma\widetilde{\sigma}_{jk}$. The i-th rule of this fuzzy structure is considered as:

$$\begin{aligned}&\textbf{IF } h_1^L \ is\ \tilde{A}_{j1} and h_2^L \ is\ \tilde{A}_{j2}\ and\ h_3^L\ is\ \tilde{A}_{j3}\ and\ h_4^L\ is\ \tilde{A}_{j4}\\&\textbf{THEN } g_i\left(h^L\right) = \alpha_0^i + \alpha_1^i h_1^L + \alpha_2^i h_2^L + \alpha_3^i h_3^L + \alpha_4^i h_4^L\end{aligned} \tag{2.23}$$

Here, $g_i\left(h^L\right)$ is a linear combination of featuresh^L= $\left[h_1^L, h_2^L, h_3^L, h_4^L\right]$ calculated in the consequent part. The regression output of the system considering the singleton fuzzifier and Center Average defuzzifier is computed as:

$$O = \int_{f^1 \in [\underline{f}^1, \overline{f}^1]} \cdots \int_{f^M \in [\underline{f}^M, \overline{f}^M]} \frac{1}{\frac{\sum_{i=1}^M f^i g_i}{\sum_{i=1}^M f^i}} \tag{2.24}$$

where $\underline{f}^i$ and $\overline{f}^i$ are defined by:

$$\begin{aligned}&\underline{f}^i\left(h^L\right) = \underline{\mu}_{\tilde{A}_{i1}}\left(h_1^L\right) * \underline{\mu}_{\tilde{A}_{i2}}\left(h_2^L\right) * \underline{\mu}_{\tilde{A}_{i3}}\left(h_3^L\right) * \underline{\mu}_{\tilde{A}_{i4}}\left(h_4^L\right)\\&\overline{f}^i\left(h^L\right) = \overline{\mu}_{\tilde{A}_{i1}}\left(h_1^L\right) * \overline{\mu}_{\tilde{A}_{i2}}\left(h_2^L\right) * \overline{\mu}_{\tilde{A}_{i3}}\left(h_3^L\right) * \overline{\mu}_{\tilde{A}_{i4}}\left(h_4^L\right)\end{aligned} \tag{2.25}$$

Here, * is a product operator utilized as the T-norm function. The firing of the i-th rule can be expressed as:

$$r_i = \frac{\underline{f}^i + \overline{f}^i}{\sum_{i=1}^M \underline{f}^i + \sum_{i=1}^M \overline{f}^i} \tag{2.26}$$

In this chapter, the Nie-Tan type reduction [138] is employed for the sake of its non-iterative solution. Hence, the regression output is computed by:

$$o = \frac{\sum_{i=1}^M (\underline{f}^i + \overline{f}^i) g_i - \sum_{i=1}^M \mathrm{sgn}(\mathrm{m}^i) \Delta \mathrm{f_i} g_i}{\sum_{i=1}^M (\underline{f}^i + \overline{f}^i) - \sum_{i=1}^M \mathrm{sgn}(\mathrm{m}^i) \Delta \mathrm{f_i}} \tag{2.27}$$

where $m^i = g_i - \frac{\sum_{i=1}^M \overline{f}^i g_i}{\sum_{i=1}^M \overline{f}^i}$ and $\Delta \mathrm{f_i} = \overline{f}^i - \underline{f}^i$.

2.4.4. Supervised End-to-end Training

After pre-training the IDBN, h^L vector is obtained and the K-Means algorithm is applied as an unsupervised clustering method on h^L activations to determine the initial mean values of the Gaussian membership functions. The number of clusters is set to the number of membership functions considered for h^L. In this method, the supervised error function is defined as:

$$J_{Sup} = \frac{1}{2}\sum_{t=1}^{T}(O_t - V_t)^2 + E_{Reg} \tag{2.28}$$

where T is the number of training samples, O_t is the DeepHybrid output and V_t is the target output of the t-th training sample. Here, E_{Reg} is the regularization error term defined by:

$$E_{Reg} = \frac{\lambda}{2}[\sum_{l=1}^{L}\sum_{i=1}^{H_l}\sum_{j=1}^{H_{l-1}}(\overline{W}_{i,j}^{l})^2 + (\underline{W}_{i,j}^{l})^2] \tag{2.29}$$

where $0 < \lambda < 1$ is the regularization coefficient. The Momentum technique is employed for SGD in order to increase the learning speed especially for the free parameters of real-valued IDBN. The gradients of J_{Sup} with respect to the free parameters in the consequent part are calculated as follows:

$$\frac{\partial J_{Sup}}{\partial \alpha_k^i} = \frac{\partial J_{Sup}}{\partial e_t}\frac{\partial e_t}{\partial O_t}\frac{\partial O_t}{\partial g_i}\frac{\partial g_i}{\partial \alpha_k^i}$$

$$\frac{\partial g_i}{\partial \alpha_k^i} = \begin{cases} 1 & k = 0 \\ h_k^L & k \neq 0 \end{cases} \tag{2.30}$$

where $e_t = O_t - V_t$ is the error of the t-th training sample. The gradients of the mean and the interval standard deviation parameters of the membership functions are computed as:

$$\frac{\partial J_{Sup}}{\partial c_{jk}} = \sum_{i=1}^{M^{H_L}}\frac{\partial J_{Sup}}{\partial e_t}\frac{\partial e_t}{\partial O_t}\left[\frac{\partial O_t}{\partial \underline{f}^i}\frac{\partial \underline{f}^i}{\partial \underline{\mu}_{\tilde{A}_{ik}}}\frac{\partial \underline{\mu}_{\tilde{A}_{ik}}}{\partial c_{jk}} + \frac{\partial O_t}{\partial \overline{f}^i}\frac{\partial \overline{f}^i}{\partial \overline{\mu}_{\tilde{A}_{ik}}}\frac{\partial \overline{\mu}_{\tilde{A}_{ik}}}{\partial c_{jk}}\right]$$

$$\frac{\partial J_{Sup}}{\partial \overline{\sigma}_{jk}} = \sum_{i=1}^{M^{H_L}}\frac{\partial J_{Sup}}{\partial e_t}\frac{\partial e_t}{\partial O_t}\frac{\partial O_t}{\partial \overline{f}^i}\frac{\partial \overline{f}^i}{\partial \overline{\mu}_{\tilde{A}_{ik}}}\frac{\partial \overline{\mu}_{\tilde{A}_{ik}}}{\partial \overline{\sigma}_{jk}}$$

$$\frac{\partial J_{Sup}}{\partial \underline{\sigma}_{jk}} = \sum_{i=1}^{M^{H_L}}\frac{\partial J_{Sup}}{\partial e_t}\frac{\partial e_t}{\partial O_t}\frac{\partial O_t}{\partial \underline{f}^i}\frac{\partial \underline{f}^i}{\partial \underline{\mu}_{\tilde{A}_{ik}}}\frac{\partial \underline{\mu}_{\tilde{A}_{ik}}}{\partial \underline{\sigma}_{jk}} \tag{2.31}$$

where M is the number of assigned membership functions to each FT2IS input; thus, M^{H_L} is the total number of type II rules defined in the proposed system. The derivatives $\frac{\partial \overline{\mu}_{\tilde{A}_{ik}}}{\partial c_{jk}}$, $\frac{\partial \overline{\mu}_{\tilde{A}_{ik}}}{\partial \overline{\sigma}_{jk}}$, and $\frac{\partial \underline{\mu}_{\tilde{A}_{ik}}}{\partial \underline{\sigma}_{jk}}$ are obtained applying the membership definition in (2.22). In order to update the rough features (interval weights and biases of IDBN), partial derivatives of J_{Sup} with respect to the upper- and lower-bound parameters of each rough unit are computed by similar formulation written in (3.19). One advantage of the proposed IPDL and FT2IS models is the differentiability with respect to the input; thus, the parameters of the whole deep network can be tuned end-to-end.

2.5. Simulation Results

2.5.1. Dataset

The wind speed time series measured for a wind site in Colorado is selected from the Western Wind Dataset [224] created by the National Renewable Energy Laboratory (NREL) and 3TIER. Weather Research and Forecasting (WRF) is applied in order to obtain the underlying dataset. WRF is a mesoscale NWP system used for atmospheric research and operational prediction tasks. The wind speed data available in the Western Wind Dataset has speed values from 2004 to 2006 with a 10-min interval between consecutive historical samples. DeepHybrid is trained using two experimental settings: *1) Offline training*: during this stage, the model is trained using the 2004 and 2005 time series data. In order to validate our model while tuning the parameters, 15% of the 2005 dataset is chosen uniformly from each season for the validation set. The training stops when the relative change in the validation RMSE is less than 5% in three consecutive training epochs. Only one hybrid model is trained and validated using the data of various seasons. Clustering wind data into different seasons and tuning distinct models each corresponding to a distinct data cluster is an extension of our proposed model that is considered as a future work. *2) Online training*: In this stage, DeepHybrid is already trained using the offline setting. During this stage, the model is evaluated using the new test samples of 2006 dataset. While testing, the neural network is trained at each step when a new unobserved sample is seen and the actual wind speed value is revealed.

For each year, there are 52560 wind speed values measured in 10-min intervals; therefore, sufficient data is available for training and testing the proposed approach. Fig. 2.4 depicts the highly varying wind speed time series of 2005. Several statistical tests such as Kolmogorov-Smirnov, Anderson-Darling, and Chi-Squared result that the wind speed data has Weibull probability distribution with 7.32 m/s mean value and 2.15 shape factor.

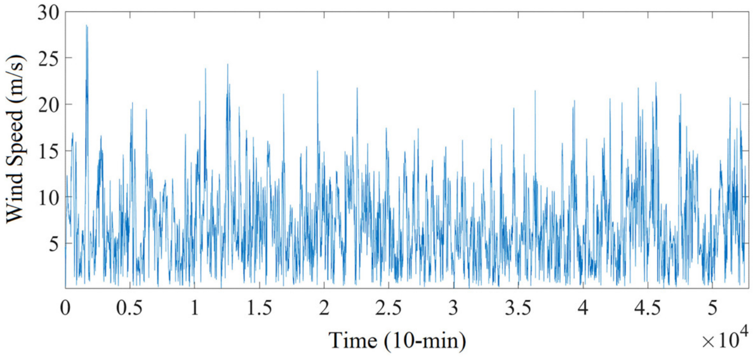


Figure 2.4. The 2005 wind speed values of Colorado wind site.

2.5.2. Input Variable Selection

Fig. 2.5 shows the MI for lag$l = 1$ to$l = 100$. It is shown that the correlation among the wind speed measurements decreases as the time-lag is increased. Wind speed data corresponding to the time-lags with MI greater than$\tau = 0.4$ are selected as the input set to highlight the correlation among the wind speed data. This would result in incorporating time-lags from $l = 1$ to $l = 24$. Suppose the model is at time $t$and the wind speed value of a future time horizon is going to be forecasted, the input set is a 24+23=47 dimensional vector $< v(t-23)$,$\Delta v(t-22)$,$v(t-22)$,...,$v(t) >$with wind speed sequential differences $\Delta v(t) = v(t) - v(t-1)$.

2.5.3. Evaluation Criteria

The Root Mean Square Error (RMSE) and the Mean Absolute Percentage Error (MAPE) are employed in order to evaluate the results obtained by the proposed model. The RMSE of M test

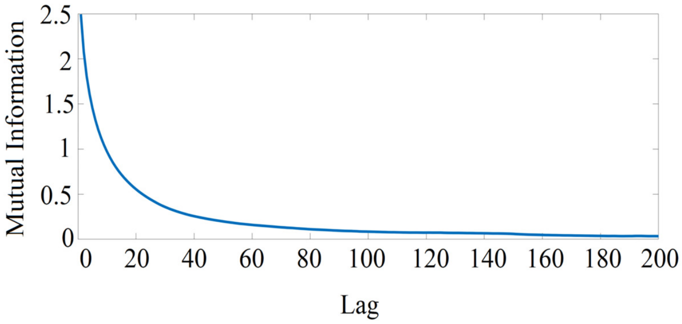


Figure 2.5. Mutual Information of various time-lags of 2005 dataset.

samples is calculated as:

$$RMSE = \sqrt{\frac{1}{M}\sum_{n=1}^{M} err(n)^2} \tag{2.32}$$

and the MAPE is expressed as:

$$MAPE = \frac{1}{M}\sum_{n=1}^{M}\left|\frac{err\,(n)}{\mathrm{t\,(n)}}\right| \times 100\% \tag{2.33}$$

where $err\,(n) = t\,(n) - o(n)$ is the test error for the n-th sample, $t\,(n)$ is the target value and $o(n)$ is the output for time step n.

2.5.4. Simulation Settings

The proposed DeepHybrid model takes a 47-dimensional input vector resulted by the Mutual Information as the feature selection algorithm. The number of activation units at each layer is chosen from the set $\varphi = \{5, 10, 15, \ldots, 45\}$ with five as the gap between consecutive members. The IDBN can contain 2 up to 5 IPDLs as the initial hidden layers of the hybrid model. The iteration number of the underlying experiment is an important factor to avoid overfitting. Here, a maximum number of 80 iterations is considered to train our model. Also, a stopping criterion for the training procedure is satisfied when the validation process varies less than a threshold value equal to 0.05 for 5 epochs. This validation procedure can help the model to avoid overfitting since the performance of the hybrid structure is evaluated by the unseen data. The learning rate η and the coefficient of the momentum term γ are set to 0.5. The weight decay parameter λ for the L2 regularization is chosen from the set $= \{0.2, 0.3, 0.4, 0.5, 0.6\}$. The optimal λ corresponds to the least validation error at the end of the training process.

In order to determine the optimal structure of the IDBN, a random search on the fix set φ is done. The optimal model is selected according to the average validation error in 100 runs. Grid search and heuristic search algorithms could obtain more accurate estimations, however, these methods lead to high computational complexity. Fig. 2.6 shows the validation RMSE for 1-hour, 10-hour, and 24-hour ahead wind speed forecasts with the increase in the number of IPDLs. As shown in this figure, IDBN with two IPDLs yields the minimum error rate for 10-min ahead prediction. This number is increased to three when the time horizon is extended to 10 hours. As the complexity of the forecasting task is increased, the optimal number of IPDLs grows. For 24-hour ahead forecasts, an IDBN with four IPDLs leads to the least error rate. Having more IPDLs than the optimal choice results in the overfitting issue while considering fewer hidden layers decreases the generalization capability of the DeepHybrid. The vanishing gradient problem can also grow the validation RMSE while increasing the number of IPDLs since the supervised error function cannot be satisfactorily informative when having large numbers of layers.

In order to compare the proposed IPDL with the classic DBN [248] recently introduced in the literature for multi-step predictions, a DBN with Bernoulli RBMs is trained to replace the proposed IDBN model in the DeepHybrid. As shown in Fig. 2.6, IDBN finds architectures with better performance on the validation set compared to the DBN model. Moreover, the DBN structure needs more number of hidden layers. For instance, in 24-hour ahead prediction task, the Bernoulli DBN requires 5 generative models (RBMs) to reach the optimal solution while the proposed architecture consists of 4 IPDL models. Hence, the computational burden of the deep belief network is decreased while better accuracy is obtained utilizing the proposed interval distribution learning methodology.

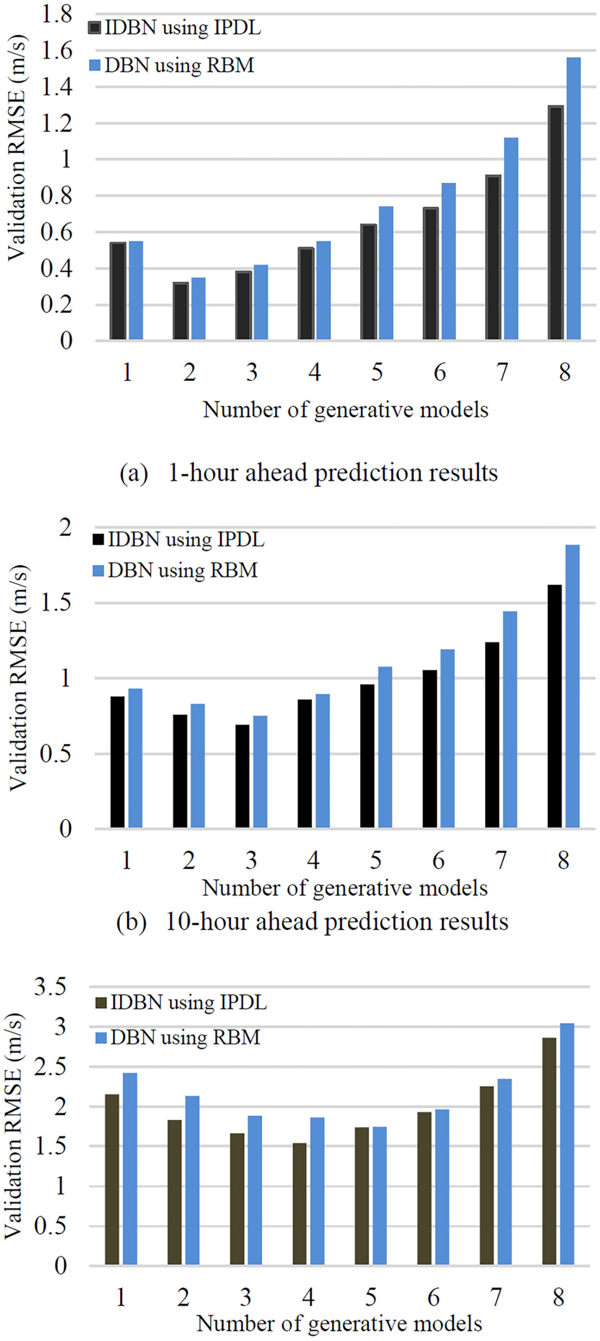
The FT2IS regression model contains four input variables that are resulted by the proposed IDBN. For each input, there are three Gaussian membership functions. Thus, the number of rules is $3^4 = 81$. K-means algorithm with three clusters is applied to the IDBN features in order to determine the initial mean values of the membership functions. The standard deviations of these functions are chosen randomly in [0.01,0.2] as well as the free parameters of the consequent part.

2.5.5. Numerical Results and Comparisons

In this study, the performance of our proposed DeepHybrid method is compared with the Persistence (PR) model as a classic benchmark for ultra-short-term and short-term wind speed forecasting. Moreover, the proposed model is compared with both single-model and hybrid approaches in the recent literature.

Single-model methods apply a single regression architecture to perform the prediction task. In order to show the effect of deep feature learning on wind data regression tasks, shallow ANN-based methodologies including Feed-forward Neural Network (FFNN) [64, 134, 191], Time Delay Neural Network (TDNN) [28], and Nonlinear Autoregressive Neural Network (NARNN) [12] are compared with our proposed approach. Very recent literature [111, 248], proposed Stacked Auto-encoders and Deep Belief Networks and compared their deep ANNs with a variety of AI methodologies such as FFNN, Support Vector Regression (SVR) [83], NARNN [12], and Adaptive Neuro-Fuzzy Inference System [154]. Both SAE and DBN showed significant improvements compared to shallow AI models. Moreover, in [150], it is shown that DBN outperforms the Persistence model and statistical models including Auto-regressive techniques. Motivated by the significant accuracy improvement of deep learning approaches, i.e., DBN and SAE, our proposed model is compared with both of these approaches as very recently proposed state-of-the-art methodologies in this research area.

The hybrid models make use of multiple wind feature extraction and regression methods in order to increase the prediction accuracy. In this study, DeepHybrid architecture is compared to the recently proposed hybrid E-GA-APSO-WNN model [224] that applies Ensemble Empirical Mode Decomposition (EEMD) for noise reduction in wind speed time series data, as well as Genetic Algorithm (GA) incorporated with Particle Swarm Optimization (APSO) as an optimization method to tune the parameters of a Wavelet Neural Network (WNN). Moreover, our work is compared with the hybrid model proposed for short-term wind speed forecasting in [247] that applied



(a) 1-hour ahead prediction results

(b) 10-hour ahead prediction results

(c) 24-hour ahead prediction results

Figure 2.6. RMSE of validation for 10-min ahead forecasting with the increase in the number of IPDLs.

a compound structure of Extreme Learning Machine (ELM) based on feature selection and parameter optimization using hybrid backtracking search algorithm (HBSA). The proposed ELM-HBSA model in [247] effectively captures the nonlinear characteristics of wind speed signals and outperforms ARIMA and SVR-based forecasting models.

2.5.5.1. Deep Learning vs. Shallow Feature Learning:

Tables 2.1 and 2.2 show the RMSE and MAPE criteria for 10-min up to 3-hour ahead wind speed forecasting. The performance of our proposed approach is compared to both shallow and deep single-model methods. The RMSE is generally increased with the extension of the forecasting time horizon. MAPE criterion has also the same behavior. The persistence method yields accurate results for short-term predictions. The RMSE of PR for 10-min predictions is 0.625 m/s which is increased to 2.785 m/s for 3-hour ahead forecasts. MAPE result of PR for 10-min predictions is 10.983 which reaches to 30.174 in the 3-hour ahead forecasting task. Therefore, applying PR for longer term predictions cannot yield reliable performance.

FFNN obtains better results compared to PR. This improvement is more significant for larger forecasting time horizons. FFNN outperforms PR with 7.04% RMSE improvement in 10-min forecasts. This improvement reaches to 24.20% for 3-hour ahead forecasts. The poor performance of PR in 3-hour ahead predictions is due to the simple smoothness assumption of this model. TDNN and NARNN models both outperform FFNN since these approaches can model the sequential attributes of time series data while capturing the temporal characteristics of the data. TDNN has 5.80% and 8.32% RMSE and MAPE improvements compared to FFNN, respectively. NARNN also outperforms FFNN with 12.56% and 11.60% better RMSE and MAPE results, respectively.

NARNN is the best shallow neural architecture compared to the FFNN and TDNN models. Comparing NARNN with SAE as a deep network, shows the better performance of deep structures compared to the conventional shallow neural networks. SAE has 7.23% RMSE and 17.82% MAPE improvements over NARNN. These improvements are further increased to 8.99% and 22.44% for the RMSE and MAPE results when the DBN model is applied. The better accuracy demonstrates the better generalization of DBN and SAE due to having more numbers of non-linear hidden layers

Table 2.1. RMSE of forecasting methods for different time horizons.

	Time Step				
Method	10-min	30-min	1-hr	2-hr	3-hr
PR	0.625	1.233	1.670	2.230	2.785
FFNN	0.581	0.921	1.560	1.860	2.111
TDNN	0.508	0.881	1.514	1.831	1.950
NARNN	0.488	0.763	1.470	1.622	1.876
SAE	0.452	0.759	1.350	1.518	1.612
DBN	0.431	0.753	1.339	1.501	1.583
Deep Hybrid	**0.419**	**0.742**	**1.280**	**1.470**	**1.564**

Table 2.2. MAPE of forecasting methods for different time horizons.

	Time Step				
Method	10-min	30-min	1-hr	2-hr	3-hr
PR	10.983	15.054	21.306	26.560	30.174
FFNN	10.241	10.763	16.665	14.862	18.684
TDNN	9.872	10.120	14.507	14.521	15.553
NARNN	9.133	9.817	14.260	13.855	15.469
SAE	6.441	7.509	13.129	11.352	13.903
DBN	6.024	6.730	12.307	11.770	12.688
Deep Hybrid	**4.108**	**4.632**	**8.814**	**9.905**	**11.126**

and capturing input distribution which helps these models to provide more precise forecasts for the wind speed as a highly varying target function. DBN yields better accuracy in comparison to the SAE proposed in [12] since the unsupervised feature extraction is done by generative models that can better capture the input distribution compared to SAE as a discriminative learning approach.

The proposed hybrid model obtains more accurate results when compared to DBN and SAE. This model decreases RMSE by 2.7% and MAPE by 23.90% when compared to the DBN proposed in [248]. The improvements of the hybrid model over DBN are due to: 1) The use of proposed interval features and FT2IS for regression to handle uncertainties that exist in the wind speed data by capturing interval knowledge from the time series, and 2) Applying real-valued input units to estimate the wind speed distribution with higher precision when compared to SAE and DBN with Bernoulli input variable assumptions.

Fig. 2.7 shows the resulted forecasting values of NARNN and DBN with the actual wind speeds for 3-hour predictions of 72 samples from May 3rd, 2006. As it is shown, DBN improves the results obtained by NARNN due to learning wind data probability distribution by generative modules. Fig. 2.8 demonstrates the hourly performance comparison of the DBN with Bernoulli RBMs proposed in [248] and our proposed IDBN model for the test samples of August 25th, 2006. Both approaches are generative models that aim to learn the distribution of the input data. As shown in this plot, our deep learning approach yields more accurate outcome compared to the Bernoulli DBN. In this diagram, the largest absolute error value of the DBN is 0.98 m/s while our model, IDBN, decreased this error to 0.43 m/s due to the following reasons: 1) The proposed IPDL model learns an interval latent representation from the data to model the probability density of the data vectors; hence, our proposed generative model is more robust compared to the classic DBNs including [248] that assume crisp latent representations. 2) In contrast to the DBN which utilizes binary input units, our model leverages real-valued input variables designed to learn distributions from the real-valued wind data. The proposed input units more accurately estimate the conditional probability of wind speed values given hidden units computing $P(x|\overline{h},\underline{h})$ in (2.21). The DBN naively assumes a Bernoulli distribution for all input variables.

2.5.5.2. *Comparison between Hybrid Methods*

Tables 2.3 and 2.4 show the RMSE and MAPE of our proposed DeepHybrid model and two hybrid baselines: E-GA-APSO-WNN and ELM-HBSA. Both baselines apply signal decomposition for noise reduction while our DeepHybrid model captures interval knowledge from the data in order to handle wind data uncertainties. The E-GA-APSO-WNN has a better average performance over the ELM-HBSA, with a 10.39% and 13.61% MAPE improvements for 1-hour and 3-hour ahead predictions, respectively. The DeepHybrid model outperforms E-GA-APSO-WNN in all time horizons. DeepHybrid model improves the MAPE result of the E-GA-APSO-WNN by 45.96% for ultra-short-term 10-min ahead forecasts. This significant improvement is due to the automatic unsupervised deep feature extraction of the proposed deep network. For 3-hr predictions, DeepHybrid architecture obtains 21.19% and 8.79% less MAPE compared to ELM-HBSA and E-GA-APSO-WNN, respectively. Besides deep feature extraction, an advantage of the proposed DeepHybrid model over other hybrid methods is automatically capturing the interval knowledge from the wind data in order to handle the uncertainties, rather than applying error-prone feature selection and signal decomposition techniques to handle the noise.

Fig. 2.9 and Fig. 2.10 depict the RMSE and MAPE results of all single-model and hybrid approaches for extended time horizons from 1-hr ahead to 24-hr ahead predictions, respectively. As shown in these figures, the shallow single-model architectures, i.e. FFNN, TDNN, and NARNN, are dominated by single-model deep learning models, SAE and DBN, in all time horizons. The SAE and DBN have relatively good accuracy compared to hybrid methods for 1-hr to 7-hr ahead predictions; however, for larger prediction time steps, the hybrid methodologies have a remarkable improvement in both RMSE and MAPE. Our Deep Hybrid model outperforms both the E-GA-APSO-WNN and ELM-HBSA significantly when the time horizon exceeds 5 hours. This leads to the noticeable gaps between DeepHybrid and other hybrid approaches in the RMSE and MAPE plots.

2.5.5.3. *Effect of Noise on the Performance*

In order to show the effect of uncertainties in the wind speed data on the performance of

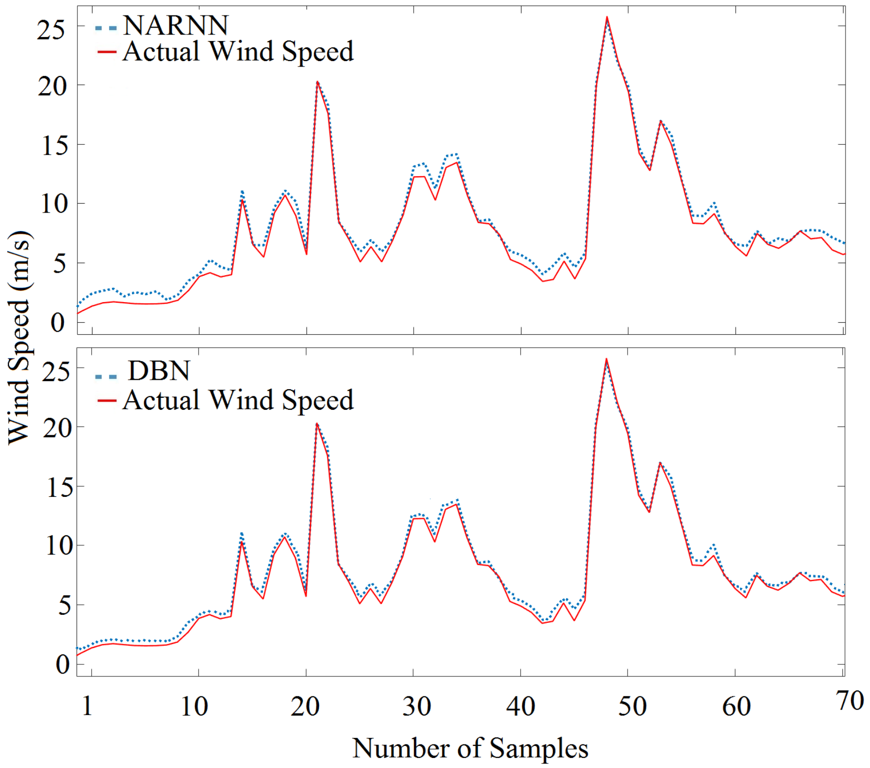


Figure 2.7. The 3-hour ahead prediction outputs of NARNN and DBN with the actual wind speeds from May 3rd 2006.

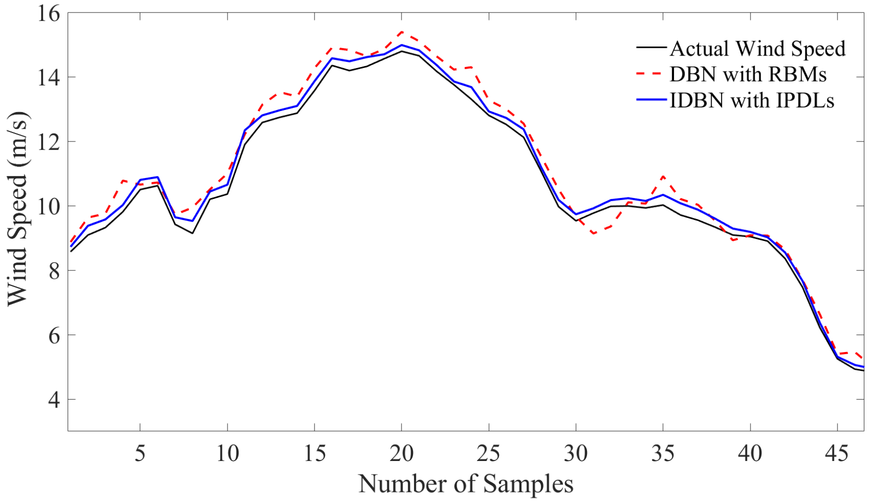


Figure 2.8. Comparison of the 1-hour ahead prediction outputs of DBN and DeepHybrid model for the test samples of August 25th 2006.

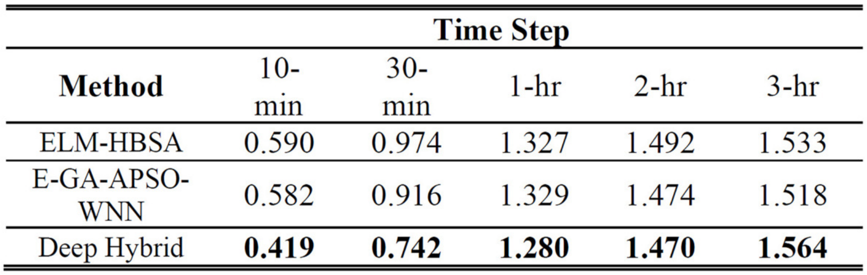
our proposed model, two extensions of DeepHybrid model are designed as our baselines; 1- DeepHybrid$_{dense}$: The first baseline methodology replaces the IPDL models with generative RBMs. Our proposed DeepHybrid method is compared with this model in order to investigate the effect of interval feature learning.2- DeepHybrid$_{TypeI}$: The second baseline replaces the FT2IS regression model of DeepHybrid with a Fuzzy Type I inference system. Our proposed DeepHybrid methodolgy is compared with this model in order to investigate the effect of capturing interval Type II rules from the deep network.

The baselines are compared to our DeepHybrid model under various noise conditions. Following the robustness experiments in [111] for wind prediction models, a Gaussian noise $\mathrm{Gauss}\,(,{}^{2})$ is considered with mean $= 0$ and standard deviation $= 0.1\ \mathrm{v}$ for each wind speed test sample v. Fig. 2.11 depicts the Box-and-Whisker plot of the absolute hourly prediction error of DeepHybrid as well as DeepHybrid$_{dense}$ and DeepHybrid$_{TypeI}$. The resulting minimum, median, and maximum values of boxes corresponding to DeepHybrid are less than the corresponding values for both DeepHybrid$_{dense}$ and DeepHybrid$_{TypeI}$. As shown in Fig. 2.11, applying our Rough feature extraction layer in the DeepHybrid model, leads to 7.08$\%$ less maximum absolute error for hourly predictions compared to the DeepHybrid$_{dense}$ extension of our model, which contains no IPDL. Moreover, the comparison of hourly prediction for DeepHybrid with DeepHybrid$_{TypeI}$ shows that capturing interval knowledge from the wind data using interval memberships in the FT2IS of our proposed approach improves the prediction accuracy while degrading the maximum of absolute error by 9.61$\%$.

2.5.5.4. IPDL vs. RBM Comparison

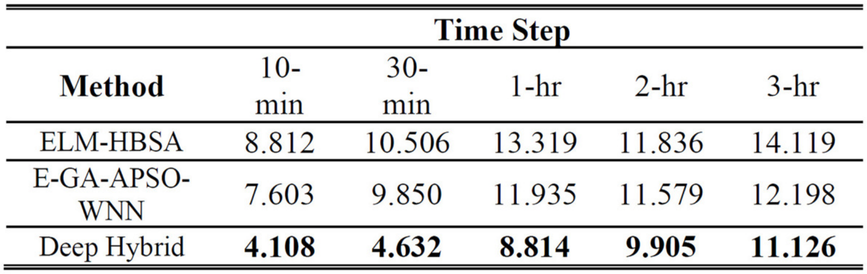
In order to have a fair comparison with respect to the prediction accuracy of the proposed IPDL methodology and the RBM [248], as the state-of-the-art deep generative model, we define two baselines, $IPDL_{reg}$and RBM_{reg}. The $IPDL_{reg}$ model is a single-model version of DeepHybrid including a stack of IPDLs and a linear regression model at the top. RBM_{reg} is the similar structure built using restricted Boltzmann machines instead of the IPDL models. The number of hidden units and number of layers are determined using similar approach in section V – D.

Table 2.3. RMSE of forecasting methods for different time horizons.



Method	Time Step				
	10-min	30-min	1-hr	2-hr	3-hr
ELM-HBSA	0.590	0.974	1.327	1.492	1.533
E-GA-APSO-WNN	0.582	0.916	1.329	1.474	1.518
Deep Hybrid	**0.419**	**0.742**	**1.280**	**1.470**	**1.564**

Table 2.4. MAPE of forecasting methods for different time horizons.



Method	Time Step				
	10-min	30-min	1-hr	2-hr	3-hr
ELM-HBSA	8.812	10.506	13.319	11.836	14.119
E-GA-APSO-WNN	7.603	9.850	11.935	11.579	12.198
Deep Hybrid	**4.108**	**4.632**	**8.814**	**9.905**	**11.126**

Fig. 2.12 depicts the performance comparison of the $IPDL_{reg}$ with RBM_{reg} in terms of the test RMSE. As previously shown in Fig. 2.6, when the forecasting time horizon is extended, more complex networks with larger number of latent representation layers are needed to achieve high accuracy; however, having too many layers will decline the performance of both models due to the vanishing gradients. Here, in Fig. 2.12, the $IPDL_{reg}$ obtains better performance (lower test RMSE) in a wider region of the structure search space, that is, the $IPDL_{reg}$ is less sensitive to increasing or decreasing the number of layers. However, the performance of RBM_{reg} is more dictated by the number of hidden layers. Moreover, the higher accuracy of $IPDL_{reg}$ compared to RBM_{reg} shows the superiority of the proposed interval distribution learning methodology compared to restricted Boltzmann machines.

2.5.5.5. *Running Time Analysis*

Fig. 2.13 depicts the offline training time of the DeepHybrid using batch gradient descent with different batch sizes. The model is implemented on a multi-GPU computer system with two

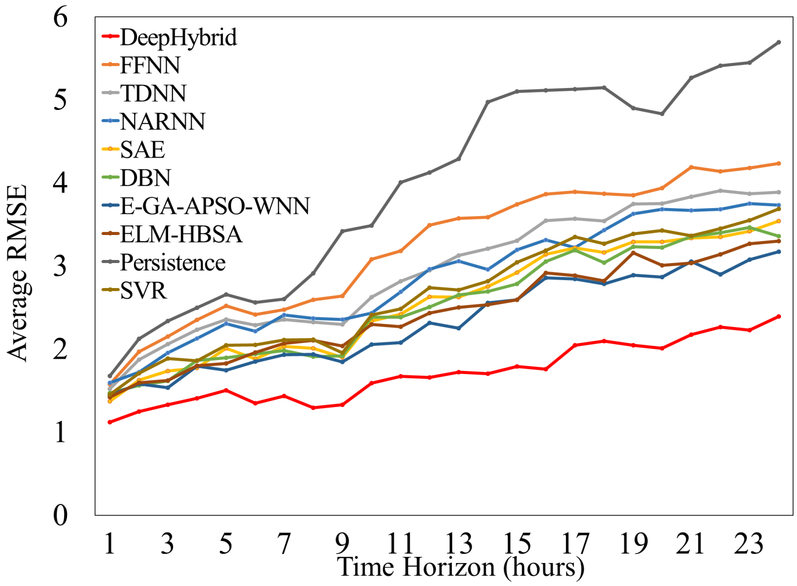


Figure 2.9. Average RMSE results of DeepHybrid model with all baselines for 1-hr up to 24-hr prediction tasks.

NVIDIA GTX980 graphics cards and a 4.2 GHz Quad-Core Processor. The Tensorflow framework [4] is utilized on the computer system to speed up the proposed deep learning algorithm using GPUs. As shown in Fig. 2.13, the offline training time increases as the time horizon is extended since the number of hidden layers grows with the complexity of the corresponding regression problem. The proposed approach is tuned in a time period less than 10 minutes for 10-min ahead forecasts; hence, our deep learning framework is applicable to short-term wind speed prediction tasks. For the applications with time horizons smaller than 10 minutes, the model can be tuned offline using the historical data before being utilized for real-world applications.

Fig. 2.14 shows the average running time of a single update, that is, observing a new test sample and updating the model in an online fashion. As shown in this figure, there is negligible change in the online running time of the model as the length of the forecasting horizon is changed. The online training time of 10-min ahead prediction is 0.187 seconds which is much lower than the corresponding time horizon; hence, the proposed model can be efficiently utilized for the short-term prediction problems.

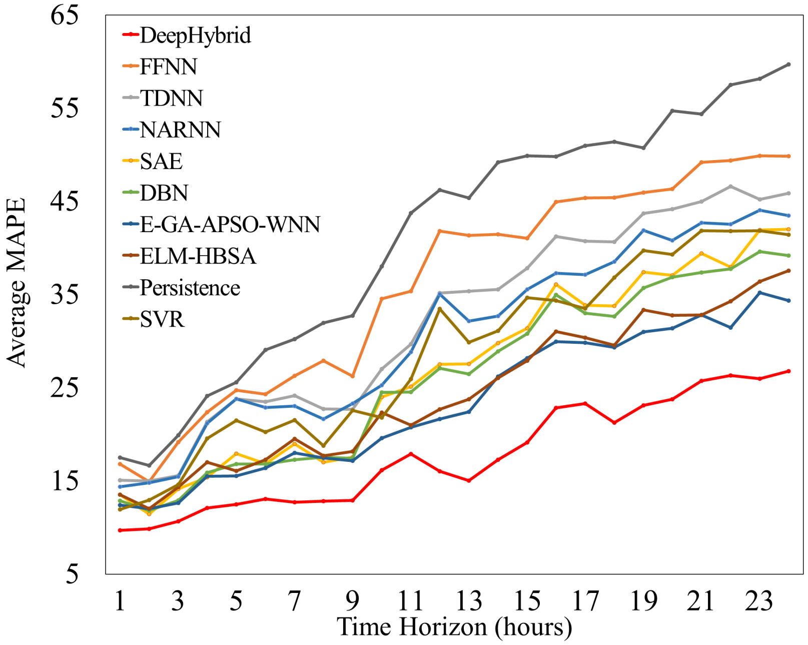


Figure 2.10. Average MAPE results of DeepHybrid model with all baselines for 1-hr up to 24-hr prediction tasks.

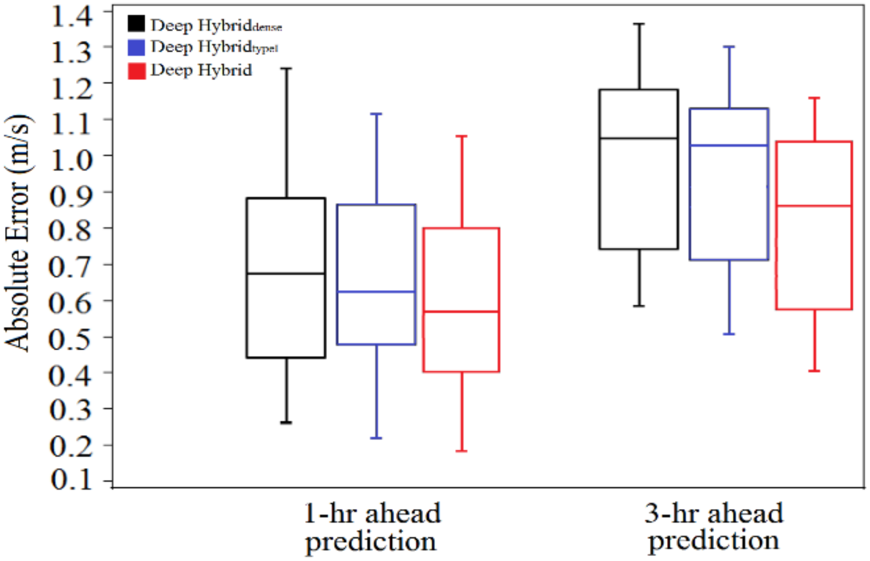


Figure 2.11. Box-and-Whisker plot of the absolute hourly prediction error of DeepHybrid, DeepHybrid$_{dense}$, and DeepHybrid$_{TypeI}$.

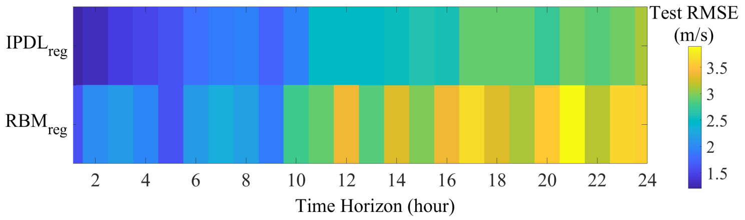


Figure 2.12. Box-and-The test RMSE comparison of $IPDL_{reg}$ and $RBM_{reg}$for multiple time horizons.

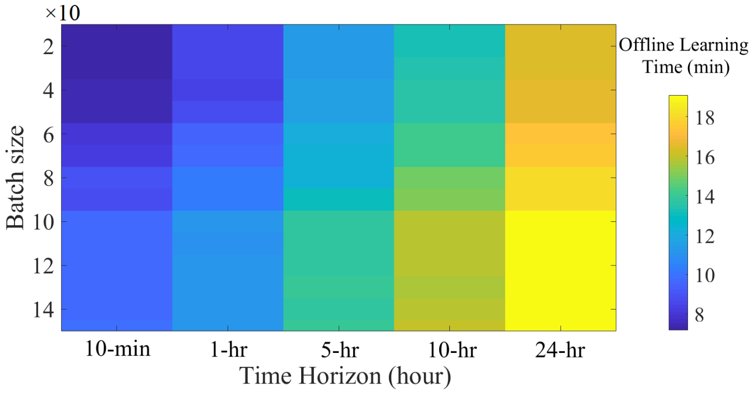


Figure 2.13. The offline training time of DeepHybrid using batch gradient descent with various batch sizes.

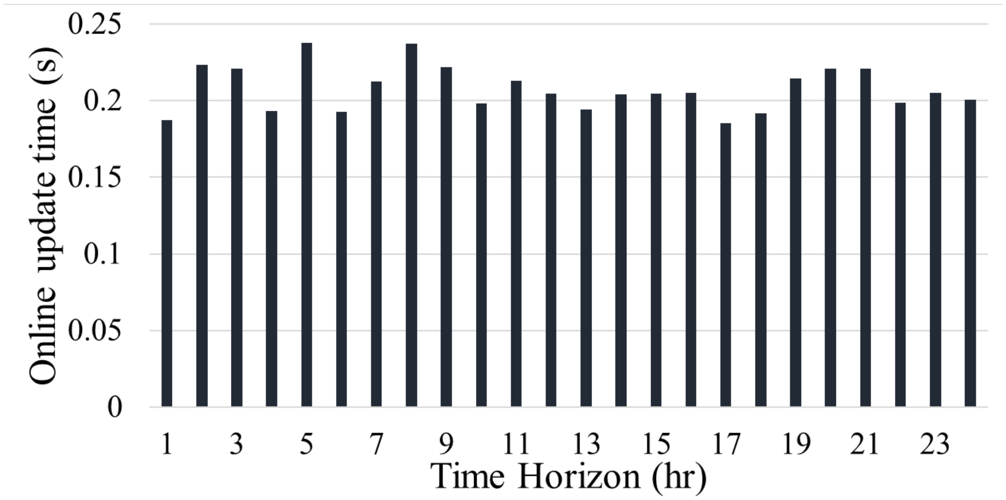


Figure 2.14. The online training running time of DeepHybrid.

2.6. Conclusions

In this chapter, a hybrid wind forecasting model based on Deep Learning, Rough set theory and Fuzzy Set theory is proposed. A generative unsupervised probability distribution learning model is designed based on the Restricted Boltzmann Machines with real-valued inputs in order to learn powerful features from the wind data probability distributions. The Rough Set theory is incorporated with deep generative models to design the proposed interval distribution learning architecture. Moreover, the inference and learning algorithms of the proposed architecture are discussed. An interval deep belief network with upper-bound and lower-bound parameter estimations is further devised based on the proposed distribution learning model and the fuzzy type II inference systems. The fuzzy system is applied for the supervised prediction of the underlying target function using the features obtained from the proposed IPDL. The Differentiability of the IPDL and FT2IS leads the model to tune the whole parameters in an end-to-end fashion using supervised desired output signals. The Generalization capability of the unsupervised feature learning method combined with the noise invariant feature extraction of rough layers and the robust fuzzy rule learning method, leads to accurate target function approximation for time series prediction. Unlike previously proposed deep networks which assumed Bernoulli input variables, the proposed model leverages real-valued input units that are suitable for learning powerful features from the real-valued wind speed time series. Simulation results show significant improvement of the proposed IPDL model and its novel learning algorithm compared to recently proposed shallow and deep architectures, including DBN, as well as recent hybrid methodologies. Moreover, the effect of the proposed methodology in handling data uncertainties is investigated. It is shown that the proposed IPDL can obtain more robust deep features compared to RBM due to using real-valued input variables as well as interval upper-bound and lower-bound parameters. The Planned future work and improvements include learning arbitrary activation functions for the input units of generative deep models, learning the contribution factors of the upper-bound and lower-bound latent units using Bayesian machine learning techniques, and devising Ensemble architectures using the proposed learning methodology as well subspace clustering techniques in order to learn time series features with higher diversity.

Chapter 3

Convolutional Graph Auto-encoder: A Deep Generative Neural Architecture for Probabilistic spatiotemporal Solar Irradiance Forecasting

Machine Learning on graph-structured data is an important and omnipresent task for a vast variety of applications including anomaly detection and dynamic network analysis. In this chapter, a deep generative model is introduced to capture continuous probability densities corresponding to the nodes of an arbitrary graph. In contrast to all learning formulations in the area of discriminative pattern recognition, we propose a scalable generative optimization/algorithm theoretically proved to capture distributions at the nodes of a graph. Our model is able to generate samples from the probability densities learned at each node. This probabilistic data generation model, i.e. convolutional graph auto-encoder (CGAE), is devised based on the localized first-order approximation of spectral graph convolutions, deep learning, and the variational Bayesian inference. We apply our CGAE to a new problem, the spatiotemporal probabilistic solar irradiance prediction. Multiple solar radiation measurement sites in a wide area in northern states of the US are modeled as an undirected graph. Using our proposed model, the distribution of future irradiance given historical radiation observations is estimated for every site/node. Numerical results on the National Solar Radiation Database show state-of-the-art performance for probabilistic radiation prediction on geographically distributed irradiance data in terms of reliability, sharpness, and continuous ranked probability score.

3.1. Introduction

In recent years, the rapid exhaustion of fossil fuel sources, the environmental pollution concerns, and the aging of the developed power plants are considered as crucial global concerns. As a consequence, the renewable energy resources including wind and solar have been rapidly integrated into the existing power grids. The reliability of power systems depends on the capability of handling expected and unexpected changes and disturbances in the production and consumption, while maintaining quality and continuity of service. The variability and stochastic behavior of photovoltaic (PV) power caused by the solar radiation uncertainty lead to major challenges including voltage fluctuations, as well as local power quality and stability issues [111, 214, 218]. Hence, accurate solar irradiance forecasting for PV estimation is required for effective operation of power grids [96]. The studies in the area of solar irradiance and PV power forecasting are mainly categorized into four major classes:

1) The persistence model is applied as a baseline that assumes the irradiance values at future time steps is equal to the same values at the forecasting time. Due to such a strong smoothness assumption, the persistence scheme is only effective for intra-hour applications [218].

2) Physical models employ physical processes to estimate the future solar radiation values using astronomical relationships [81], meteorological parameters, and numerical weather predictions (NWPs) [176]. In [130], an hourly-averaged day-ahead PV forecasting approach is presented based on least squares optimization of NWPs using global horizontal irradiance (GHI) and the zenith angle. Some NWPs make use of the clear sky radiation modeled by earth-sun geometry [166] or panel tilt/orientation along with several meteorological parameters such as temperature or wind speed [147]. Other works apply cloud motion vector (CMV) frameworks [43] for accurate short-term predictions, using static cloud images [157], satellite images [92], or the sensor networks [21].

3) Statistical and Artificial intelligence (AI) techniques are recently presented for a number of solar irradiance and PV power estimation/regression problems. As discussed in [13], the non-stationary and highly nonlinear characteristics of solar radiation time series lead to the superiority of AI approaches over the traditional statistical models. Machine learning algorithms are employed as target function approximators, to estimate future solar irradiance or PV power. Highly nonlinear regression methodologies including ANNs [42, 113] and support vector machines/regression (SVM/R) [132] have been employed for short-term purposes. [132] presents a benchmarking of supervised neural networks, Gaussian processes and support vector machines for GHI predictions. In [217], [133] a bootstrapping approach is presented to estimate uncertainties involved in the prediction of wind/solar time series. Here, a number of Extreme Learning Machine (ELM) ANNs are

trained as regression models using resampled training data. The uncertainties in solar/wind data and the model uncertainties are modeled as two classes of uncertainties to provide probabilistic predictions. This model has low generalization capability as both uncertainties are associated with a strong prior knowledge that forces the uncertainties to be Gaussian. [255] employs k-nearest neighborhood (k-NN) method to find days with similar weather condition. Kernel Density Estimation (KDE) is further applied to estimate the probability density function (PDF) of PV for the neighbors of k-NN. [211] provides a comprehensive review of non-parametric methods that employ k-NN to find the expected value of their assumed probability distribution functions for solar irradiance and PV forecasting. [39] applies k-NN for short-term predictions with less than 20-min ahead horizons. Also, [87] employs k-NN and gradient boosting with various meteorological measurements such as surface pressure, total cloud cover, and relative humidity for 24-hr ahead forecasts.

Quantile Regression (QR) is another statistical method employed in non-parametric prediction models. In recent literature, QR is well-studied for the estimation of statistical parameters (e.g. mean and variance) of predefined probability distributions for future solar values [211]. In [71], the ELM neural network utilizes a QR-based parameter estimation for hourly solar predictions. Also, [215] employs the combination of QR and ELM for very short-term applications with 5-min horizon length. In [131], a probabilistic prediction model is proposed based on linear QR, combining the point prediction obtained by a deterministic forecasting approach with the information retrieved from ground measurements. Moreover, QR is recently utilized as a non-parametric model in combination with physical methods [211]. In [70], a combination of QR and NWP is presented for daily predictions. Furthermore, [101] proposes an intra-day prediction approach based on multiple QR in combination with the radial basis functions and the alternating direction method of multipliers.

As discussed in [54, 115], fuzzy logic has been recently applied to capture the uncertainties exits in solar datasets. In [36], fuzzy systems are incorporated with neural networks to accurately estimate the real values of future solar irradiance under different sky and temperature conditions. Moreover, [152] presents a fuzzy clustering algorithm to find days with similar irradiance patterns.

The solar data corresponding to similar days is further fed to an ELM optimized by Genetic Algorithm (GA) in order to compute daily irradiance predictions. Evolutionary algorithms including GA, Ant Colony [211], and Particle Swarm Optimization [54] help fuzzy systems and ELM to find near-optimal solutions by avoiding erroneous parameter settings caused by poor local optima solutions.

Bayesian approaches have been widely applied to solar prediction problems. In [23], two advanced probabilistic models are proposed based on Bayesian inference for short-term PV prediction. Moreover, new probabilistic indices are presented to compare probabilistic approaches in such a way that the estimated PV values are partially anticipated by the forecasters in their quality-assessment procedures. [16] presents a Naïve Bayes model for the prediction of daily PV energy production. The model uses daily average temperature, total sunshine duration, as well as total global solar radiation to predict future power generation. Furthermore, [40] presents a multi-ahead prediction Multi-Layer Perceptron Neural Network, whose parameters are estimated by a probabilistic Bayesian learning technique. The Bayesian model computes the confidence intervals and estimates the error bars of the Neural Network predictions.

Ensemble methods aggregate a set of predictors (i.e. base learners) to increase the prediction accuracy of individual prediction models. As shown by [80], several top-entry PV forecasting models employ ensemble frameworks including QR Forest (QRF) with Gradient Boosting Decision Trees [223], Multiple QR [253], and Gradient Boosting Machines incorporated with NWP [210]. The ensemble models generally use bagging techniques that apply bootstrap sampling to obtain data subsets for training the base learners [253]. Also, some ensemble approaches apply the boosting algorithm which improves the performance of base models by combining them together using a particular cost function (i.e. majority vote) [223], [210]. These techniques decrease prediction variance; hence, prevents the prediction model from overfitting on the training set. In this line of research, [22] proposed a novel probabilistic prediction model based on a competitive ensemble of various base predictors for short-term forecasting of PV power. Three probabilistic methods including Bayesian model, Markov Chain model, and QR were trained as base predictors in order to obtain an ensemble of the predictive distribution with optimal sharpness and reliability metrics. The simulation results of ensemble models show improvement in these metrics compared to single-model methodologies; however, such models need more computational power and increase the time complexity of the predictor [211].

In this chapter, a new problem, probability distribution learning in graph-structured data, is solved as a recent pattern recognition challenge. First, generative modeling (learning mathematical patterns from a dataset for the aim of generating new samples under the observed data distribution) is introduced as an optimization problem where the probability of observed data in a given dataset is maximized. Then, our novel graph learning model, Convolutional Graph Auto-encoder (CGAE), is presented that is mathematically proved to learn continuous probability density functions from the nodes in an arbitrary graph. Our CGAE is defined based on the first-order approximation of graph convolutions (for learning a compact representation from an input graph) and standard function approximation (more specifically, deep neural architectures with high generalization capacity). The proposed deep learning model is able to generate new samples corresponding to each node, after observing historical graph-structured data, while learning the nodal distributions.

In this study, the problem of spatiotemporal probabilistic solar radiation forecasting is presented as a graph distribution learning problem solved by the CGAE. First, a set of solar measurement sites in a wide area is modeled as an undirected graph, where each node represents a site and each edge reflects the correlation between historical solar data of its corresponding nodes/sites. CGAE is applied to the graph in order to learn the distributions corresponding to the solar data at each site/node. Our CGAE is mathematically guaranteed to efficiently generate samples corresponding to the future solar irradiance values. The samples generated by this model result in a probabilistic solar radiation forecast for the future time step.

The key contributions of this work are: 1) Our CGAE is the first model devised in the area of machine learning, for the problem of nodal distribution learning in graph-structured data. The presented work is a universal model/algorithm that can be applied to any arbitrary graph for the probability approximation problems. 2) This is the first study of generative modeling for the prediction of renewable resources. Although generative adversarial networks have been applied in [37] for the problem of scenario generation of renewable energy production, this category of machine learning models has not been studied for the prediction tasks as these models do not estimate the probability densities of future observations given the historical measurements. The previous prediction works including all ANNs [42], [217], [133], regression [211], and kernel methods such as SVMs and SVRs [132], as well as all KNN-based methodologies [255], follow discriminative modeling [116], and no generative modeling was introduced in the literature of solar forecasting. Also, in similar areas such as probabilistic load forecasting, most approaches including ANNs [54] and Quantile Regression models [249] are discriminative rather than generative. As shown by the mathematical proof, our generative model leads to accurately understanding the underlying distribution of solar data, while discriminative modeling cannot provide such capability. 3) A spatiotemporal probabilistic forecasting framework is presented that makes use of the knowledge obtained from neighboring solar sites to enhance the prediction reliability and sharpness. 4) In contrast to previous ANN-based approaches [42], [217], [133] that merely apply shallow architectures, i.e. models with a small number of hidden layers, here, our model is able to have as many latent layers as it needs in order to provide the optimal generalization capability to increase the validation accuracy. As a result, the generalization capability and the learning capacity of our proposed deep network are much higher than previous works. Increasing the number of layers in previous models, even with the existence of a regularization error term, is infeasible as it would lead to the vanishing gradient problem. However, here, we solve the issue of having low gradient magnitude that arises in ANN architectures. 5) CGAE is compared with state-of-the-art temporal approaches including Quantile Regression [131], Kernel Density Estimation [255], and Extreme Learning Machine [217], [133] in terms of reliability, sharpness, and Continuous Ranked Probability Score (CRPS) using the National Solar Radiation Database (NSRDB) [188]. Moreover, CGAE is compared with recently proposed state-of-the-art spatiotemporal models including Space-time Copula (ST-Copula) [209], spatiotemporal QR-Lasso (ST-QR-Lasso) [5], Compressive spatiotemporal Forecasting (CSTF) [208], and spatiotemporal Support Vector Regression (ST-SVR) [132], [11]. As shown by the simulation results, CGAE outperforms all temporal as well as spatiotemporal methodologies for 0.5-hr up to 6-hr ahead predictions. CGAE improves the average reliability of the best temporal benchmark, ELM, by 3.64% in hourly predictions which grows to 4.49% in 6-hr ahead forecasting. Moreover, CGAE improves the CRPS of ELM by 3.35% for hourly predictions which is further increased to 5.22% in 6-hr ahead forecasts. Among spatiotemporal approaches, CGAE outperforms all approaches by improving the best spatiotemporal benchmark, ST-SVR, by 2.46% in hourly predictions which is further increased to 4.35% for 6-hr ahead forecasts. CGAE also improves the CRPS of ST-SVR by 1.12% and 4.19% for hourly and 6-hr ahead predictions, respectively. Furthermore, the average widths, as well as the entropies of CGAE's prediction intervals show the significant improvement of prediction sharpness of the proposed method compared to the state-of-the-art benchmarks.

The chapter is organized as follows: In Section 3.2 the problem of probabilistic solar irradiance forecasting is defined. In section 3.3, first, our proposed generative modeling paradigm is defined mathematically. Then, our CGAE model is formulated and its application for solving the forecasting problem is explained. Theoretical guarantee of the proposed methodology is available in this section. Section 3.4 explains the performance metrics and shows numerical results on a large dataset. Finally, the conclusions and future works on generative modeling are presented in Section 3.5.

3.2. Problem Formulation for Probabilistic Solar Irradiance Forecasting

The solar irradiance time series measured at 75 solar sites in northern states of the US near the Lake Michigan are collected in the National Solar Radiation Database (NSRDB) [188] by the National Renewable Energy Laboratory. Fig. 3.1 depicts the latitude-longitude map of solar sites where the spatiotemporal solar radiation data is collected. The data at each site contains the GHI time series with 30-min intervals from 1998 up to 2016. Fig. 3.2 is the plot of GHI values at the solar site 14 in 2015. As shown here, GHI increases from 8:00 to 13:00, and then, decreases until it reaches zero from about 18:00 to 20:00. Generally speaking, we have larger GHI around the day 200 (mid-July), and as we go further, the GHI declines.

The spatiotemporal data is modeled as an undirected graph where each node represents a solar site and each edge reflects the correlation between the corresponding nodes/sites. Let us define a weighted graph $G = (V_G, E_G)$ where V_G is the set of nodes v_i $i = 1, 2, ..., n$ and E_G is the set of edges e_{kl} connecting v_k to v_l. The weighted adjacency matrix A is defined by the following formulation:

$$A(k,l) = \begin{cases} \mathbf{e}^{-D(k,l)} & MI\,(k,l) \geq \alpha \\ 0 & MI\,(k,l) < \alpha \end{cases} \tag{3.1}$$

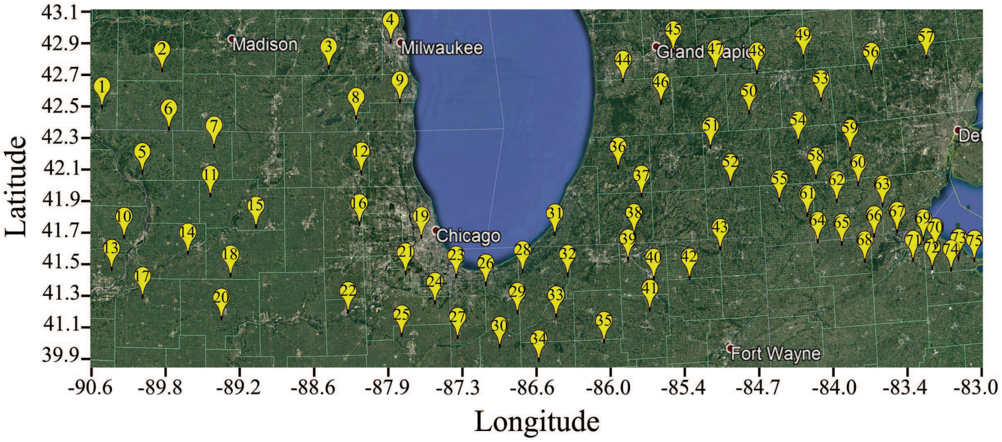


Figure 3.1. Latitude-Longitude map of 75 solar sites in the NSRDB.

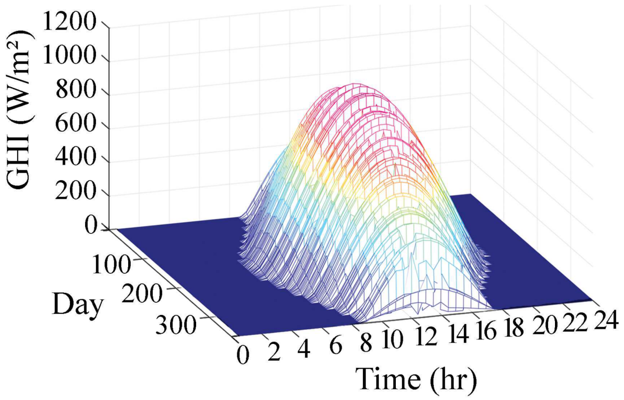


Figure 3.2. Solar Irradiance of 2015 at solar site 14.

where **e** is the Euler's number, and the edge weight between the nodes v_k and v_l is denoted by $A(k,l)$, while their distance is $D(k,l)$. Also, the normalized mutual information (MI) between the historical GHI measurements of these two nodes is denoted by $MI\,(k,l)$. The edge sparsity parameter $\alpha = 0.8$ acts as a threshold on MI values; that is, for each pair of nodes v_k and v_l, if the corresponding MI exceeds α, we consider an edge $e_{k\,l}$ associated with a weight $e^{-D(k,l)}$ while for the nodes with MI less than α, no edges are considered.

Fig. 3.3 depicts the MI values corresponding to all pairs of solar sites (i.e. nodes in V_G). Considering the latitude-longitude map in Fig. 3.1 and the MI matrix in Fig. 3.3, we can see that the MI of historical GHI for each pair of sites has high negative correlation with their distance inside the latitude-longitude space. That is, shorter distances lead to higher solar irradiance correlations, which further lead to larger edge weights in the modeled graph G.

Fig. 3.4 depicts the structure of our graph with 75 nodes and 464 weighted edges clustered into six communities using the Girvan–Newman algorithm [69]. Each community consists of a subset of nodes densely connected to each other with relatively large edge weights due to their high mutual information. The dense edges inside communities and the sparse edges between the communities reflect the strong relationship between the distance of the nodes and their MI.

At each time step t, each node v_i contains a GHI time series $T(v_i, t)$ corresponding to the historical GHI data used as the input to the forecasting model in order to predict some future GHI value ${v_i}^*(t' = t + k)$ with forecast horizon length $k > 0$. The problem is to learn a conditional probability distribution $P^*(V^*(t')|\pi)$ with future GHI tensor $V^*(t') =< {v_1}^*(t'), {v_2}^*(t'), ..., {v_n}^*(t') >$ and historical GHI tensor $\pi =< T(v_1, t), T(v_2, t), ..., T(v_n, t) >$. Considering a training set TS that contains $|TS|$ historical examples $(\pi_j, V_j^*(t'))\ \ 1 \leq j \leq\ |TS|$, we need to estimate P^* using the observed π_j and $V_j^*(t')$ in the j-th training example.

The data of 1998-2015 is considered for training our model while the 2016 dataset is used as a test set to evaluate our method. Fig. 4. shows the mutual information between a GHI value at the time $\tilde{t}$ with previous time steps $\tilde{t} - l$ with lag $1 \leq l \leq 300$ for the GHI time series of 1998-2015. As shown in this plot, the GHI values are more correlated with their most recent lags as well as the time lags near $l \in \{24, 48, 72, 96, 120, 144\}$. In this study, in order to make the information in $T(v_i, t)$ useful for the estimation of P^*, we define $T(v_i, t)$ for each node i to be the GHI values corresponding to the lags where the mutual information is equal or greater than some threshold

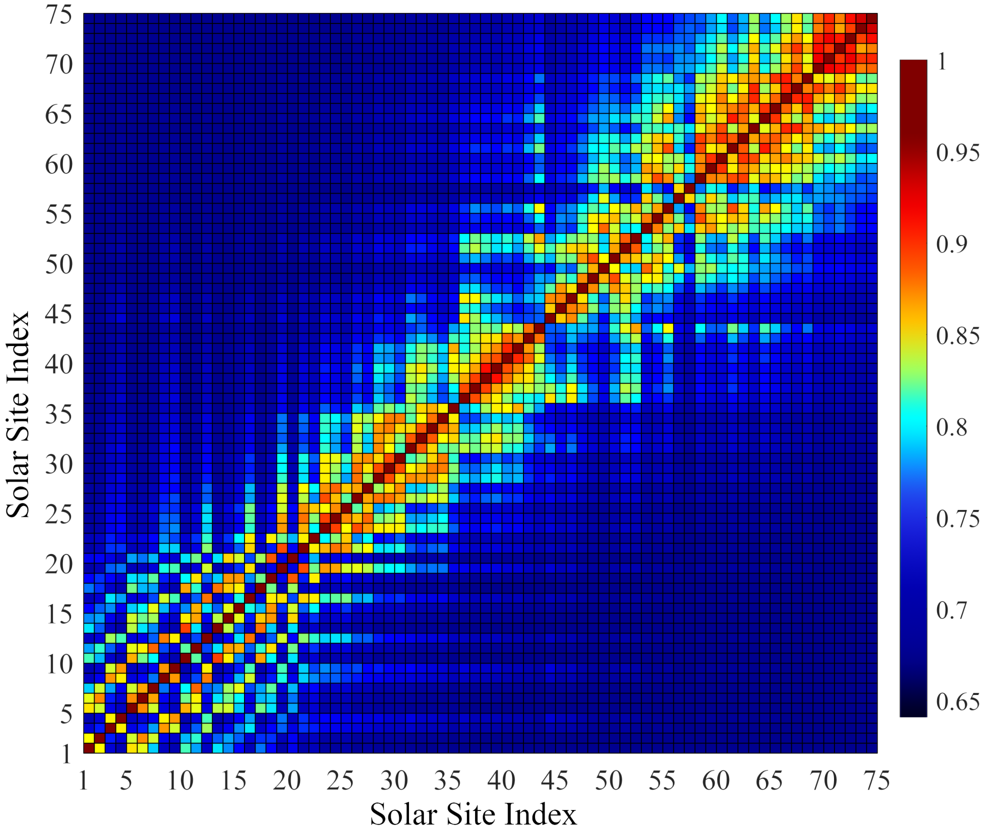


Figure 3.3. Mutual Information matrix for all pairs of nodes in V_G. The indices correspond to the indices of solar sites in 3.1.

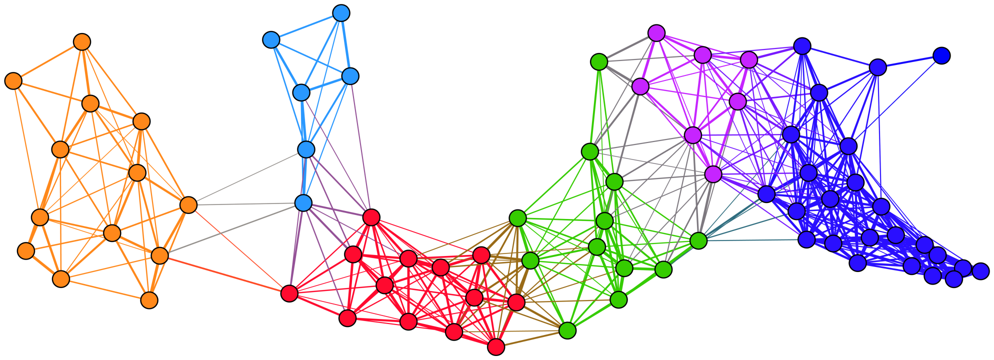


Figure 3.4. Structure of the modeled graph G with 75 nodes and 464 edges. The graph is clustered into six Girvan–Newman communities. The width of each edge reflects the magnitude of MI between the corresponding nodes.

$\tau \geq 0$. Here, τ is a hyperparameter for our model.

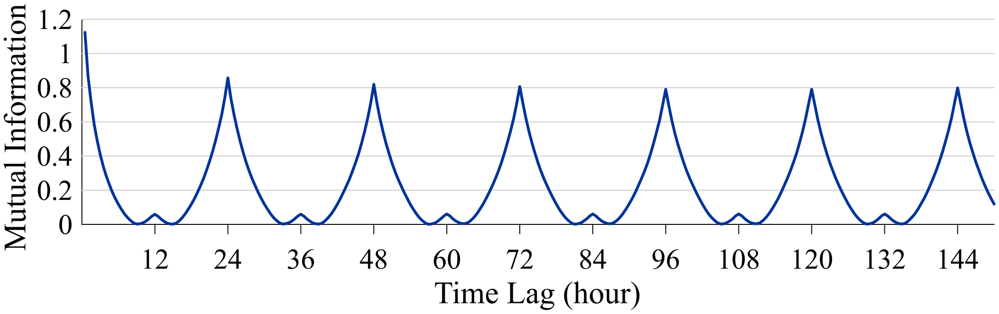


Figure 3.5. Mutual Information of future GHI with previous time lags

3.3. Proposed Generative Learning Formulation for Nodal Probability Density Estimation in Graphs

3.3.1. Generative Learning for PDF approximation

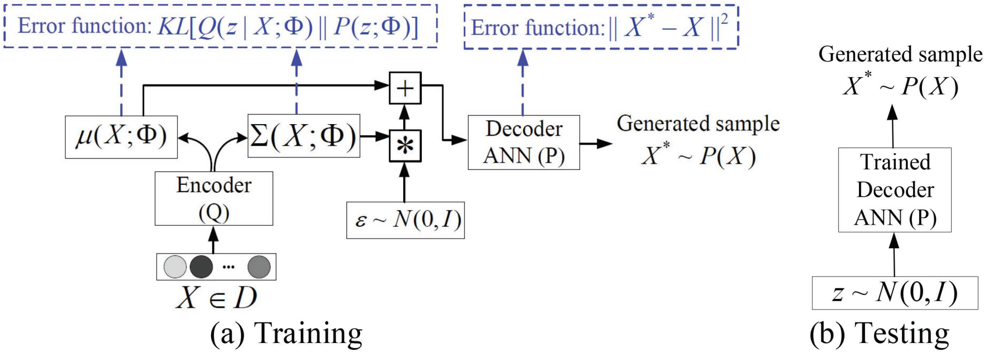
Here, our problem is to capture a probability distribution $P(X)$ over n-dimensional data points X in a potentially high dimensional vector space $\mathcal{X} \subseteq \mathbb{R}^n$. In fact, we want to be able to generate many samples X^* as close as possible to X. As the complexity of the dependencies between variables of $\mathcal{X}$ grows, the difficulty of learning the true $P(X)$ increases. Hence, we define a "latent variable"-based model in which the hidden random vector $z \in Z$ embodies the major characteristics of $P(X)$ (e.g. the PDF of the future GHI, or any desired nodal PDF in a graph-structured data). More specifically, z is sampled following some unknown distribution $P(z)$ over the high dimensional space Z. To justify that our approach is generative (i.e. the model can generate samples X^*), we ensure that there exists at least one configuration $\hat{z} \in Z$ that causes the model to generate some sample $\hat{X}$ in $\mathcal{X}$. Assuming a family of deterministic functions $f(z;\theta)$ with parameters $\theta \in \Theta$, each "latent variable-parameter" pair is mapped to a sample in $\mathcal{X}$ using $f : Z \times \Theta \to \mathcal{X}$. We find an optimal $\theta^* \in \Theta$ such that when $z \sim P(z)$, the value of $X^* = f(z; \theta = \theta^*)$ is as close as possible to some $X \in \mathcal{X}$. In other words, the probability of f creating an output X^* similar to the observed data X is maximized; hence, our optimization is written as:

$$\theta^* = \underset{\theta}{\arg\max} \left[P(X) = \int f(z;\theta)P(z)dz \right] \tag{3.2}$$

$f(z)$ is a deterministic function of a random variable z ; hence, for a fixed θ, $f(z;\theta)$ is a random variable in the space $\mathcal{X}$. Therefore, $P(X)$ in (3.2) can be written as:

$$P(X) = \int P(X|z;\theta)P(z)dz \tag{3.3}$$

As shown in (3.2), generating X depends on the latent vector z. Using the Maximum Likelihood framework, if the model converges to the solution θ^*, our generative model is likely to produce X^*. Here, $f(z;\theta)$ is defined as a Gaussian distribution $P(X|z;\theta) = N(X|f(z;\theta), \sigma^2 * I\,)$ with mean f and a diagonal covariance matrix with entries computed using the hyperparameter σ as the standard deviation.



(a) Training

(b) Testing

Figure 3.6. Structure of CGAE. (a) shows the training process where the model generates $X^* \simeq X$. (b) shows the testing process where the trained decoder generates as many samples $X^* \sim P(X)$ as required simply by feeding a random $z \sim N(0, I)$ to the decoder ANN. The decoder captures PDF $P(X)$.

In order to solve the optimization (3.2)-(3.3), z should be mathematically defined. Moreover, an estimation for the integral in (3.2) should be provided. Our main goal is to learn variable z automatically; that is, we opt to avoid describing the dependencies between the dimensions of Z, as no prior knowledge is available/required to solve the problem. Thus, the latent vector is set to

$z \sim N(0, I\ )$ considering Theorem (3.1):

Theorem (3.1)**:** *In any space Λ, any complicated probability density function over samples can be modeled using a set of $\ dim(\Lambda)\ $ random variables with normal distribution, mapped through a high capacity function.*

As a consequence, an approximator can be learned to map z to some required (desired) hidden variable ξ further mapped to $X \in \mathcal{X}$, to maximize the likelihood of samples X in the dataset D. Here, our f is modeled by an ANN as a standard function approximator capable of learning highly nonlinear target functions using multiple hidden layers. The first layers of these architectures provides a non-linear mapping from $z \in Z$ (with a predefined simple distribution as discussed in this section) to ξ (with an unknown complicated distribution). ξ is further mapped to a sample $X \in \mathcal{X}$ available in D. Notice that if the model has sufficient capacity (ample number of hidden layers, as in the case of deep neural networks), the neural network is able to solve the maximization in (3.1) to obtain θ^*. Let us rewrite our optimization in (3.2) using $z \sim N(0, I\ )$ from Theorem (3.1):

$$\theta^* = \underset{\theta}{\arg\max} \int N(X|f(z;\theta),\, \sigma^2 * I\ )N(z|0, I\ )dz \tag{3.4}$$

To solve (3.4), a distribution function $Q(z|X)$ is defined to decide the importance of an arbitrary configuration $\hat{z} \in Z$ in the generation of a sample X. As a consequence, the expected value of $P(X|z)$ with respect to z, $\mathrm{E}_{z\sim Q}\,[P(X|z)]$, can be computed using the Kullback–Leibler (KL) divergence:

$$KL[Q(z)||p(z|X)] = \mathrm{E}_{z\sim Q}\,[\log\ Q(z)\ -\ \log\ P(z|X)] \tag{3.5}$$

applying the Bayesian rule for $P(z|X)$, (3.5) can be written as:

$$\begin{aligned} KL[Q(z)||p(z|X)] &= \mathrm{E}_{z\sim Q}\left[\log\ Q(z) - \log(\tfrac{P(X|z)P(z)}{P(X)})\right] \\ &= \mathrm{E}_{z\sim Q}\,[\log\ Q(z) - \log\ P(X|z)\ - \log\ P(z)\ + \log P(X)] \end{aligned} \tag{3.6}$$

This equality is further written as:

$$
\begin{aligned}
&\log P(X) - KL[Q(z|X)||P(z|X)] \\
&= \mathrm{E}_{z\sim Q}\left[\log P(X|z) - KL[Q(z|X)||P(z)]\right]
\end{aligned}
\tag{3.7}
$$

In order to generate X (that is, create samples $X^* \approx X$), our objective is to maximize $\log P(X)$ while minimizing the KL divergence in the left-hand side of (3.7); hence, we minimize $\mathrm{E}_{z\sim Q}\left[\log P(X|z) - KL[Q(z|X)||P(z)]\right]$ using SGD. Notice that, in the formulation of (3.7), Q can be viewed as an ANN encoding X into z, while P is an ANN decoding z to obtain X. To solve the optimization, Q is defined as:

$$
Q(z|X) = N(z|\mu(X;\Phi), \Sigma(X;\Phi)) \tag{3.8}
$$

with deterministic functions μ and Σ defined by an ANN with free parameters set Φ trained by SGD. As Q and P are both dimensional multivariate Gaussian distributions, the term $KL[Q(z|X)||P(z)]$ in (3.7) is computed by:

$$
\begin{aligned}
&KL\left[Q(z|X)||P(z)\right] \\
&= KL\left[N(z|\mu(X;\Phi), \Sigma(X;\Phi))||N(0,I)\right] \\
&= \tfrac{1}{2}\left[\log \tfrac{\det(I)}{\det(\Sigma)} - d + tr(\Sigma) + (0-\mu)^T(0-\mu)\right] \\
&= \tfrac{1}{2}\left[-\log(\det(\Sigma)) - d + tr(\Sigma) + \mu^T\mu\right]
\end{aligned}
\tag{3.9}
$$

Therefore, in order to optimize (3.7), the following optimization problem is solved:

$$
\theta^* = {}^{\arg\max}_{\theta}\mathrm{E}_{X\sim D}\begin{bmatrix} \mathrm{E}_{z\sim Q}[\log P(X|z;\Phi)] \\ -KL[Q(z|X;\Phi)||P(z;\Phi)] \end{bmatrix} \tag{3.10}
$$

Applying the reparametrization technique, (3.10) can be written as:

$$\theta^* = {}^{\arg\max}_{\theta} E_{X\sim D}\left[\begin{array}{l} E_{\varepsilon\sim N(0,I)}\left[\begin{array}{l}\log P\Big(X|z=\mu(X)\\ +\Sigma^{1/2}(X)*\varepsilon\,;\Phi\Big)\end{array}\right]\\ -KL[Q(z|X;\Phi)||P(z;\Phi)]\end{array}\right] \tag{3.11}$$

Fig. 3.6(a) shows the training structure of our generative model based on (3.8) and (3.11) to generate $X^* \approx X$. The encoder ANN, Q, takes X observed in dataset D and outputs μ and Σ (see (8)). The error of the encoder ANN is $KL[Q(z|X)||P(z)]$ computed in (3.9). The gradient of this error function is used by Stochastic Gradient Descent (SGD) method to train this ANN. After computing μ and Σ using Q, our latent variable $z = \mu(X;\Phi) + \Sigma^{1/2}(X;\Phi)\ *\varepsilon$ is obtained using (3.11). Then, z is fed to the decoder ANN, P, to obtain our generated sample $X^* \approx X$. The error function of this ANN is computed by $||X - X^*||^2$ to reflect the distance between the generated sample X^* and its true (observed) value X. When Q and P are trained by SGD, in order to generate a new sample $X^* \approx X$, one can simply feed some $z \sim N(0, I)$ to P and obtain X^* as shown in Fig. 5(b).

3.3.2. Convolutional Graph Auto-encoder

In Section III-A, our objective was to learn $P(X)$ in some high dimensional space $\mathcal{X}$ by generating $X^* \approx X$. Here, we aim to learn $P^*(V^*|\pi)$, i.e. PDF of V^* in G given π. We present our CGAE shown in Fig. 3.7 as the first generative model that captures nodal distribution $P^*(V^*(t')|\pi)$ in a graph G. Given historical GHI π, our objective is to generate ρ samples $\hat{V} \approx V^*$ to estimate $P^*(V^*|\pi)$.

Let us mathematically formalize how CGAE generates $\hat{V}$ as an estimation for V^* :

$$\hat{V} = \mu(\pi, z) + \varepsilon \textit{ s.t. } z \sim N(0,1)\,,\ \varepsilon \sim N(0,1) \tag{3.12}$$

both z and ε are white Gaussian noises. μ is implemented by an ANN as in Section III-A. Assuming $z \sim Q$ using PDF $Q(z)$, Bayes rule [56] is applied to compute $\mathrm{E}_{z \sim Q}[\log P(V^*(t')|z, \pi)]$:

$$\begin{aligned}\mathrm{E}_{z \sim Q}[\log P(V^*(t')|z, \pi)] = \mathrm{E}_{z \sim Q}[\log P(z|V^*(t'), \pi) - \\ \log P(z|\pi) + \log P(V^*(t')|\pi)]\end{aligned} \tag{3.13}$$

(3.13) is rewritten as:

$$\begin{aligned}\log P(V^*(t')|\pi) - \mathrm{E}_{z \sim Q}[\log Q(z) - \log P(z|\pi, V^*(t')) = \\ \mathrm{E}_{z \sim Q}[\log P(V^*(t')|z, \pi) + \log P(z|\pi) - \log Q(z)]\end{aligned} \tag{3.14}$$

Now, following (3.8), we have $Q = N(\mu'(\pi, V^*(t'))\ ,\ \sigma'(\pi, V^*(t')))$ where μ' and σ' are ANNs trained alongside μ. Let us denote Q by $Q(z|\pi, V^*)$, (3.14) is written as:

$$\begin{aligned}\log P(V^*|\pi) - KL[Q(z|\pi, V^*)||P(z|\pi, V^*)] = \\ \mathrm{E}_{z \sim Q}[\log P(V^*|z, \pi)] - KL[Q(z|\pi, V^*)||P(z|\pi)]\end{aligned} \tag{3.15}$$

Considering (3.15), our objective is to increase $E_1 = \log P(V^*|z, \pi)$ and $E_2 = -KL[Q(z|\pi, V^*)||P(z|\pi)]$. CGAE is trained by SGD to maximize $E_T = E_1 + E_2$. This leads to maximizing the likelihood of V^* while training Q to accurately estimate $P(z|\pi, V^*)$. Note that, similar to our optimization in Section III-A, we have $P(z|\pi) = N(0, 1)$. Our latent vector is $z = \mu'(\pi, V^*(t'))\ + \alpha \circ\ \sigma'(\pi, V^*(t'))$ where $\alpha \sim \mathrm{N}(0, 1)$ and $\circ$ is the element-wise product operation. E_T is differentiable with respect to the whole parameters of CGAE (including the parameters in ANNs corresponding to μ, μ' and σ'); hence, the whole CGAE model can be easily tuned by SGD to maximize E_T. In Section III-C, the neural architecture corresponding to our CGAE is defined based on ANNs.

3.3.3. CGAE Architecture

CGAE consists of three ANNs; 1- Graph Feature Extraction ANN, which gives us a compact representation of π stored in G, denoted by $R(G)$, 2- Encoder ANN, Q, that implements μ' and σ' to capture $Q(z|\pi, V^*)$, and 3- Decoder ANN, P, that implements $\mu(\pi, z)$ in (3.12), to produce samples $\hat{V}$ drawn from the true future GHI distribution $P^*(V^*(t')|\pi)$.

3.3.4. Graph Feature Extraction ANN (Computing R(G))

At each training step t, the spectral graph convolutions of G, which stores $\pi =< T(v_1, t), T(v_2, t), ..., T(v_n, t) >$ inside its nodes, is computed by $\psi_\theta * \pi = U\psi_\theta U^T \pi$. Here, U is the eigenvector matrix of the normalized Laplacian $L = U\Omega U^T$ and $\theta \in \mathrm{R}^n$ is the parameter vector for the convolutional filter $\psi_\theta = diag(\theta)$ in the Fourier domain. Notice that the Fourier transformation of π is computed by $U^T\pi$. ψ_θ is defined as a function of L 's eigenvalues; hence, our filter is denoted by $\psi_\theta(\Omega)$. Estimating $\psi_\theta(\Omega)$ by Chebyshev Polynomials [114, 122] P_j, we have $\psi_\omega \approx \sum_{j=0}^{J} \omega_j P_j(\frac{2}{\gamma_{\max}}\Omega - I)$ where $\gamma_{\max}$ is the maximum eigenvalue of L, and ω_j is the j -th Chebyshev coefficient. Therefore, the spectral graph convolution function on G is:

$$\psi_\omega * \pi \approx \sum_{j=0}^{J} \omega_j P_j(\frac{2}{\gamma_{\max}}\Omega - I)\pi \tag{3.16}$$

The convolution in (3.16) is further simplified by $\delta = \omega_0 = -\omega_1$ which decreases parameters' size while $\gamma_{\max} = 2$ for $J = 1$; As a result, (3.16) can be computed by:

$$\psi_\omega * \pi \approx \omega_0 P_0(L - I)\pi + \omega_1 P_1(L - I)\pi = \delta(I + D^{-\frac{1}{2}} A D^{-\frac{1}{2}})\pi \tag{3.17}$$

Based on the convolution (3.17), a graph feature extraction neural network (GFENN) with L_G hidden layers is defined to extract spatiotemporal features from GHI observations at all nodes/sites of G. Here, the output of each layer $1 \leq k \leq L_G$ is:

$$O^k = \mathrm{ReLU}(MO^{k-1}\mathrm{W}^k) s.t. M = \tilde{D}^{-\frac{1}{2}}(A + I)\tilde{D}^{-\frac{1}{2}} \tag{3.18}$$

where $\tilde{D}_{ii} = \sum_j (A+I)_{ij}$. The input of GFENN is $O^0 = \pi$ while the output is G 's spatiotemporal representation $R(G) = O^{L_G}$.

3.3.5. The encoder (Q) and Decoder (P)

Since GFENN captures spatiotemporal features of π, and stores them in $R(G)$, one can view CGAE as a model estimating $P^*(V^*|R(G))$ instead of $P^*(V^*|\pi)$. In Section III-A, (3.8) showed that Q can be viewed as an ANN encoding input tensor X into the latent vector z while P is a decoding ANN that maps z to X. As depicted in Fig. 3.7, Here, the input to the encoder Q is $X = R(G)$. Our encoder Q is defined by a deep ANN with L_Q hidden layers and ReLU activations for each hidden layer, trained to encode V^* into a latent vector $z \in Z$, such that the resulting z can be decoded back to V^*. As discussed in (3.15) and also shown in Fig. 3.7, the error function for the encoder Q is defined by:

$$\begin{aligned} Err_Q &= KL[Q(z|\pi, V^*)||N(0,1)] \\ &= KL[Q(z|R(G), V^*)||N(0,1)] \end{aligned} \tag{3.19}$$

Similar to Q, our decoder, P, is implemented by a deep ANN with L_P hidden layers using ReLU activations to take the latent vector z learned by Q, as well as the graph representation $R(G)$, and decode them to generate an approximation of V^*, denoted by $\hat{V}$. To make the generated sample $\hat{V}(t')$, as close as possible to the real future value $V^*(t')$ we minimize the following reconstruction error for P :

$$Err_P = ||V^*(t') - \hat{V}(t')||^2 \tag{3.20}$$

Therefore, the total error optimized by the stochastic gradient descent method is $E = Err_Q + Err_P$.

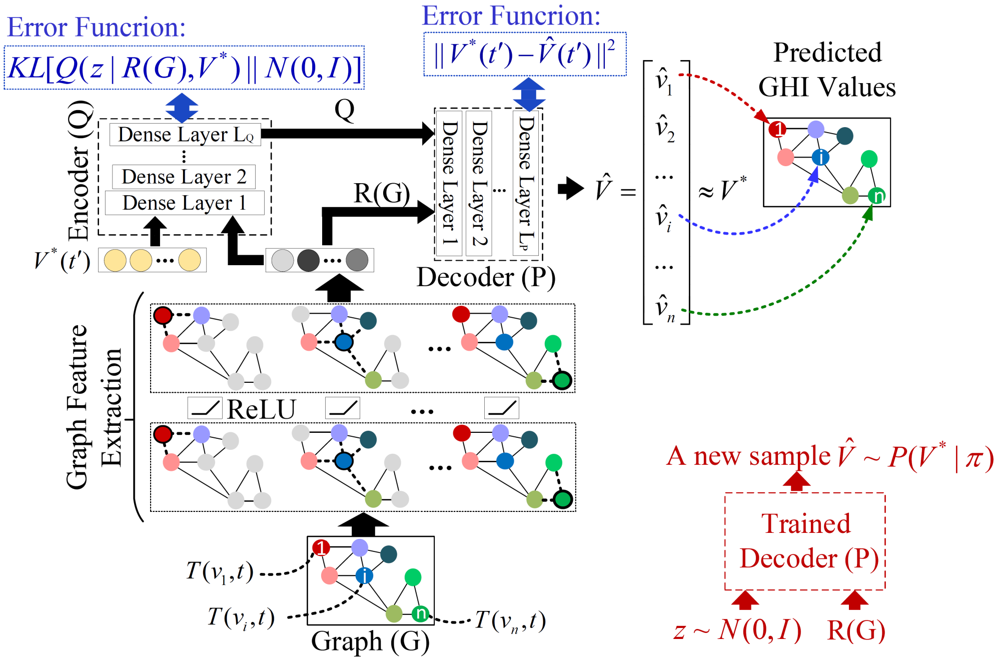


Figure 3.7. Convolutional Graph Auto-encoder.

3.3.6. Estimation of $P(V^*|\pi)$

As shown in Fig. 6(b), during test time, $R(G)$ and $z \sim N(0, I)$ are fed to the decoder ANN and the estimation $\hat{V}(t')$ is obtained. No encoding is needed; hence, generating estimations $\hat{V}(t') \approx V^*(t')$ is dramatically fast. All we need to do to generate a new sample $\hat{V}(t')$, is to sample a new $z \sim N(0, I)$ and run feed-forward algorithm on the GFENN (to obtain $R(G)$) and the decoder ANN (to obtain the desired result, i.e. $\hat{V}(t')$). Following this approach, we generate ρ number of samples $\hat{V} \sim P(V^*|\pi)$ to estimates $P(V^*|\pi)$ using the decoder. As a result, our decoder P generates the PDF of future GHI mapping $N(0, I)$ to $P(V^*|\pi)$.

3.4. Numerical Results

CGAE is compared with recent temporal as well as spatiotemporal benchmarks utilized for short-term irradiance/PV probabilistic forecasting. The temporal models include Quantile Regression (QR) [211], Kernel Density Estimation (KDE) [255], Extreme Learning Machines (ELM) [217], and Probabilistic Persistence (PP) [7], while the spatiotemporal benchmarks include the

Space-time Copula [209], spatiotemporal QR-Lasso [5], Compressive Spatiotemporal Forecasting [208], and Spatiotemporal Support Vector Regression [132], [11]. The advantages of spatiotemporal feature learning for the underlying problem is shown. Since no generative model was presented in the literature, the experiments motivate further research on generative modeling for renewable resources prediction.

3.4.1. Experimental Settings

As explained in Section II, the NSRD dataset is applied to train/test our model. The 1998-2015 data is used to train CGAE while the 2016 data is applied to evaluate the prediction performance. In this study, CGAE is trained/tested to forecast GHI time series from 30 min (horizon length $k = 1$) up to 6 hours ahead ($k = 12$). Batch Gradient Descent with learning rate $\eta = 5{*}10^{-4}$ is employed to train our CGAE (including GFENN, encoder ANN, and decoder ANN) by minimizing the error $Err_Q + Err_P$ using batch size k equal to 400. In this study, the number of generated samples is $\rho = 10^4$, and the number of GFENN layers is set to $L_G = 2$ while $L_P = 4$ and $L_Q = 3$. The feature selection hyperparameter is $\tau = 0.45$.

We employed the Information Theoretical Estimators (ITE) library [203] to compute the mutual information matrix corresponding to the historical GHI time series in Section II. The ITE is used as a free and open source toolbox in Matlab 2018. The graph modeling process of Section II is implemented in Gephi 0.9.2 [15] which is an open-source software for network visualization and analysis. Moreover, our proposed deep neural network, CGAE, is implemented in Python 3.6 with Keras 2.2.4 library [38] and GPU-based Tensorflow 1.7.0 [4] backend. The model is implemented on a computer system with Intel Core-i7 4.1GHz CPU and NVIDIA GeForce GTX 1080-Ti GPU. Our GPU supports CUDA 9.0 which is a parallel computing platform that helps Tensorflow to speed up all the computations in Keras.

3.4.2. Performance Comparison (Quantitative Results)

The prediction quantiles of our model are compared with both temporal and spatiotemporal methodologies in terms of reliability, sharpness and Continuous Ranked Probability Score (CRPS):

3.4.2.1. Reliability

This criterion shows how closely the prediction probabilities correspond to the observed (real) frequencies of the GHI data. Here, the bias $R^{1-2\alpha}$ is computed by:

$$R^{1-2\alpha} = \left(\frac{N^{1-2\alpha}}{N} - (1 - 2\alpha) \right) \times 100\% \tag{3.21}$$

where N is the number of test examples, $N^{1-2\alpha}$ is the number of observations covered by the nominal coverage rate $(1 - 2\alpha) \times 100\%$. The closer the nominal coverage of prediction intervals is to the observed (actual) coverage rate, the higher the reliability is; hence, small $R^{1-2\alpha}$ shows better accuracy. In fact, $R^{1-2\alpha} = 0$ corresponds to the perfect (ideal) reliability.

Fig. 3.8 depicts the reliability measurements averaged over all GHI nodes/sites with various nominal coverage rates ranging from 10% to 90%. As shown in this figure, the spatiotemporal prediction models including CGAE, ST-Copula, ST-QR-Lasso, CSTF, and ST-SVR, lead to more reliable probabilistic forecasts compared to the temporal models such as ELM, KDE, QR, and PP. For instance, the ST-QR-Lasso model which is a spatiotemporal version of QR, leads to an average deviation of 5.46% while the QR obtains 9.13% deviation compared to the ideal prediction model with zero deviation. Among the temporal models, PP has the worst reliability which results in the largest average deviation equal to 10.62%. ELM leads to the highest reliability among temporal models with 6.71% absolute deviation. This model yields 36.81%, 26.49%, and 22.59% more reliable (less deviated) predictions compared to PP, QR, and KDE, respectively. The major reason for this observation is the better generalization of neural network-based approaches compared to the traditional statistical approaches. In contrast to other temporal benchmarks, ELM has a large nonlinear parameter space which helps this model to improve generalization and obtain more reliable outcomes. Our deep learning-based generative model, CGAE, outperforms all temporal benchmarks, with 86.35%, 84.12%, 83.28%, and 78.40% better reliability compared to PP, QR, KDE, and ELM. The smaller deviation of CGAE compared to ELM is mainly due to CGAE's graph-based spatial feature extraction as well as its larger hypothesis space caused by the higher number of nonlinear computational layers.

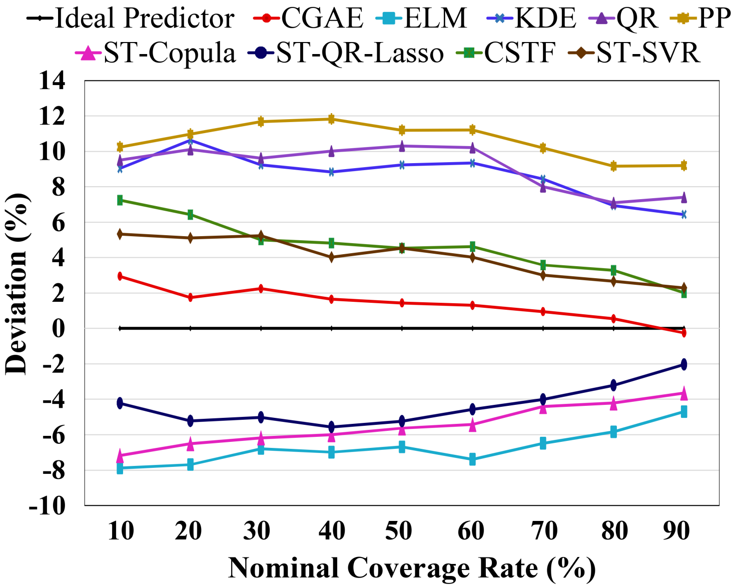


Figure 3.8. Reliability measurements averaged over all GHI nodes/sites.

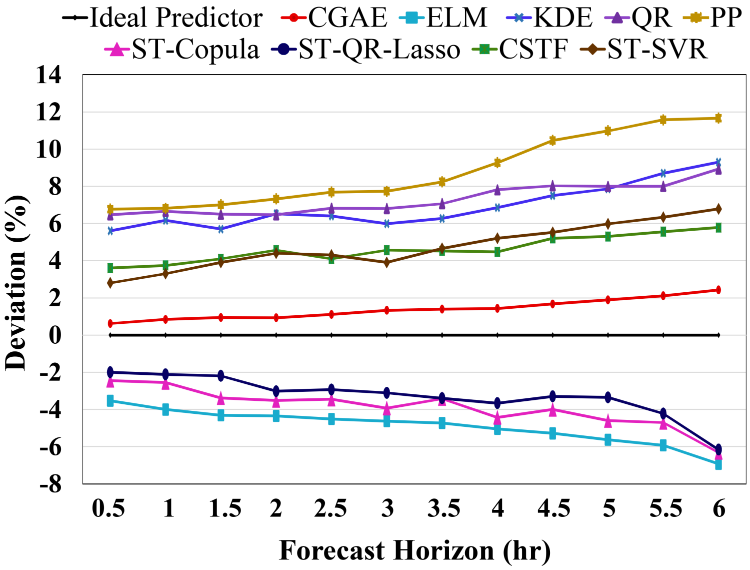


Figure 3.9. Average reliability with different look-ahead times.

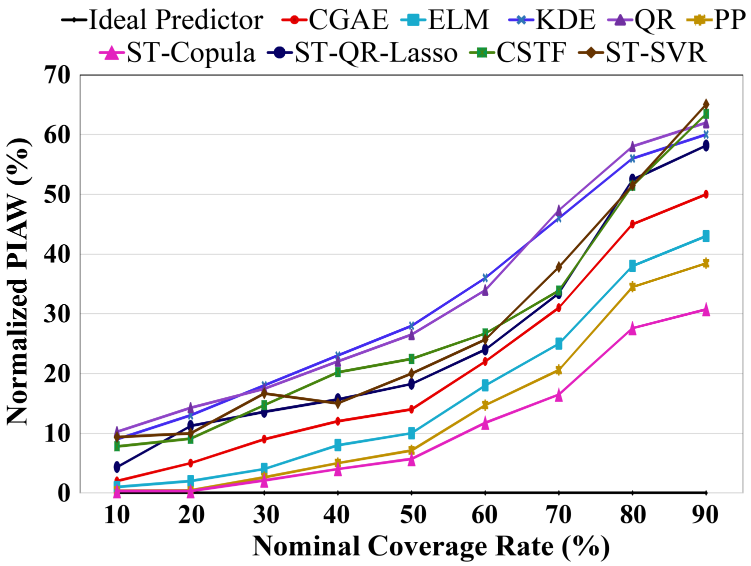


Figure 3.10. Sharpness evaluation using normalized PIAW.

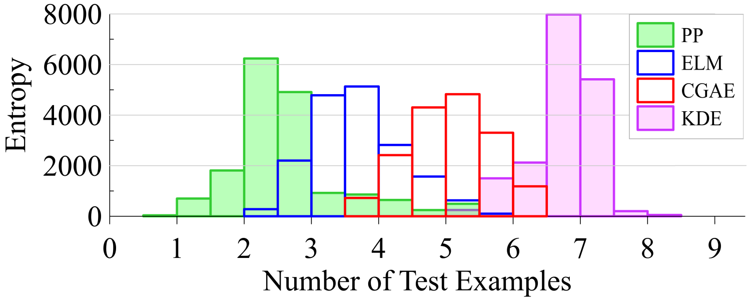


Figure 3.11. Entropy diagram of CGAE with various temporal benchmarks for the 6-hr ahead forecasts.

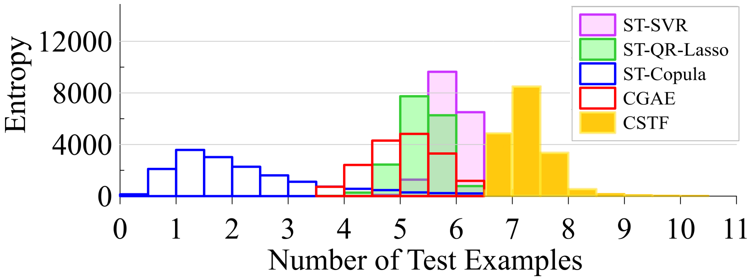


Figure 3.12. Entropy diagram of CGAE with various spatiotemporal benchmarks for the 6-hr ahead forecasts.

Among the spatiotemporal prediction benchmarks, CGAE and ST-SVR have the least deviated predictions with 1.45% and 4.02% average absolute deviations, respectively. The reliable performance of ST-SVR is due to its ability to handle complex high-dimensional feature spaces using the kernel trick. The smaller deviation of CGAE in comparison with other spatiotemporal benchmarks shows the effectiveness of our GFENN in providing powerful spatial information from the underlying solar sites.

Fig. 3.9 shows the average reliability with different look-ahead times for various temporal and spatiotemporal benchmarks. As shown in this plot, the slope of the deviation curve for all benchmarks start to increase significantly from the 3.5-hr horizon, while CGAE has a much smaller slope. As the time horizon expands, the improvement of CGAE becomes more significant. PP has the worst performance, especially in longer horizons, compared to other methodologies. This is due to its low generalization capacity resulted from its smoothness assumption of the target function, which undermines its efficiency in practice. The spatiotemporal approaches have less than 6.31% deviation for all time horizons while even the most reliable temporal model, ELM, exceeds this limit for 5.5-hour and 6-hour ahead predictions. CGAE yields 1.10% and 4.49% better reliability in 3-hr and 6-hr forecasts compared to ELM, respectively. This shows the superiority of generative modeling over discriminative modeling introduced in previous ANN methods in the literature. The relatively small deviation of spatiotemporal models is resulted by their good unbiased prediction, while temporal models are more biased, which degrades their efficiency in practical applications.

Among the spatiotemporal approaches, the CGAE, CSTF, and ST-SVR have smaller deviation slope with respect to the time horizon. While ST-QR-Lasso and ST-Copula have a significant growth in their deviation slope after the 5-hr time horizon, the CGAE, CSTF, and ST-SVR show a smooth deviation curve with a relatively small gradient. As shown in Fig. 3.9, CGAE shows more reliable predictions in comparison with all spatiotemporal benchmarks. As the time horizon expands, the superiority of CGAE becomes more noticeable. For the 6-hr ahead prediction, CGAE obtains 4.35%, 3.88%, 3.72%, and 3.35% better reliability in terms of the deviation from the ideal prediction compared to ST-SVR, ST-Copula, ST-QR-Lasso, and CSTF, respectively.

3.4.2.2. Sharpness:

Sharpness is a complementary metric to the reliability, which evaluates the concentration of the prediction distribution. The criterion shows how informative a forecast is by narrowing down the predicted GHI values. Sharpness should be analyzed with respect to reliability, as high sharpness does not necessarily show better prediction when the model has low reliability (high deviation in Fig. 3.8 and Fig. 3.9). Sharpness is investigated using two performance metrics:

3.4.3. Prediction Interval Average Width (PIAW)

This metric, $PIAW^{\alpha}$, evaluates sharpness for the nominal coverage rate $(1-2\alpha)\times 100\%$ by:

$$PIAW_{\alpha} = \frac{1}{N}\sum_{n=1}^{N}|q^{\alpha}(n) - q^{1-\alpha}(n)| \qquad (3.22)$$

where $q^{\alpha}(n)$ and $q^{1-\alpha}(n)$ represent the α and $1-\alpha$ prediction quantiles for the n -th test sample. Fig. 3.10 shows the average sharpness of 10%-90% nominal coverage rates normalized by maximum observed GHI. As shown in this diagram, among temporal models, PP has the sharpest intervals in all nominal coverage rates; however, as shown by Fig. 3.8 and Fig. 3.9, it has poor reliability compared to other benchmarks especially when the horizon is expanded. Moreover, ELM provides overly narrow quantiles leading to higher sharpness compared to CGAE. However, such high sharpness does not contribute to forecast accuracy/reliability. Large amount of sharpness might work in the case of clear sky when no significant uncertainty is present and GHI is predictable with high accuracy; however, in other cases (e.g. when GHI is varying during a rainy day), it would lead to poor performance as the model would neglect the risk of uncertainties in GHI. CGAE provides medium sharpness which is not too high to lead to erroneously narrow quantiles (as in the case of PP and ELM), and not too low to lose information about future GHI (as in the case of KDE and QR).

Generally speaking, the spatio-temproal models obtain moderate sharpness values that are neither as high as KDE nor as low as PP. Among this category of models, ST-Copula is an exception which provides prediction intervals even sharper than the PP. The sharpness metric shows that ST-

Copula is likely to provide biased predictions that are over-confident. In practice, such confidence can lead to poor performance since the reliability of ST-Copula is lower than the other spatiotemporal benchmarks. As shown by Fig. 9, the ST-QR-Lasso, CSTF, and ST-SVR provide similar sharpness for 60% and 70% nominal coverage rates; however, for other coverage values, the prediction intervals of ST-QR-Lasso, CSTF, and ST-SVR become too sharp while CGAE maintains its moderate sharpness.

3.4.4. PDF Entropy

The sharpness of a forecast can be estimated using the entropy of the prediction PDF. Sharper forecasts lead to smaller PDF entropies. Fig. 3.11 (a) shows the histogram of the entropies of all temporal benchmarks for the 6-hr ahead prediction task. As shown in this plot, the majority of forecasting PDFs for PP and ELM correspond to low values. The mean entropy of PP and ELM are 2.77 and 3.69, respectively. The low entropy of PP is due to the consecutive clear days in the testing set where the variance of the prediction PDF is small. Such small entropies/variances result in overconfident predictions caused by the lack of knowledge about future GHI uncertainties. The overly narrow prediction quantiles in ELM lead to low PDF entropies which degrade accuracy since the uncertainties in the future GHI are disregarded by predictions less reliable than CGAE (see Fig. 3.8 and Fig. 3.9). CGAE has moderate sharpness and medium entropy values with mean 5.15. KDE has high entropies with mean 6.77 and a small variance of 0.22 that result in high uncertainty boundaries for the future GHI and less informative forecasts compared to CGAE and ELM. In contrast to ELM and KDE, our CGAE model has entropies that are not too low (as in the case of ELM) to disregard GHI uncertainties and not too high (as in the case of KDE) to provide under-confident predictions.

Fig. 3.12 depicts the histogram of the entropies of all temporal benchmarks for the 6-hr ahead prediction task. As shown in this diagram, ST-Capula obtains relatively small entropy which is reflected by the over-confidence and large bias in the prediction PDFs of this model. On the other hand, the CSTF leads to under-confident results with high entropies. The mean entropy of CSTF is 7.26 which is 19.01%, 23.83%, and 29.06% higher than the ST-SVR, ST-QR-Lasso, and

CGAE, respectively. This is mainly due to having high variance (high uncertainty) in consecutive sunny days when predicting by CSTF. Such variance is degraded by ST-SVR, ST-QR-Lasso, and CGAE as these models provide a better bias (larger bias) when they encounter multiple consecutive sunny days in the test set. The moderate entropy obtained by CGAE shows that this model is not too biased (as in the case of ST-Copula) to neglect GHI uncertainties in the dataset, and not too uncertain (as in the case of CSTF) to provide uninformative predictions with high unreliability.

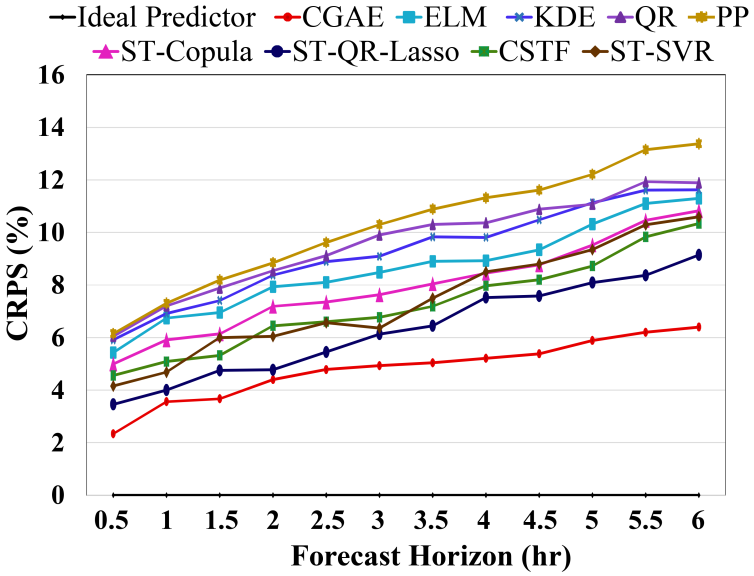


Figure 3.13. CRPS results of 30-min up to 6-hr ahead predictions.

3.4.4.1. Continuous Ranked Probability Score

CPRS is a metric evaluating the entire prediction distribution reflecting the deviations between the CDF of the predicted and observed data. One can view CRPS as a metric combining reliability and sharpness to provide a comprehensive performance evaluation. CRPS is computed by:

$$\begin{aligned} CRPS(F,v) &= \int_{-\infty}^{\infty} (F(x) - U(x-v))^2 dx \\ s.t.\ \ U(x) &= \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases} \end{aligned} \tag{3.23}$$

with the prediction CDF F and the Heaviside function U. The average CRPS of all benchmarks for 30-min up to 6-hr ahead GHI forecast is depicted in Fig. 3.13. The smaller CRPS a model obtains, the better the accuracy it provides. As shown in this plot, the ANN-based methodologies, ELM and CAGE, outperform the temporal methods PP, QR, and KDE. ELM achieves 1.24% and 1.38% better CRPS on average over all time horizons compared to KDE and QR, respectively. KDE has slightly better performance in comparison with QR for 30-min up to 2.5-hr ahead predictions. The better accuracy of KDE becomes more noticeable in the horizon range of 3 hr up to 4.5 hr. Similar superiority is reflected by the better reliability curve of KDE compared to QR in Fig. 3.9. Among all temporal benchmarks, PP has the worst performance. This model has 1.77% and 1.49% more CRPS on average for 6-hr prediction, compared to KDE and QR, respectively. As the forecast horizon length grows, the CRPS of PP increases by larger amounts compared to other benchmarks. This is due to low generalization capability and erroneously high sharpness (low entropy as shown in Fig. 3.11) which results in unreliable predictions, especially when the weather condition changes from sunny to cloudy since this approach suffers from the naïve smoothness assumption. As depicted in Fig. 3.13, CGAE shows better performance in comparison with all temporal models because of its high reliability (shown by Fig. 3.8 and Fig. 3.9) and appropriate sharpness (i.e., moderate PIAW and entropy in Fig. 3.10, Fig. 3.11 and Fig. 3.12). CGAE outperforms ELM by 2.98% CRPS for hourly prediction, which is increased significantly for time horizons of length more than 3 hours and reaches the 4.90% CRPS improvement for 6-hr ahead predictions.

The spatiotemporal models generally have smaller CRPS due to modeling the spatial behavior of GHI observations as well as the temporal characteristics. For instance, the ST-Copula leads to 1.29% CRPS improvement compared to ELM for hourly predictions. Moreover, the ST-QR-Lasso model obtains 3.29% better average CRPS over all time horizons compared to its temporal version i.e. QR. While CSTF and ST-SVR obtain close CRPS curves especially for time horizons longer than 4 hours, the ST-QR-Lasso significantly dominates with lower CRPS values. The better performance of ST-QR-Lasso is mainly due to directly handling the high dimensionality and overfitting issues that characterize the use of large amounts of data. In fact, the Lasso technique is very

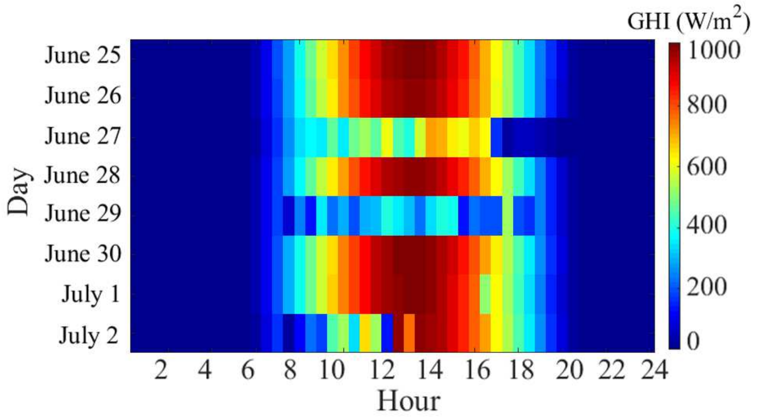


Figure 3.14. Observed GHI data from June 25^{th} to July 2^{nd}.

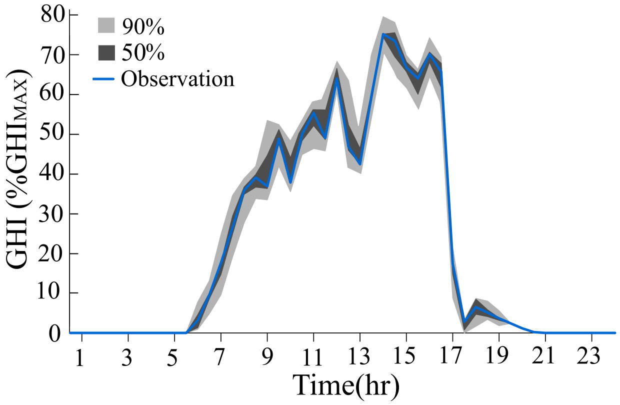


Figure 3.15. Estimated solar irradiance on June 27^{th}

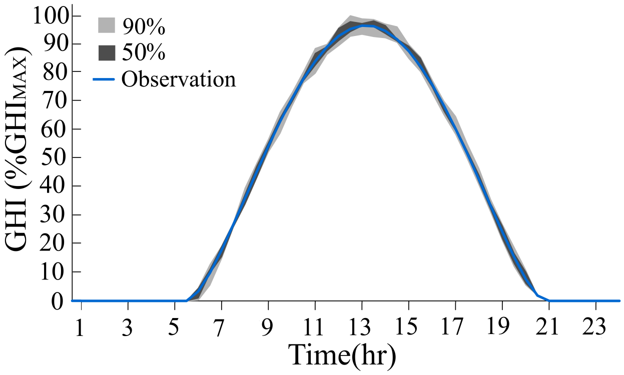


Figure 3.16. Estimated solar irradiance on June 28^{th}

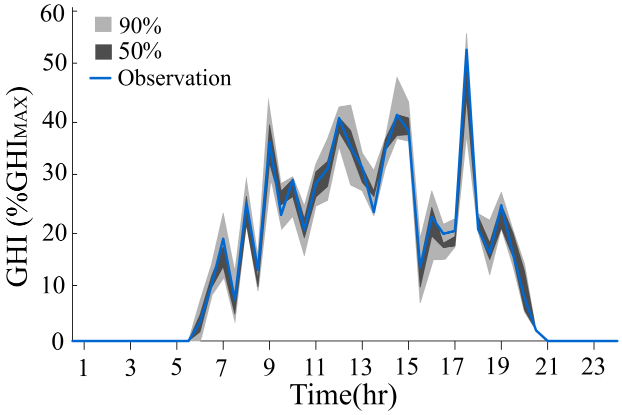


Figure 3.17. Estimated solar irradiance on June 29^{th}

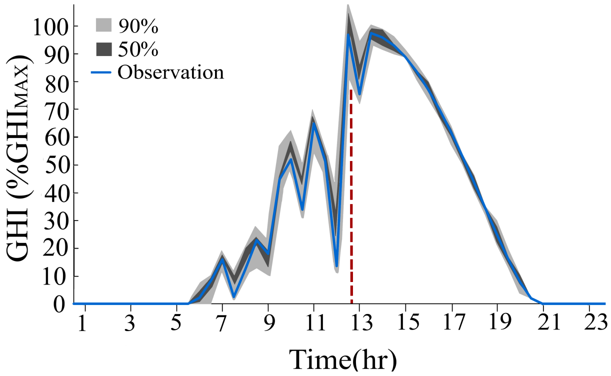


Figure 3.18. Estimated solar irradiance on July 2^{nd}

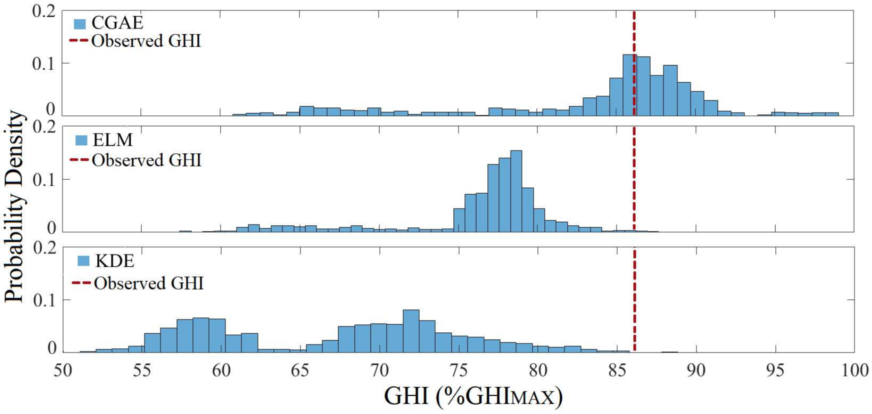


Figure 3.19. Histogram of predicted GHI for July 2^{nd} 12:30 PM

useful to reduce the likelihood of overfitting for most practical applications where a large number of observations are available. CGAE obtains 2.53% better CRPS in comparison with the ST-QR-Lasso. Although both models use L1-regularization techniques to avoid overfitting, the CGAE model obtains better accuracy due to providing a very large hypothesis space which leads to better generalization capacity.

3.4.5. Qualitative Results

The probabilistic prediction of CGAE is investigated to show the capability of our model under different weather conditions. Fig. 3.14 shows the GHI values of eight days, from June 25^{th} to July 2^{nd} in 2016, for a site near the Michigan Lake. As shown in this plot, the selected days contain various weather conditions including sunny, partly cloudy, and overcast, in a short period of time. June 25^{th} and 26^{th} are both sunny with high GHI, while the subsequent day, June 27^{th}, is mostly cloudy with many variations. The next day, June 28^{th} is sunny with high GHI while June 29^{th} is overcast with very small irradiance. June 30^{th} and July 1^{st} are sunny, and the last day, July 2^{nd} is a combination of partly cloudy and sunny. This test case evaluates the performance of CGAE when the weather changes dramatically from one day to the other, and within each day. As shown in Fig. 3.15-3.18, the prediction intervals of CGAE with 50% and 90% confidence rates follow the actual GHI values with high accuracy resulting in good reliability. In Fig. 3.15, as the weather changes from sunny to partly cloudy around 9:00, the confidence boundaries expand showing the increase in the prediction uncertainty. In Fig. 3.16, June 28^{th} has a very smooth GHI curve measured on a clear sunny day, hence, the model's uncertainty is very small. In Fig. 3.17-3.18 the weather has significant changes during overcast in June 29^{th} and partly cloudy and sunny conditions in July 2^{nd}. As seen in these two figures, although the uncertainty is increased in such conditions, the model still follows the observed GHI with high reliability. On July 2^{nd}, at 12:30, the GHI jumps drastically from 12% of maximum GHI, GHIMAX, to 86%. Fig. 3.19 shows the histogram of the predicted GHI for this observation. As shown in this figure, CGAE could capture this jump more reliably having heavier probability density around 85%-90% GHIMAX. However, ELM and KDE assign a high probability to smaller values as these models are more affected by previous small measurements. Moreover, KDE does not provide enough sharpness for this example, hence, its prediction cannot be informative. Having much higher generalization capability and being able to leverage spatiotemporal information from GHI observations, our CGAE can capture uncertainties in the solar data with higher accuracy and appropriate sharpness.

3.4.6. Running Time Analysis

As mentioned in Section IV-A, our proposed model, CGAE, is trained offline using the batch gradient descent method. In the batch gradient descent with batch size k, the gradients of the error function with respect to k training samples are aggregated in each batch at each training iteration; therefore, increasing the batch size k would lead to an increase in the training speed. Fig. 3.20 depicts the effect of batch size on the training time of CGAE for the prediction tasks with different time horizons. As shown in this figure, the running time decreases with the increase of batch size. For instance, in the 1-hr ahead prediction task, $k = 50$ leads to a training time equal to 21.39 min, while using $k = 400$ takes 19.90 min.

Fig. 3.20 also shows the effect of the forecast horizon in the training time of the proposed model. As shown in this figure, for a fixed k, the training time increases as the time horizon is extended. For instance, when $k = 200$ CGAE takes 20.33 min to train its parameters for the 1-hr ahead prediction task, while the training time increases to 25.32 min for 6-hr ahead forecasts.

As discussed in Section III-D, CGAE uses a simple feed-forward approach during the testing time; therefore, our model leads to fast predictions. The average testing time of CGAE for all forecast time horizons is less than 0.35 sec; hence, the proposed approach can be effectively used for all real-world applications.

3.5. Conclusions

A novel deep generative model, Convolutional Graph Auto-encoder, is presented for a new problem, nodal distribution learning in graphs. The model captures deep convolutional features from an arbitrary graph-structured data, to learn the corresponding probability densities of nodes. Here, the problem of spatiotemporal solar irradiance forecasting is presented as a graph distribution learn-

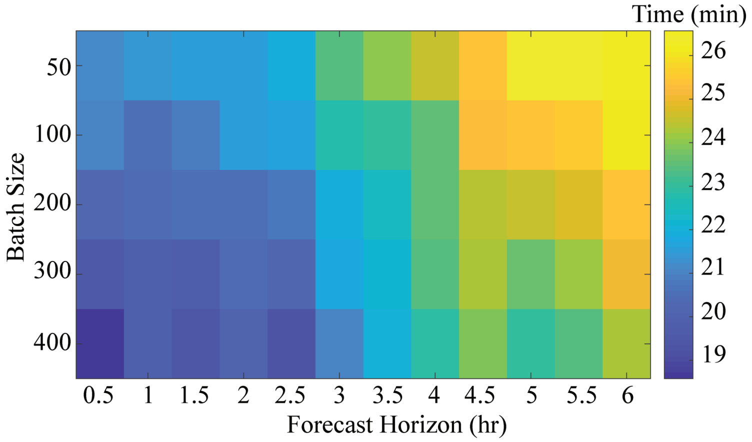


Figure 3.20. Running time of the CGAE using various batch size values.

ing problem where each node of the graph represents a solar irradiance measurement site, while each edge represents the distance between the sites. Using graph spectral convolutions, the spatial features of the solar data are extracted, that are further used by an encoding and decoding ANN to capture the distribution of future solar irradiance. Our deep learning model is used to provide probabilistic forecasts for the National Solar Radiation Database. Simulation results show better reliability, sharpness and Continuous Ranked Probability Score compared to recent baselines in the literature.

Chapter 4
Spatiotemporal Behind-the-Meter Load and PV Power Prediction via Deep Graph Dictionary Learning

In recent years, with the rapid growth of rooftop photovoltaic (PV) generation in distribution networks, power system operators call for accurate predictions of Behind-the-Meter (BTM) load and PV generation. However, the existing prediction methodologies are incapable of quantifying such BTM measurements as the smart meters can merely measure the net load time series. Motivated by this challenge, this chapter presents the spatiotemporal BTM load and PV Prediction (ST-BTMLPVP) problem. The objective is to disaggregate the historical net loads of neighboring residential units into their BTM load and PV generation, and predict the future values of these unobservable time series. To solve ST-BTMLPVP, we model the units as a spatiotemporal graph (ST-Graph) where the nodes represent the net load measurements of units and edges reflect the mutual correlation between the units. A ST-Graph autoencoder (STGAE) is devised to capture the spatiotemporal manifold of the ST-Graph, and a novel spatiotemporal graph dictionary learning (STGDL) optimization is proposed to utilize the latent features of the STGAE to find the most significant spatiotemporal features of the net load. STGDL utilizes the captured features to estimate the historical BTM load and PV measurements, which are further used by a deep recurrent structure to predict the future values of BTM load and PV generation at each unit. Numerical experiments on a real-world load and PV dataset show the state-of-the-art performance of the proposed model both for the BTM disaggregation and prediction tasks.

4.1. Introduction

The rapid increase in the penetration of renewable energy resources installed close to the customers affects the realized load profile by the distribution network operators. Quantifying the realized load plays a crucial role in determining the network operation strategies and allocating the generation capacities to serve the electricity demand [213]. Among other renewable generation technologies, photovoltaic (PV) solar generation is the prominent generation resource that could be installed behind the customers' meter [187]- [29]. Rooftop solar panels reduce the energy costs by decreasing the peak demand as well as the overall energy consumption, and improves the resilience of energy supply in extreme weather conditions.

The variability and uncertainty in Behind-the-Meter (BTM) PV generation impose several challenges for the operation of the distribution network including the violations of voltage limits, fluctuation in the voltage profile, reverse power flow, and the malfunction of protection devices in distribution networks. Forecasting the BTM residential load profile with PV generation is crucial to determine the operation decisions and handle the large variations in load and solar PV generation resources. Furthermore, incorporating demand response practices in distribution networks requires an accurate understanding of the BTM load profile.

Earlier research on electricity demand prediction addressed the prediction of residential net load (i.e., summation of BTM load and PV generation) in distribution networks using statistical models. In this categorty, the linear and nonparameteric regression is used in [120, 201] and [185]. This approach has a small computational burden due to its linear formulation; however, it is not able to accurately model the high variations in net load time series. In [158], an autoregressive integrated moving average (ARIMA) model is proposed and optimized by a particle swarm optimization algorithm to obtain the optimal parameters for residential load prediction in distribution networks. Furthermore, the authors of [93] incorporate hybrid optimization into the support vector regression (SVR) to improve the accuracy of SVR in short-term residential net load prediction. Also, the authors of [156] proposed a novel strategy for automatic time series lag selection based on SVRs, and applied their model for short-term predictions of electricity demand in residential units. Gaussian Process Regression (GPR) [164] is another major class of methods for load forecasting where a nonparametric Bayesian model computes the probability distribution of load magnitude over all admissible functions that fit the load data. The study in [139] presents an enhanced version of GPR that makes use of a hybrid structure with multiple Gaussian Processes to improve the prediction accuracy of classic GPR. In this class of models, the research in [190] presents a probabilistic GPR algorithm with high reliability and sharpness for short-term residential load predictions. Also, the authors of [233] presented an integrated GPR that computes the joint probability of load magnitude for multiple customers to address the load uncertainties caused by distributed energy resources.

In recent years, the developments in artificial neural networks (ANNs) introduced novel ANN-based load prediction approaches capable of modeling highly varying load time series and capturing the uncertainties in load datasets. In this domain, the authors of [136] presented a combination of wavelet decomposition and second-order gray ANNs to extract the nonlinear features of residential load measurements for short-term predictions. Also, a feedforward neural network is designed in [178] to decompose the electricity demand of residential units into active and reactive loads, and forecast their future values in a real-time fashion. In this group of models, an improved wavelet ANN is proposed by [182] to decompose residential demand into its most significant components. These components are further used by an extreme learning machine to predict the future load values.

Recent data-driven load prediction methodologies employ deep ANNs, which is a cutting-edge family of ANNs that can train a large number of computational layers [85]. These models have high generalization capability resulted from their large parameter space; thus, they can effectively model large and frequent changes in load measurements. In this class of methods, the long short-term memory (LSTM) network [98, 124, 125] is widely used as a deep recurrent model that can learn complex temporal patterns from load datasets. In this line of research, gated recurrent units (GRUs) [206, 245] are utilized as smaller versions of the LSTM network with less number of parameters which would lead to better generalization capability and smaller probability of overfitting when the number of training samples are limited.

The prediction of BTM load while considering BTM rooftop PV generation is a crucial task for utilities to ensure a reliable and secure power system operation; However, since the smart meters can merely measure the net load, determining the BTM load profile of the customer as well as the BTM PV generation is challenging. Although recent net load prediction models [139, 164, 190, 233] provide accurate estimations for the future net load, they are unable to give a meaningful approximation of the BTM load and PV generation as they lack decomposition procedures. In recent years, energy disaggregation algorithms employ decomposition techniques such as sparse coding [171], dictionary learning (DL) [118], and hidden Markov modeling [55] to decompose the net load of residential buildings into the load of appliances; however, the disaggregation and prediction of BTM load and BTM PV generation is not addressed since these methodologies cannot handle the highly nonlinear variations in PV generation measurements and provide a mapping between the decomposed load signals and the future BTM load and PV generation.

4.1.1. chapter Contributions

This chapter presents and solves the Behind-the-Meter Load and PV Prediction (BTMLPVP) problem as a new problem in the area of power systems. The problem is to disaggregate the historical net load signal of a residential unit into its historical BTM load and BTM PV generation values, and forecast the future BTM load and PV generation based on these estimations.

New studies have discovered a crucial correlation between the amount of load at a target residential unit and the load of its surrounding houses [68]. In addition to load, the amount of PV generation of a target house is shown to be highly correlated to the houses in its vicinity as the solar radiation and cloud cover measurements have similar values in close regions inside the spatial domain [129, 149]. In order to utilize these correlations to improve the BTMLPVP accuracy, we extend the BTMLPVP problem to the spatiotemporal BTMLPVP (ST-BTMLPVP) where the problem is solved for multiple neighboring residential units.

We cast the spatiotemporal BTMLPVP to a novel deep spatiotemporal graph dictionary learning (DeepSTGDL) problem, where the net load time series at each residential unit is considered as a node of a spatiotemporal graph (ST-graph) while the edges reflect the correlations between the net load measurements at the corresponding nodes. A new spatiotemporal graph autoencoder (ST-GAE) is developed to observe the ST-graph and reconstruct its nodes and edges, hence, learning the nonlinear spatiotemporal manifold of the net load data. Using ST-GAE, a novel optimization is proposed for DeepSTGDL that extracts the most significant spatiotemporal net load patterns from the ST-graph, and applies those patterns to disaggregate the net load time series into the historical

BTM load and PV for each node. The presented optimization forecasts the future values of BTM load and PV using the estimated historical BTM values in a deep recurrent fashion. The major contributions of our work are:

1) The spatiotemporal BTMLPVP problem is defined and solved for the first time in the area of power systems. The solution to this problem is very crucial for electricity utilities as it provides an estimation for future BTM load and PV values which are not observable by utilities. Spatiotemporal BTMLPVP is the first load disaggregation problem that takes into account the spatial relationships between net load measurements.

2) A novel spatiotemporal graph autoencoder is designed to extract deep spatial and temporal features from net load datasets. This deep learning model can be utilized to extract powerful spatiotemporal features for a large variety of spatiotemporal applications in power systems including wind and solar energy prediction.

3) The presented DeepSTGDL algorithm is the first dictionary learning algorithm that extracts a dictionary of patterns from a spatiotemporal dataset. While the most recent studies in energy disaggregation [61,76,118] and signal decomposition [174,228,256] merely compute a dictionary of temporal patterns from the net load, we study spatiotemporal dictionary learning for the first time in the domain of machine learning.

This chapter is organized as the following: Section 4.2 defines the new BTMLPVP problem and its spatiotemporal extension. Section 4.3 presents our DeepSTGDL model that employs a novel optimization to solve the spatiotemporal BTMLPVP problem. In Section 4.4, the deep learning implementation of DeepSTGDL using ST-GAE is explained. Moreover, Section 4.4.3 shows the formulations to optimize the presented deep learning model. Section 4.6 shows the numerical results on a real-world dataset. Finally, the conclusions of this research are discussed in Section 4.7.

4.2. Problem Formulation

In this section, first, the Behind The Meter Load and PV Prediction (BTMLPVP) is introduced as a novel problem in the area of power systems. Then, the problem is further expanded to a new spatiotemporal graph learning problem that we seek to solve in this study.

4.2.1. BTMLPVP Problem

Let us assume n residential units (houses) $V = \{v^i\}_{i=1}^{n}$ in a wide area. For each unit v^i that consumes L_t^i kW of electricity at time t and generates PV_t^i kW of solar energy at that time instance, the net load $NL_t^i = L_t^i - PV_t^i$ is measured by a smart meter. At each time instance t' the electricity supplier is able to observe $NL_{t'}^i$; however, the BTM measurements including the BTM load $L_{t'}^i$ and BTM PV generation $PV_{t'}^i$ are not accessible. Hence, the prediction of future BTM load values $L_{t'+k}^i$ $(0 \leq k)$ as well as future BTM PV generation $PV_{t'+k}^i$ $(0 \leq k)$ is a challenging problem.

To predict $L_{t'+k}^i$ at time t', one needs to first give an approximation of $m+1$ historical loads $h_{t'}^{L,i} = \left\langle L_{t'-m}^i, ..., L_{t'-1}^i, L_{t'}^i \right\rangle$ and learn a nonlinear mapping (prediction function) $g^L(h_{t'}^{L,i}) = L_{t'+k}^i$.

Similarly, to estimate the future PV generation $PV_{t'+k}^i$ at time t', we need to first estimate $m+1$ historical PV generation values $h_{t'}^{PV,i} = \left\langle PV_{t'-m}^i, ..., PV_{t'-1}^i, PV_{t'}^i \right\rangle$, and learn a nonlinear mapping $g^{PV}(h_{t'}^{PV,i}) = PV_{t'+k}^i$. Therefore, we define the BTMLPVP problem as two major objectives:

4.2.1.1. Net load Disaggregation

For each unit i at time instance t', the measured historical net load $h_{t'}^{NL,i} = \left\langle NL_{t'-m}^i, ..., NL_{t'-1}^i, NL_{t'}^i \right\rangle$ should be disaggregated into its two components, that is, the BTM load $h_{t'}^{L,i}$ and BTM PV generation $h_{t'}^{PV,i}$.

4.2.1.2. Load and PV Prediction

For each unit i at time instance t', when the historical measurements $h_{t'}^{L,i}$ and $h_{t'}^{PV,i}$ are estimated, one can learn the two nonlinear functions $g^L(.)$ and $g^{PV}(.)$ to compute the future BTM energy consumption $L_{t'+k}^i$ and PV generation $PV_{t'+k}^i$ for any forecast horizon $0 \leq k$.

Since the results of the net load disaggregation stage (i.e., $h_{t'}^{L,i}$ and $h_{t'}^{PV,i}$ for all units $1 \leq i \leq n$) are directly used by the prediction stage, we seek to solve both problems simulteneously, hence, avoiding suboptimal local estimations. In other words, we use the information obtained from the net load disaggregation stage to tune the prediction functions $g^L(.)$ and $g^{PV}(.)$. Also, we let the information obtained from learning these functions to improve the net load disaggregation accuracy.

4.2.2. Spatiotemporal BTMLPVP

We cast the BTMLPVP problem to a novel graph learning problem to simultaneously solve the net load disaggregation as well as load and PV prediction problems in a spatiotemporal manner. The idea is to model the n residential units at each time instance $t = t'$ as a weighted undirected ST-graph $\mathcal{G}_{t'}$ defined by a tensor of $m + 1$ graphs written as $\mathcal{G}_{t'} = \left\langle G_{t'-m}, ..., G_{t'-1}, G_{t'} \right\rangle$. Each undirected graph $G_t = \left\langle V, E_t, F_t \right\rangle$ $(1 \leq t \leq t')$ represents a snapshot of all units at time t. G_t has three elements: $V = \{v^i\}_{i=1}^n$ is the set of n nodes where v^i represents the i-th residential unit; $E_t = \{e_t^{i,j}\}_{i,j=1,2,...,n}$ is the set of edges corresponding to G_t where each $e_t^{i,j}$ shows the edge weight between v^i and v^j at time t. The edge weight $e_t^{i,j}$ reflects the correlation between the net load measurements $h_t^{NL,i}$ and $h_t^{NL,j}$ by:

$$e_t^{i,j} = \begin{cases} 0 & \text{if } MI_t(i,j) < \tau \\ e^{-Euc_t(i,j)} & \text{if } MI_t(i,j) \geq \tau \end{cases} \tag{4.1}$$

where $MI_t(i,j)$ is the mutual information between $h_t^{NL,i}$ and $h_t^{NL,j}$ while $Euc_t(i,j)$ is the Euclidean distance between the two time series. As computed in (4.1), $e_t^{i,j}$ is a function of the distance between the historical measurements when the MI is larger than a threashold τ, and zero otherwise. Note that, we do not use load or PV power measurements to compute the correlations between the units as these BTM measurements are not observed by the electricity provider. $F_t = \left\langle F_t^1, F_t^2, ..., F_t^n \right\rangle$ is the tensor of features (measurements) at all nodes (units) where each $F_t^i = NL_t^i$ is the measured net load of v^i at time t.

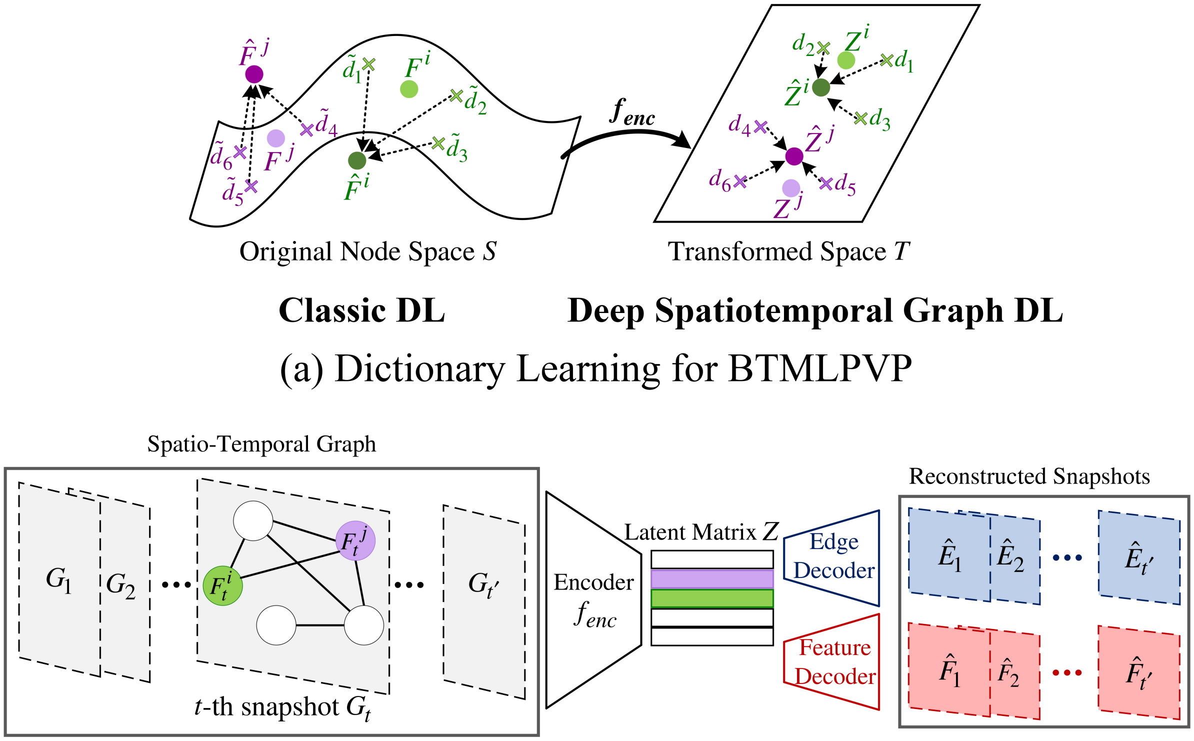
Similar to section II-A, here, the spatiotemporal BTMLPVP problem is to estimate the historical $h_{t'}^{L,i}$ and $h_{t'}^{PV,i}$ for all nodes $v^i \in V$ by observing the ST-graph $\mathcal{G}_{t'}$. Also, computing these historical BTM measurements, we seek to predict the future BTM load $L_{t'+k}^i$ and BTM PV power

$PV^i_{t'+k}$ $(0 \leq k)$.

4.3. Deep Spatiotemporal Graph Dictionary Learning for ST-BTMLPVP

To solve the ST-BTMLPVP problem presented in Section II-B, one needs to encode the highly nonlinear spatiotemporal patterns of the input data $\mathcal{G}_{t'}$ (i.e., F^i_t for all $1 \leq i \leq n$ and $t' - m \leq t \leq t'$). Also, we need to compute nonlinear mappings that transform these patterns into an estimation for $h^{L,i}_{t'}$ and $h^{PV,i}_{t'}$, as well as $L^i_{t'+k}$ and $PV^i_{t'+k}$ $(0 \leq k)$.

To address this problem, first, we propose a classic dictionary learning solution. Then, the drawbacks of classic DL are analyzed, which motivate us to propose our DeepSTGDL model as a novel solution to ST-BTMLPVP.



(a) Dictionary Learning for BTMLPVP

(b) Structure of the Spatiotemporal Graph Autoencoder

Figure 4.1. Classic DL and Deep Spatiotemporal Graph Dictionary Learning for ST-graph

4.3.1. Classic DL: Motivations and Drawbacks

As shown in Section II-A, at each time instance t', each node v^i $(i = 1, 2, ..., n)$ in the ST-graph $\mathcal{G}_{t'}$ contains a $(m+1)$-dimensional net load measurement vector $F^i = h_{t'}^{NL,i}$. To capture and encode the spatiotemporal net load patterns of $\mathcal{G}_{t'}$, one can compute a dictionary of K atoms (patterns) $D =< d_1, d_2, ..., d_K >\in \mathbb{R}^{(m+1)\times K}$ using:

$$F^i \simeq \hat{F}^i = D\ a^i = \sum_{k=1}^{K} d_k a_k^i \tag{4.2}$$

where $a^i \in \mathbb{R}^K$ is a sparse coefficient vector that determines the contribution of each pattern d_k in the construction of feature vector $\hat{F}^i \simeq F^i$ for the i-th node. a^i can be viewed as a K-dimensional overcomplete sparse code that can efficinetly represent F^i. Therefore, to solve the ST-BTMLPVP, we define the nonlinear function $g_L(a^i)$ to estimate $h_{t'}^{L,i}$, and $g_{PV}(a^i)$ to estimate $h_{t'}^{PV,i}$. Moreover, two nonlinear functions $f_L(g_L(a^i))$ and $f_{PV}(g_{PV}(a^i))$ are defined to predict $PV_{t'+k}^i$ and $L_{t'+k}^i$ for any $0 < k$, respectively.

To compute the optimal dictionary D^* as well as optimal coefficient vectors $a^{i,*}$ for all $i = 1, 2, ..., n$, one can solve the optimization:

$$\begin{aligned} D^*, \{a^{i,*}\}_{i=1}^n = \underset{D,a^i}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \Big(||F^i - D\ a^i||_2^2 + \lambda ||a^i||_1 \Big) \\ s.t.\ ||d_k||_2^2 \leq 1 \quad \forall\, 1 \leq k \leq K \end{aligned} \tag{4.3}$$

where the first term of the summation is computing the dictionary encoding error while the second term adds a regularization loss with coefficient $0 < \lambda$ to ensure the sparsity of the computed code $a^{i,*}$ for all nodes i. The condition $||d_k||_2^2 \leq 1$ restricts the magnitude of the dictionary atoms to avoid naive solutions with arbitrarily small a^i.

Fig. 4.1(a) shows the encoding results of the i-th and j-th node of $\mathcal{G}_{t'}$ using the classic DL in (4.3). Three atoms $\{\tilde{d}_k\}_{k=1}^3$ are learned to estimate $\hat{F}^i \simeq F^i$ and three atoms $\{\tilde{d}_k\}_{k=4}^6$ are captured to obtain $\hat{F}^j \simeq F^j$. As shown in this figure, the resulting estimations of classic DL (i.e., $\hat{F}^i$ and $\hat{F}^j$) cannot accurately approximate their true values F^i and F^j due to two reasons:

4.3.1.1. Linearity assumption of D

The space S of $\{F^i\}_{i=1,2,...,n}$ is highly nonlinear while the dictionary formulations in (4.2) and (4.3) are linear. Therefore, D and a^i are incapable of modeling the nonlinear net load data in ST-graph $\mathcal{G}_{t'}$. In this study, we overcome this issue by devising a novel nonlinear spatiotemporal dictionary learning model for ST-BTMLPVP.

4.3.1.2. Naive correlation structure

The DL model presented in (4.2) and (4.3) ignores the correlations between $\mathcal{G}_{t'}$'s nodes encoded in the edge weights $\{E_t\}_{t=t'-m}^{t'}$. In this study, we seek to define a novel model that considers these correlations to improve the accuracy of ST-BTMLPVP.

4.3.2. DeepSTGDL Model: Objectives

To address the drawbacks of classic DL, we present DeepSTGDL as a novel graph pattern recognition model for the ST-BTMLPVP problem. Fig. 4.1(b) shows the overall structure of the proposed model. The idea is to learn a dictionary D of net load patterns in $\mathcal{G}_{t'}$ in the latent space of a spatiotemporal graph autoencoder. To solve the ST-BTMLPVP, our model has four objectives:

4.3.2.1. Deep Spatiotemporal Graph Feature Learning

To capture the spatiotemporal patterns of net load measurements in $\mathcal{G}_{t'}$, we define a new ST-GAE that observes $\mathcal{G}_{t'}$ and reconstructs it, hence, capturing meaningful patterns from the net load data stored in $\mathcal{G}_{t'}$. Learning a deep encoding function $f_{enc}(G_t) = Z_t \in \mathbb{R}^{n \times d_h}$, the ST-GAE represents $\mathcal{G}_{t'} = \left\langle G_{t'-m}, ..., G_{t'-1}, G_{t'} \right\rangle$ by a sequence of latent matrices $\Phi = \left\langle Z_{t'-m}, ..., Z_{t'-1}, Z_{t'} \right\rangle$. Each i-th row of Z_t denoted by $Z_t^i \in \mathbb{R}^{d_h}$ is the extracted spatiotemporal feature corresponding to the i-th node of the t-th snapshot G_t. At each time $t \in [t'-m, t']$, an edge decoder $f_e(Z_t)$ observes the latent code $Z_t = f_{enc}(G_t)$ and computes an adjacency estimation $\hat{E}_t \approx E_t$ while a deep node decoder $f_n(Z_t)$ estimates G_t's node feature matrix $\hat{F}_t \approx F_t$. This encoding-decoding architecture helps the ST-GAE to learn highly nonlinear spatiotemporal latent features Φ of $\mathcal{G}_{t'}$ that are powerful enough to reconstruct $\mathcal{G}_{t'}$.

4.3.2.2. Spatiotemporal Dictionary Learning

We compute $Z = \frac{1}{m+1} \sum_{t=t'-m}^{t'} Z_t$ as an average of latent matrices Z_t over the entire historical time window $t \in [t' - m, t']$. Each i-th row $Z^i \in d_h$ is an encoded feature corresponding to the i-th node v^i. For all Z^i $(1 \leq i \leq n)$, a dictionary $D = [d_1, d_2, ..., d_K] \in \mathbb{R}^{d_h \times K}$ with a sparse coefficient vector $a^i \in \mathbb{R}^K$ is computed to capture the significant spatiotemporal patterns of the data F^i $(1 \leq i \leq n)$ stored in $\mathcal{G}_{t'}$. Similar to (4.2), here, the i-th node is encoded by $\hat{Z}^i = D\ a^i \approx Z^i$. Fig. 4.1(a) depicts the dictionary atoms $\{d_k\}_{k=1}^{3}$ and $\{d_k\}_{k=4}^{6}$ used to compute $\hat{Z}^i \approx Z^i$ and $\hat{Z}^j \approx Z^j$, respectively. As shown in this figure, the dictionary estimations $\hat{Z}^i$ and $\hat{Z}^j$ are closer to their target values Z^i and Z^j compared to the classic DL since D is computed in a linear transformed space T using a deep learning transformation f_{enc}, rather than the original nonlinear ambient space S of the raw data F^i $(1 \leq i \leq n)$.

4.3.2.3. Net Load Disaggregation

As shown by (4.2), each a^i is a compressed overcomplete sparse code to represent a node v^i. Hence, the two functions $g_L(F^i, a^i) = \hat{h}_{t'}^{L,i}$ and $g_{PV}(F^i, a^i) = \hat{h}_{t'}^{PV,i}$ are learned to compute estimations of the historical BTM load and PV measurements, $\hat{h}_{t'}^{L,i} \approx h_{t'}^{L,i}$ and $\hat{h}_{t'}^{PV,i} \approx h_{t'}^{PV,i}$, for all valid i, respectively.

4.3.2.4. Load and PV Prediction

Two nonlinear recursive functions $f_L(\hat{h}_{t'}^{L,i})$ and $f_{PV}(\hat{h}_{t'}^{PV,i})$ are respectively defined to predict $L_{t'+k}^{i}$ and $PV_{t'+k}^{i}$ for all valid i and forecast horizon $0 < k$.

4.3.3. DeepSTGDL Model: Optimization

To simultaneously fulfill the four DeepSTGDL objectives presented in Section III-B, we propose a novel optimization:

$$
\begin{aligned}
\min_{\substack{D,A,f_{enc},f_e,f_n,\\ g_L,g_{PV},f_L,f_{PV}}} J = \Bigg(& J_{dic} + \lambda_e J_e + \lambda_n J_n + \lambda_L J_L + \lambda_{PV} J_{PV} \\
& + \lambda_L^{pred} J_L^{pred} + \lambda_{PV}^{pred} J_{PV}^{pred} \Bigg) \\
= \Bigg[& \frac{1}{n} \sum_{i=1}^{n} \Big(|| \frac{1}{m+1} \sum_{t=t'-m}^{t'} \underbrace{f_{enc}^{i}(G_t)}_{Z_t^i} - D\, a^i ||_2^2 + \lambda ||a^i||_1^1 \Big) \\
& + \lambda_e \cdot \frac{1}{m+1} \sum_{t=t'-m}^{t'} ||E_t - f_e(Z_t)||_F^2 \\
& + \lambda_n \cdot \frac{1}{m+1} \sum_{t=t'-m}^{t'} ||F_t - f_n(Z_t)||_F^2 \\
& + \lambda_L \cdot \frac{1}{n} \sum_{i=1}^{n} ||g_L(F^i, a^i) - h_{t'}^{L,i}||_2^2 \\
& + \lambda_{PV} \cdot \frac{1}{n} \sum_{i=1}^{n} ||g_{PV}(F^i, a^i) - h_{t'}^{PV,i}||_2^2 \\
& + \lambda_L^{pred} \cdot \frac{1}{n} \sum_{i=1}^{n} ||f_L(\hat{h}_{t'}^{L,i}) - L_{t'+k}^{i}||_2^2 \\
& + \lambda_{PV}^{pred} \cdot \frac{1}{n} \sum_{i=1}^{n} ||f_{PV}(\hat{h}_{t'}^{PV,i}) - PV_{t'+k}^{i}||_2^2 \Bigg] \\
& s.t.\ ||d_k||_2^2 \leq 1 \ \forall\, 1 \leq k \leq K
\end{aligned}
\tag{4.4}
$$

where $A = [a^1, a^2, ..., a^n] \in \mathbb{R}^{K \times n}$ is the matrix of sparse coefficient vectors. J is the total error function that we seek to minimize by training several deep nonlinear functions including the encoder f_{enc}, edge decoder f_e, node decoder f_n, historical load approximator g_L, historical PV approximator g_{PV}, future load approximator f_L, and future PV approximator f_{PV}.

The first error metric J_{dic} is the ST-graph dictionary learning error that computes the L2-norm distance $||.||_2$ between each v^i's encoded latent feature Z^i and the dictionary learning approximation $\hat{Z}^i = D\ a^i$ for node v_i. J_{dic} is minimized to decrease the distance between each Z^i and its DL approximation $\hat{Z}^i$. Note that, similar to the classic DL in (4.3), the L1-norm $||.||_1$ of each a^i is calculated to avoid trivial solutions with arbitrarily small dictionary atoms. Minimization of J_{dic} not only leads to the optimization of D and A but also finds the optimal encoder f_{enc}.

J_e is the edge decoding error term that computes the squared Frobenius norm $||.||_F^2$ of the distance between the estimated adjacency matrix $\hat{E}_t = f_e(Z_t)$ and the actual adjacency E_t at time t. J_n is the node decoder error defined as the average distance between the feature vector F_t at time t and its estimated value $\hat{F}_t = f_n(Z_t)$. J_L is the historical load disaggregation error that computes the distance between the estimated historical load $g_L(F^i, a^i)$ and the actual load history $h_{t'}^{L,i}$. Similarly, J_{PV} computes the PV disaggregation error. The load prediction error function J_L^{pred} computes the squared L2-norm of the distance between the estimated future load $\hat{L}_{t+k}^i = f_L(\hat{h}_{t'}^{L,i})$ and the actual value $L_{t'+k}^i$ while the PV prediction error J_{PV} computes the error of the PV prediction $PV_{t'+k}^i$ averaged over all nodes v^i $(1 \leq i \leq n)$.

4.4. DeepSTGDL: Deep Learning Implementation

To optimization (4.4), one needs to define the functions $f_{enc}, f_e, f_n, g_L, g_{PV}, f_L, f_{PV}$ and train them (optimize their parameters) using the total error J. In this section, we present a novel ST-GAE that implements f_{enc}, f_e, as well as f_n. Also, we present the neural network architectures of the disaggregation functions g_L and g_{PV} as well as the prediction functions f_L and f_{PV}.

4.4.1. Spatio-Temporal Graph Autoencoder (f_{enc}, f_e, and f_n)

As shown in Fig. 4.2, the proposed ST-GAE is an encoding-decoding neural architecture that represents each G_t $(t' - m \leq t \leq t')$ by Z_t using a novel Spatio-Temporal Long Short Term Memory (ST-LSTM) that can simulteneously capture the spatial and temporal patterns of G_t. The resulting patterns are further decoded by f_n and f_e to reconstruct G_t, hence, learning powerful Z_t that is able to compute the original data G_t. In the next sections, we use Z_t to estimate the desired variables $h_{t'}^{L,i}$, $h_{t'}^{PV,i}$, $L_{t'+k}^{i}$, and $PV_{t'+k}^{i}$.

4.4.1.1. Graph Encoder f_{enc}

As explained in section III-B and shown in Fig. 4.2, at each time $t' - m \leq t \leq t'$, $f_{enc}(G_t)$ observes the snapshot G_t and generates Z_t. Let us assume we are at time t. To extract the spatial features of snapshot G_t, the graph feature matrix $\tilde{F}_t = F_t^T \in \mathbb{R}^{n \times 1}$ is filtered by a non-parametric kernel $\mathcal{K}(\theta)$ with filtering parameters $\theta \in \mathbb{R}^n$ in the Fourier domain using:

$$O_t = \mathcal{K}(\theta) \circledast_G \tilde{F}_t = \mathcal{K}(\theta) U \Lambda U^T F_t = U \mathcal{K}(\Lambda) U^T F_t \tag{4.5}$$

where θ is the n-dimensional vector of Fourier coefficients and $U \in \mathbb{R}^{n \times n}$ is the eigenvector matrix of G_t's normalized graph Laplacian $L = I_n - D^{-\frac{1}{2}} E_t D^{-\frac{1}{2}} = U \Lambda U^T \in \mathbb{R}^{n \times n}$. I_n is the n-dimensional Identity matrix, $\Lambda \in \mathbb{R}^{n \times n}$ is the diagonal matrix of L's eigenvalues, while $D \in \mathbb{R}^{n \times n}$ is G_t's degree matrix defined by $D(i,i) = \sum_j e_t^{i,j}$ with $D(i,j) = 0$ for all $i \neq j$. One can compute the graph Fourier transformation of $\tilde{F}_t$ by $U^T \tilde{F}_t$. As shown in (4.5), one can represent the filter $\mathcal{K}(\theta)$ as a function of the eigenvalues of L; therefore, the filter is rewritten as $\mathcal{K}(\Lambda)$. The matrix multiplication of (4.5) takes $O(n^2)$ time complexity and the eigendecomposition of L leads to $O(n^3)$ time complexity; hence, to avoid computational burden for large n, we estimate $\mathcal{K}(\Lambda)$ using the Chebyshev Polynomials $P_j(x)$:

$$\mathcal{K}(\Lambda) \approx \sum_{j=0}^{J} \omega_j P_j\left(\frac{2}{\lambda_{max}} \Lambda - I_n\right) \tag{4.6}$$

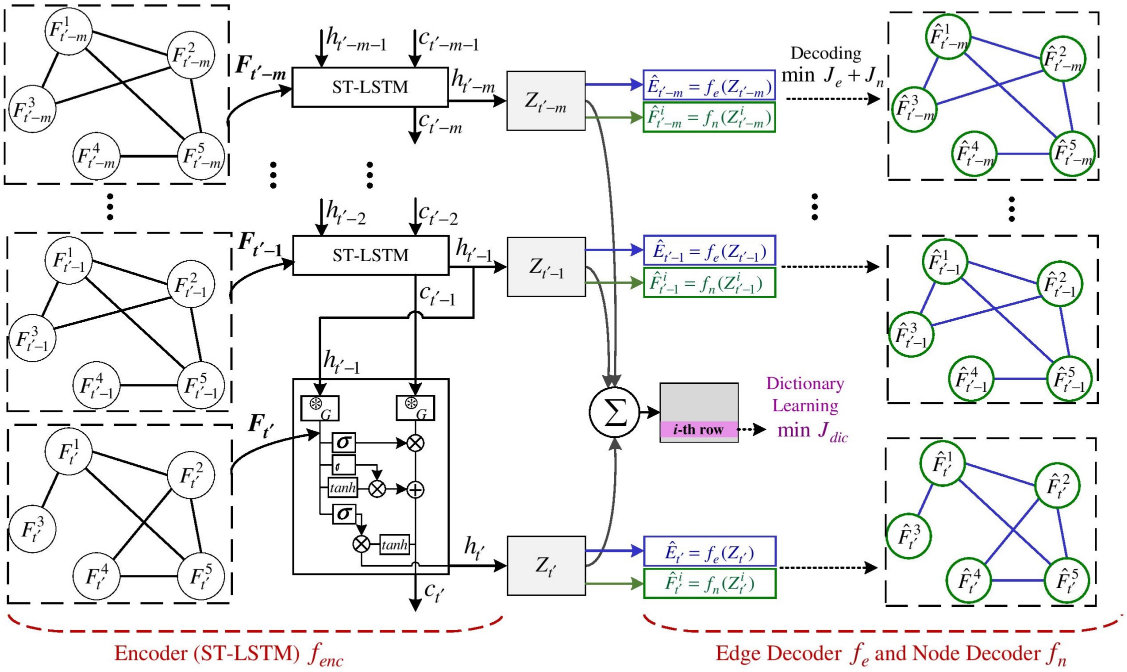
where λ_{max} is the largest eigenvalue of the Laplacian L while $\omega \in \mathbb{R}^J$ is the vector of Chebyshev coefficients and P_j is the j-th Chebyshev polynomial defined by:

$$\begin{aligned} P_{j+1} &= 2x P_j(x) - P_{j-1}(x) \ (j > 1) \\ P_0(x) &= 1, P_1(x) = x \end{aligned} \tag{4.7}$$

Having (4.6), the convolution O_t in (4.5) can be computed using the scaled Laplacian matrix ζ_t for each G_t:

$$\begin{aligned} O_t &= \mathcal{K}(\theta) \circledast_G \tilde{F}_t \approx \sum_{j=0}^{J} \omega_j P_j(\zeta_t)\tilde{F}_t \\ \zeta_t &= \frac{2}{\lambda_{max}} L - I_n \end{aligned} \tag{4.8}$$

Note that (4.8) is a J-ordered polynomial with Chebyshev terms P_j $(1 \leq j \leq J)$; hence, one can view (4.8) as a J-localized formulation. In other words, at each time t, to extract the spatial features of v^i (i.e. F_t^i), the filtering operation $\circledast_G$ observes the data F_t^j at all nodes v^j that are J nodes away from v^i according to the adjacency matrix E_t.



$F_{t'-m}^1$ $F_{t'-m}^2$ $F_{t'-m}^3$ $F_{t'-m}^4$ $F_{t'-m}^5$
$F_{t'-m}$
$h_{t'-m-1}$
$c_{t'-m-1}$
ST-LSTM
$h_{t'-m}$
$c_{t'-m}$
$Z_{t'-m}$
$\hat{E}_{t'-m} = f_e(Z_{t'-m})$
$\hat{F}_{t'-m}^i = f_n(Z_{t'-m}^i)$
Decoding
min $J_e + J_n$
$\hat{F}_{t'-m}^1$ $\hat{F}_{t'-m}^2$ $\hat{F}_{t'-m}^3$ $\hat{F}_{t'-m}^4$ $\hat{F}_{t'-m}^5$
$h_{t'-2}$
$c_{t'-2}$
$F_{t'-1}$
ST-LSTM
$h_{t'-1}$
$c_{t'-1}$
$Z_{t'-1}$
$\hat{E}_{t'-1} = f_e(Z_{t'-1})$
$\hat{F}_{t'-1}^i = f_n(Z_{t'-1}^i)$
$F_{t'-1}^1$ $F_{t'-1}^2$ $F_{t'-1}^3$ $F_{t'-1}^4$ $F_{t'-1}^5$
$\hat{F}_{t'-1}^1$ $\hat{F}_{t'-1}^2$ $\hat{F}_{t'-1}^3$ $\hat{F}_{t'-1}^4$ $\hat{F}_{t'-1}^5$
$h_{t'-1}$
$\circledast_G$
$\circledast_G$
σ
tanh
σ
tanh
$F_{t'}$
$h_{t'}$
$c_{t'}$
$Z_{t'}$
Σ
i-th row
Dictionary
Learning
min J_{dic}
$\hat{E}_{t'} = f_e(Z_{t'})$
$\hat{F}_{t'}^i = f_n(Z_{t'}^i)$
$F_{t'}^1$ $F_{t'}^2$ $F_{t'}^3$ $F_{t'}^4$ $F_{t'}^5$
$\hat{F}_{t'}^1$ $\hat{F}_{t'}^2$ $\hat{F}_{t'}^3$ $\hat{F}_{t'}^4$ $\hat{F}_{t'}^5$
Encoder (ST-LSTM) f_{enc}
Edge Decoder f_e and Node Decoder f_n

Figure 4.2. Structure of the Proposed Spatiotemporal Graph Autoencoder

Our graph encoder f_{enc} is a novel Spatio-Temporal Long Short-term Memory Network (ST-LSTM) that incorporates the convolution filter $\circledast_G$ of (4.8) with LSTM to simulteneously capture the spatial as well temporal featurs of G_t. At each iteration $t \in [t'-m, t']$, the ST-LSTM observes a snapshot $G_t =< V, E_t, F_t >$ and computes the latent spatio-temporal feature matrix $Z_t = f_{enc}(G_t)$

using:

$$x_t = \tilde{F}_t$$

$$\begin{aligned}
i_t &= \sigma(x_t W_{xi} + \sum_{j=0}^{J} \omega_j^{i,z} P_j(\zeta_{t-1}) Z_{t-1} + b_i) \\
f_t &= \sigma(x_t W_{xf} + \sum_{j=0}^{J} \omega_j^{f,z} P_j(\zeta_{t-1}) Z_{t-1} + b_f) \\
\bar{c_t} &= tanh(x_t W_{xc} + \sum_{j=0}^{J} \omega_j^{c,z} P_j(\zeta_{t-1}) Z_{t-1} + b_c) \\
c_t &= f_t \odot \Big(\sum_{j=0}^{J} \omega_j^{c} P_j(\zeta_{t-1}) c_{t-1} \Big) + i_t \odot \bar{c_t} \\
o_t &= \sigma(x_t W_{xo} + \sum_{j=0}^{J} \omega_j^{o,z} P_j(\zeta_{t-1}) Z_{t-1} + b_o) \\
Z_t &= o_t \odot tanh(c_t)
\end{aligned} \tag{4.9}$$

where $\sigma(.)$ and $tanh(.)$ are the sigmoid and tangent hyperbolic functions, respectively. At each iteration t, the ST-LSTM observes input $x_t = \tilde{F}_t$ and computes the input gate signal $i_t \in \mathbb{R}^{n \times d_h}$ using a sigmoid with weight $W_{xi} \in \mathbb{R}^{d_h}$, convolution coefficients $\omega_{1 \le j \le J}^{i,z} \in \mathbb{R}$, and bias $b_i \in \mathbb{R}^{n \times d_h}$. Also, the forget gate f_t is computed as a sigmoid with weight $W_{xf} \in \mathbb{R}^{d_h}$, convolution coefficients $\omega_{1 \le j \le J}^{f,z} \in \mathbb{R}$, and bias $b_f \in \mathbb{R}^{n \times d_h}$. While the input gate i_t decides the amount of information to store in the ST-LSTM's temporal memory $c_t \in \mathbb{R}^{n \times d_h}$ at time t, the forget gate f_t represents the information c_t can forget. $\omega_{1 \le j \le J}^{c} \in \mathbb{R}$ are the convolution filtering parameters of the memory unit. ST-LSTM's memory is updated by the update matrix $\bar{c_t} \in \mathbb{R}^{n \times d_h}$ with weight $W_{xc} \in d_h$, filtering coefficients $\omega_{1 \le j \le J}^{c,z} \in \mathbb{R}$, and bias b_c. The output $o_t \in \mathbb{R}^{n \times d_h}$ is computed as a sigmoidal function with output weight $W_{xo} \in \mathbb{R}^{d_h}$, filtering coefficients $\omega_{1 \le j \le J}^{o,z}$, and bias b_o. At each iteration t, the temporal feature matrix $Z_t \in \mathbb{R}^{n \times d_h}$ is computed as a function of ST-LSTM's temporal output o_t and the memory c_t.

4.4.1.2. Edge Decoder f_e

The edge decoder $\hat{E}_t = f_e(Z_t)$ estimates the weight at each edge $e_t^{i,j}$ $(1 \leq i, j \leq n)$. Thus, $\hat{E}_t$ is probabilistically defined by:

$$\begin{aligned} P(\hat{E}_t) &= \prod_{i=1}^{n} \prod_{j=1}^{n} P(\hat{E}_t^{i,j} | Z_t^i, Z_t^j) \\ P(\hat{E}_t^{i,j} = 1 | Z_t^i, Z_t^j) &= \sigma(Z_t^i * Z_t^{j^T}) \end{aligned} \tag{4.10}$$

where $\hat{E}_t^{i,j}$ is the element at i-th row and j-th column of $\hat{E}_t$.

4.4.1.3. Node Decoder f_n

The node decoder $\hat{F}_t = f_n(Z_t)$ is modeled by a deep Rectified Linear Unit (ReLU) neural network with $\mathcal{N}$ activation layers. The neural network observes each row Z_t^i $(1 \leq i \leq n)$ as its input, and estimates the corresponding node feature F_t^i. Thus, for each row Z_t^i $(1 \leq i \leq n) \in \mathbb{R}^{d_h}$, the input layer is defined as $O^0 = Z_t^i$ while Each layer $1 \leq l \leq \mathcal{N}$ is defined by:

$$O^l = ReLU(W^l * O^{l-1} + b^l) \tag{4.11}$$

where $ReLU$ is the Rectified Linear Unit function while W^l and b^l are the weight and bias of the l-th layer. The output layer $O^{\mathcal{N}} = \hat{F}_t^i$ is an estimation of the true F_t^i; hence, f_n computes $\hat{F}_t \approx F_t$ in n steps. At each step i, the input $O^0 = Z_t^i \in \mathbb{R}^{d_h}$ is fed to the neural network and $O^{\mathcal{N}} = \hat{F}_t^i \in \mathbb{R}$ is computed as the output.

4.4.2. Disaggregation Functions g_L and g_{PV}

Both $g_L(F^i, a^i)$ and $g_{PV}(F^i, a^i)$ are implemented as deep ReLU neural networks with $\mathcal{N}_L$ and $\mathcal{N}_{PV}$ number of activation layers with similar formulation in (4.11). The input layer of $g_L(F^i, a^i)$ is defined by $O^0 =< F^i, a^i >\in \mathbb{R}^{m+K}$ with output $O^{\mathcal{N}_L} = \hat{h}_{t'}^{L,i} \in \mathbb{R}^m$. Similarly, the input of $g_{PV}(F^i, a^i)$ is $O^0 =< F^i, a^i >\in \mathbb{R}^{m+K}$ and the output vector is defined as $O^{\mathcal{N}_{PV}} = \hat{h}_{t'}^{PV,i} \in \mathbb{R}^m$.

4.4.3. Prediction Functions f_L and f_{PV}

Both load prediction f_L and PV prediction f_{PV} functions can be implemented by any recurrent neural network. Here, to provide high generalization while avoiding the overfitting problem, we implement f_L by a Gated Recurrent Unit (GRU) with input $\hat{h}_{t'}^{PV,i}$ and output $\hat{L}_{t'+k}^{i} \approx L_{t'+k}^{i}$. Similarly, f_{PV} is implemented by a GRU with input $\hat{h}_{t'}^{PV,i}$ and output $\hat{PV}_{t'+k}^{i} \approx PV_{t'+k}^{i}$.

4.5. DeepSTGDL: Optimization Algorithm

This section provides an optimization algorithm to solve the DeepSTGDL optimization proposed in (4.4) using functions implemented in Section IV. Algorithm 1 trains D, A, as well as all parameters of f_{enc} , f_e, f_n, g_L, g_{PV}, f_L and f_{PV} to obtain the optimal objective J in (4.4). The three major steps of this algorithm are mathematically defined in this section.

Algorithm 1: DeepSTGDL Optimization

```
Input: Spatio-Temporal Graph $\mathcal{G}_{t'}$, Historical load measurement $\{h_{t'}^{L,i}\}_{i=1}^{n}$, Historical PV
measurement $h_{t'}^{PV,i}{}_{i=1}^{n}$, future load $\{L_{t'+k}^{i}\}_{i=1}^{n}$, and future PV power $\{PV_{t'+k}^{i}\}_{i=1}^{n}$
while While $D$ and $A$ not converged do
    1) Deep Learning: Update functions $f_{enc}$ , $f_e$, $f_n$, $g_L$, $g_{PV}$, $f_L$ and $f_{PV}$ Using GD in
    (4.12) and (4.13).
    2) Dictionary Learning: Find the optimal dictionary $D^*$ using (4.19).
    3) Sparse Coefficient Matrix Learning: Find the optimal sparse coefficient matrix $A^*$
    using (4.22).
end
```

4.5.1. Deep Learning Optimization

In this step, having a fixed D and A, the ST-GAE functions f_{enc}, f_e, and f_n are simultaneously trained with the disaggregation neural networks (i.e., g_L and g_{PV}) in addition to the prediction GRUs (i.e., f_L and f_{PV}). As all functions are implemented by deep neural networks in Section IV, one can efficiently employ the gradient descent method to train their parameters. For instance, to update W_{xo} in (4.9) at each time instance t, the gradient of objective J w.r.t this parameter is computed as:

$$\begin{aligned}\frac{\partial J}{\partial W_{xo}} = \frac{\partial J}{\partial Z_t} \times \frac{\partial Z_t}{\partial o_t} \times \frac{\partial o_t}{\partial W_{xo}} = \frac{2}{n}\sum_{i=1}^{n}\Bigg[\bigg(||\frac{1}{m+1} \\ Z_t^i - D\ a^i||_2) \cdot (tanh(c_t^i)) \cdot (o_t^i(1-o_t^i)) \cdot x_t^i\Bigg]\end{aligned} \tag{4.12}$$

where o_t^i and x_t^i are the i-th row of x_t and o_t respectively. The GD updates W_{xo} by:

$$W_{xo}^{new} \leftarrow W_{xo}^{old} - \eta\frac{\partial J}{\partial W_{xo}} \tag{4.13}$$

where $\eta \in [0,1]$ is GD's learning rate. All deep neural network parameters are updated using GD with formulations similar to 4.12 and 4.13.

4.5.2. Dictionary Optimization (Learning D)

To update D, the functions f_{enc} , f_e, f_n, g_L, g_{PV}, f_L and f_{PV} as well as the sparse matrix A are fixed, The optimizaiton (4.4) is rewritten as a least squares problem with quadratic constraints:

$$\begin{aligned}&\min_D \tilde{J} = J_{dic} = \frac{1}{n}\sum_{i=1}^{n}\bigg(||\frac{1}{m+1}\sum_{t=t'-m}^{t'} f_{enc}^i(G_t) - D\ a^i||_2^2 \\ &= \frac{1}{n}\sum_{i=1}^{n}\bigg(||Z^i - D\ a^i||_2^2\ \ s.t.\ \ ||d_k||_2^2 \leq 1\ \ \forall\, 1 \leq k \leq K\end{aligned} \tag{4.14}$$

To efficiently solve (4.14), we apply the Lagrange multipliers. Let us define the Lagrangian as:

$$(D,\mu) = \frac{1}{n}\sum_{i=1}^{n}||Z^i - Da^i||_2^2 + \sum_{k=1}^{K}\mu_k(||d_k||_2^2 - 1) \tag{4.15}$$

where $\{\mu_k \geq 0\}_{k=1}^{K}$ are nonnegative Lagrangian multipliers. To find the analytic solution of D, we solve $\frac{\partial \mathcal{L}(D,\mu)}{\partial D} = 0$; hence, the optimal dictionary D^* is computed by:

$$\begin{aligned} D &= \mathcal{Z}A^T(AA^T + \Psi)^{-1} \\ \mathcal{Z} &= [f_{enc}^1(G_t), f_{enc}^2(G_t), ..., f_{enc}^n(G_t)] \\ &= [Z^1, Z^2, ..., Z^n] \in \mathbb{R}^{d_h \times n} \\ \Psi &= n\ diag(\mu) \in \mathbb{R}^{K \times K} \end{aligned} \tag{4.16}$$

Thus, one can write the Lagrangain dual function as:

$$\begin{aligned} \mathcal{L}_{dual}(\mu) &= \min_D \mathcal{L}(D, \mu) \\ &= \frac{1}{n}\sum_{i=1}^{n} ||f_{enc}^i(G_t) - \mathcal{Z}A^T(AA^T + \Psi)^{-1}a^i||_2^2 \\ &+ \sum_{k=1}^{K} \mu_k(||\mathcal{Z}A^T(AA^T + \Psi)^{-1}u_k||_2^2 - 1) \end{aligned} \tag{4.17}$$

where $u_k \in \mathbb{R}^K$ is the k-th unit vector. To solve (4.17), we maximize the Lagrangaian dual $\mathcal{L}_{dual}(\mu)$ w.r.t the variables $\{\mu_k\}_{k=1}^{K}$ using GD similar to the update rule in (4.13). The gradient of $\mathcal{L}_{dual}(\mu)$ w.r.t any μ_k $(1 \leq k \leq K)$ is computed by:

$$\frac{\partial \mathcal{L}_{dual}(\mu)}{\partial \mu_k} = ||\mathcal{Z}A^T(AA^T + \Psi)^{-1}u_k||_2^2 - 1 \tag{4.18}$$

Computing the optimal Lagrangian multipliers $\mu^* =< \mu_1^*, \mu_2^*, ..., \mu_k^*, ..., \mu_K^* >$, we calculate the optimal dictionary D^* by:

$$\begin{aligned} D^* &= \mathcal{Z}A^T(AA^T(AA^T + \Psi^*)^{-1} \\ \Psi^* &= n\ diag(\mu^*) \in \mathbb{R}^{K \times K} \end{aligned} \tag{4.19}$$

4.5.3. Sparse Matrix Optimization (Learning A)

To obtain the optimal sparse matrix A^* for a fixed D, f_{enc} , f_e, f_n, g_L, g_{PV}, f_L and f_{PV}, we rewrite the optimization (4.4) as:

$$\begin{aligned} a^{i,*} &=_{a^i} \bar{\bar{J}} = \frac{1}{n}\sum_{i=1}^{n}\Big(||\frac{1}{m+1}\sum_{t=t'-m}^{t'} f^i_{enc}(G_t) - D\ a^i||_2^2 \\ &+ \lambda||a^i||\Big) =_{a^i} \frac{1}{n}\sum_{i=1}^{n}\Big(||Z^i - D\ a^i||_2^2 \\ A^* &= [a^{1,*}, a^{2,*}, ..., a^{n,*}] \end{aligned} \tag{4.20}$$

To solve this l_1-regularized problem for each a^i, the derivative of $\bar{\bar{J}}$ is computed by the epsilon-l_1 norm technique:

$$\begin{aligned} \frac{\partial \bar{\bar{J}}}{\partial a^i} &= -2D^T(f^i_{enc}(G_t) - D\ a^i) + \lambda\ \Omega\ a^i \\ \Omega^{i,i} &= \sum_{j=1}^{K}((a^i(j))^2 + \epsilon)^{-\frac{1}{2}}\ \ with\ \epsilon \rightarrow 0 \end{aligned} \tag{4.21}$$

where $\Omega \in \mathbb{R}^{K\times K}$ is a diagonal matrix with diagonal elements $\Omega^{i,i}$ $(i = 1, 2, ..., K)$. The optimal $a^{i,*}$ $(i = 1, 2, ..., n)$ is computed by:

$$\frac{\partial \bar{\bar{J}}}{\partial a^i} = 0 \Rightarrow\ a^{i,*} = \left(D^T D + \frac{\lambda}{2}\Omega\right)^{-1} D^T f_{enc}(G_t)^i \tag{4.22}$$

4.6. Numerical Experiments

4.6.1. Dataset

The Pecan Street dataset [2] provided by the Dataport database contains 15-min behind-the-meter load and PV generation measurements of $n = 100$ homes in Texas in 2017 and 2018. 80% of the 2017 data are used to train the model while the rest is considered as the validation data and the 2018 data is used for testing.

4.6.2. Performance Metrics

4.6.2.1. Load and PV disaggregation metrics

For each home $1 \leq i \leq 100$, at each time $t = t'$, the DeepSTGDL model estimates the historical load $h_{t'}^{L,i}$ by $\hat{h}_{t'}^{L,i}$ and the historical PV power $h_{t'}^{PV,i}$ by $\hat{h}_{t'}^{PV,i}$. To show the performance of the proposed model, we compute the Root mean square (RMSE), Mean Absolute Error (MAE), as well as the Mean Absolute Percentage Error (MAPE) of $\hat{h}_{t'}^{L,i}$ by:

$$\begin{aligned} RMSE &= \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\frac{1}{m+1}\,||h_{t'}^{L,i} - \hat{h}_{t'}^{L,i}||_2^2)} \\ MAE &= \frac{1}{n}\sum_{i=1}^{n}\frac{1}{m+1}||h_{t'}^{L,i} - \hat{h}_{t'}^{L,i}||_1 \\ MAPE &= \left(\frac{100}{n}\sum_{i=1}^{n}\frac{1}{m+1}||\frac{h_{t'}^{L,i} - \hat{h}_{t'}^{L,i}}{h_{t'}^{L,i}}||_1\right)\% \end{aligned} \tag{4.23}$$

The RMSE, MAE, and MAPE of $\hat{h}_{t'}^{PV,i}$ is computed using similar formulations.

4.6.2.2. Load and PV prediction metrics

For each home i, at each time $t = t'$, DeepSGDL estimates the future load $L_{t'+k}^{i}$ by $\hat{L}_{t'+k}^{i}$ and the future PV power $PV_{t'+k}^{i}$ by $\hat{PV}_{t'+k}^{i}$. We compute the RMSE, MAE, as well as MAPE of these estimations w.r.t their true values using the same formulation in (4.23).

4.6.3. Experimental Settings

The model is implemented on a computer system with NVIDIA GTX 1080 Ti graphics card and Intel Core i7-7700K Quad-Core 4.2 GHz processor. The method is designed using Python 3 with Keras package using Tensorflow 1.13.1 backened, CUDA 10.0, and cuDNN 7.3 libraries.

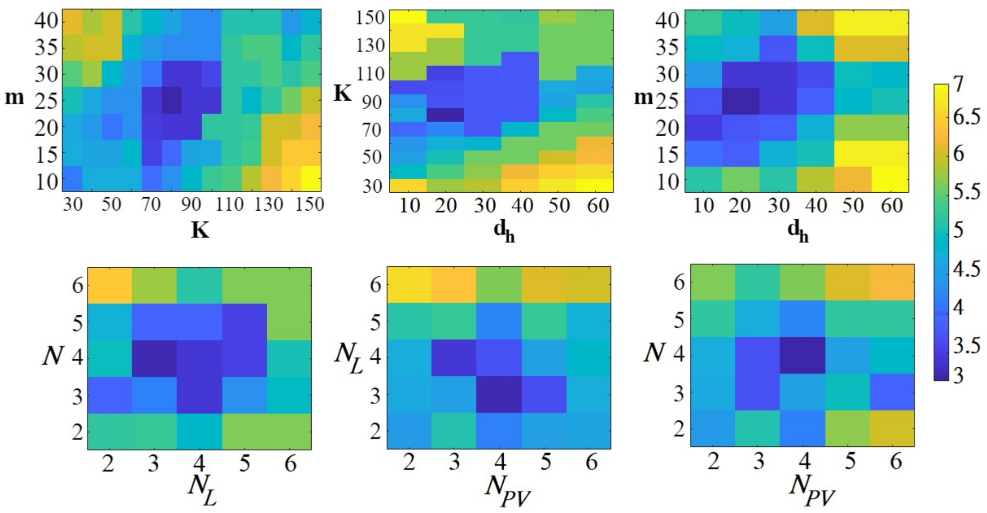


Figure 4.3. Average validation MAPE(%) for hourly load and PV prediction

4.6.4. Validation Resutls

To find the optimal values of hyperparameters, the model is evaluated on different configurations of the temporal window length $m \in [10, 40]$, number of dictionary atoms $K \in [30, 150]$, latent feature dimension $d_h \in [10, 60]$, error regularization coefficients $\lambda, \lambda_e, \lambda_n, \lambda_L, \lambda_{PV}, \lambda_L^{pred}, \lambda_{PV}^{pred} \in [0.1, 0.9]$, node decoder depth $\mathcal{N} \in [2, 6]$, load disaggregation depth $\mathcal{N}_L \in [2, 6]$, and PV disaggregation depth $\mathcal{N}_{PV} \in [2, 6]$. The configuration with the least validation MAPE computed in (4.23) is considered as the optimal settings. As shown in Fig. 4.3, the configuration with $m = 25$, $K = 80$, and $d_h = 20$ leads to the least MAPE for hourly predictions (i.e., $k = 4$). Larger parameters lead to the overfitting problem while smaller values decrease the generalization capacity of the model, hence increasing the validation error. Furthermore, the optimal configuration is shown to have $\mathcal{N} = 4, \mathcal{N}_L = 3$, and $\mathcal{N}_{PV} = 4$. Larger number of layers would damage the generalization accuracy while lower numbers lead to poor computational capability for the estimation of load and PV features. The optimal regularization coefficients are $\lambda = 0.40, \lambda_e = 0.25, \lambda_n = 0.25, \lambda_L = 0.30, \lambda_{PV} = 0.35, \lambda_L^{pred} = 0.30$ and $\lambda_{PV}^{pred} = 0.40$.

4.6.5. Test Results

4.6.5.1. Net Load Disaggregation Results

We compare DeepSTGDL with recent signal disaggregation benchmarks including the Seasonal Trend with Loess (STL) [229], K-Singular Value Decomposition (K-SVD) [155], Powerlet-based Energy Disaggregation (PED) [61], Nonlinear Dictionary Learning (NDL) [82], as well as the Temporal Dictionary Learning (TDL) [118]. Also, we consider the Classic Dictionary Learning (CDL) presented in Section III-A as a baseline to show the advantages of the presented approach in the disaggregation of net load. Table I contains the RMSE, MAE, as well as MAPE resutls of load and PV disaggregation for all disaggregation benchmarks. As shown in this table, the dictionary learning approachs such as CDL, PED, NDL, TDL, and STGDL provide lower error rates compared to the STL and K-SVD due to capturing more complex sparse patterns from the input net load data. While CDL and PED seek to find linear dictionaries in the original space of the net load, NDL captures dictionary atoms in the latent space of an autoencoder, hence leading to a better disaggregation accuracy. NDL provides 8.33% and 12.93% better MAPE compared to CDL for load and PV disaggregation tasks, repsectively. Moreover, it leads to 4.88% less MAPE for load disaggregation and 10.45% less PV disaggregation MAPE compared to the PED. The TDL applies a deep recurrent autoencoder instead of the classic feed-forward autoencoding approach in NDL, hence outperforming NDL by 1.34% MAPE in load estimation and 0.99% MAPE in PV power estimations.

As shown in Table I, our proposed STGDL method obtains the best load and PV disaggregation accuracy compared to all benchmarks. STGDL shows 19.37% better MAPE in load estimation, and 17.38% better MAPE in PV power calculations. The superiority of the proposed method compared to recent dictionary learning and signal decomposition approaches is due to understanding the spatial correlations of load and PV data measured in multiple locations while learning a highly nonlinear set of temporal net load patterns using a deep recurrent formulation. Fig. 4.4 depicts the net load disaggregation results of STGDL for house 6248 from March 31st to April 2nd, 2018. As shown in this figure, the proposed model can accurately estimate the historical load and PV energy.

The maximum error for load estimation is $0.6423\ kW$ and the maximum PV estimation error is $0.5030\ kW$.

Table 4.1. Net Load disaggregation Error Metrics

	Disaggregation Errors					
Methods	**Load**			**PV Power**		
	RMSE	**MAE**	**MAPE**	**RMSE**	**MAE**	**MAPE**
STL	0.5209	0.4187	12.2989	0.3797	0.2618	9.1076
K-SVD	0.4523	0.3508	10.0795	0.3218	0.2391	7.0904
CDL	0.3934	0.3310	9.0309	0.3076	0.2369	7.0673
PED	0.3156	0.2769	8.7032	0.2510	0.1838	6.8712
NDL	0.2393	0.2084	8.2786	0.1850	0.1369	6.1533
TDL	0.2184	0.1712	8.1677	0.1652	0.1323	6.0924
STGDL	**0.1352**	**0.0828**	**6.5853**	**0.0971**	**0.06321**	**5.0336**

4.6.5.2. Load and PV Prediction Results

To assess the prediciton performance of STGDL, we compare the load and PV prediction results with state-of-the-art prediction benchmarks such as the Convolutional Graph Autoencoder (CGAE) [112], Deep Residual Network (DRN) [33], Long Short-Term Memory Netowrk [98], Gated Recurrent Unit [232], and Support Vector Regression (SVR) [241].

Table II compares the RMSE, MAE, and MAPE results of STGDL with all benchmarks for 1-hr ahead load and PV prediction. As shown in this table, GRU leads to a better accuracy compared to the SVR due to taking into account the temporal structures of the load and PV data. Compared to GRU, the LSTM has 3.43% and 9.49% better MAPE for load and PV predictions, respectively. The better accuracy is due to the larger parameter space of LSTM which leads to higher generalization capacity compared to the GRU. As shown in Table II, the DRN shows a better performance compared to the LSTM with 3.96% and 4.36% less MAPE for load and PV signals, respectively. The CGAE contains a convolutional graph autoencoder that captures powerful spatiotemporal features from the load and PV measurements. In this study, CGAE outperforms DRN by 4.23% in load MAPE and 11.13% in PV MAPE due to its larger generalization capacity as well as better understanding of spatial relationships between the residential units. The proposed STGDL model outperforms all benchmarks with a significant 12.23% load prediction MAPE improvement and 18.75% PV prediction MAPE improvement over the state-of-the-art CGAE model. Fig. 4.5 shows the 1-hr ahead prediction results of the proposed method for house 6248 from June 10th to 12th, 2018. As shown in this plot, STGDL can effectively follow the actual patterns of future load and PV data with 1.6445 kW and 0.5763 kW maximum error for load and PV energy, respectively.

Fig. 4.6 shows the changes in load and PV prediction MAPE with respect to the changes in the horizon length. As shown in this diagram, the MAPE is generally increased for all benchmarks with the increase of horizon length. As the load and PV MAPEs of CGAE are respectively increased by 7.16% and 4.42% from 1-hr to 24-hr horizons, we observe a slight increase rate of 2.91% for load and 2.50% for PV predictions corresponding to the STGDL model. While SVR, GRU, LSTM, DRN, and CGAE show a large rate of MAPE increase after 6-hr load and 2-hr PV predictions, the proposed STGDL model shows a small increase rate which reflects the high accuracy and robustness of the model.

Table 4.2. Hourly Load and PV Prediction Error Metrics

	Prediction Errors					
Methods	**Load**			**PV Power**		
	RMSE	**MAE**	**MAPE**	**RMSE**	**MAE**	**MAPE**
SVR	0.5306	0.4722	11.0297	0.5106	0.3977	9.2851
GRU	0.4296	0.4310	10.2763	0.4476	0.3498	8.8203
LSTM	0.3872	0.3901	9.9240	0.3721	0.3105	8.2406
DRN	0.3508	0.3749	9.5311	0.3208	0.2519	7.8810
CGAE	0.2891	0.2914	9.1283	0.2614	0.1739	7.0042
STGDL	**0.1908**	**0.1348**	**8.0120**	**0.1471**	**0.0925**	**5.6912**

4.7. Conclusions

This chapter presents a novel spatiotemporal Behind-the-Meter Load and PV prediction problem that aims to predict the BTM load and PV generation of neighboring residential units. The

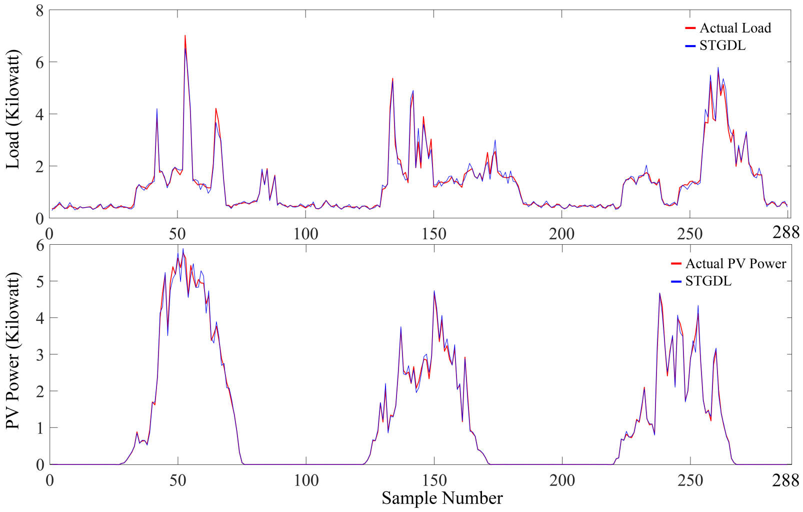


Figure 4.4. Net load disaggregation results of STGDL for house 6248 from March 31st to April 2nd, 2018.

problem is cast to a novel spatiotemporal graph dictionary learning problem where the units are modeled as a spatiotemporal graph. Each node represents the net load time series at each unit and the edges show the correlation between the corresponding units. A new spatiotemporal graph autoencoder model is designed to compute the spatiotemporal manifold of the net load measurements, and capture highly nonlinear features of the net load data. Moreover, a novel optimization is proposed to utilize the latent features of the autoencoder to learn a deep nonlinear dictionary of patterns from the net load data. The dictionary atoms are used to disaggregate the net load at each unit into the corresponding BTM load and PV time series. The estimated BTM values are mapped to the future values of BTM load and PV generations using a deep recurrent neural network. The proposed method is trained and evaluated on the Pecan Street dataset, as a real-world publically available load and PV dataset. Numerical results show the merit of the presented model compared to the state-of-the-art temporal and spatiotemporal models both in terms of disaggregation of the net load as well as prediction of future BTM load and PV.

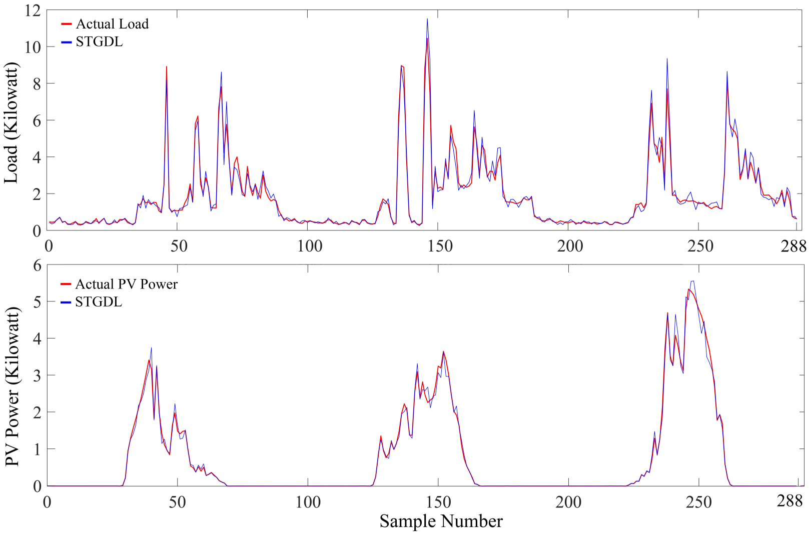


Figure 4.5. Hourly prediction results of the proposed method for house 6248 from June 10th to 12th, 2018.

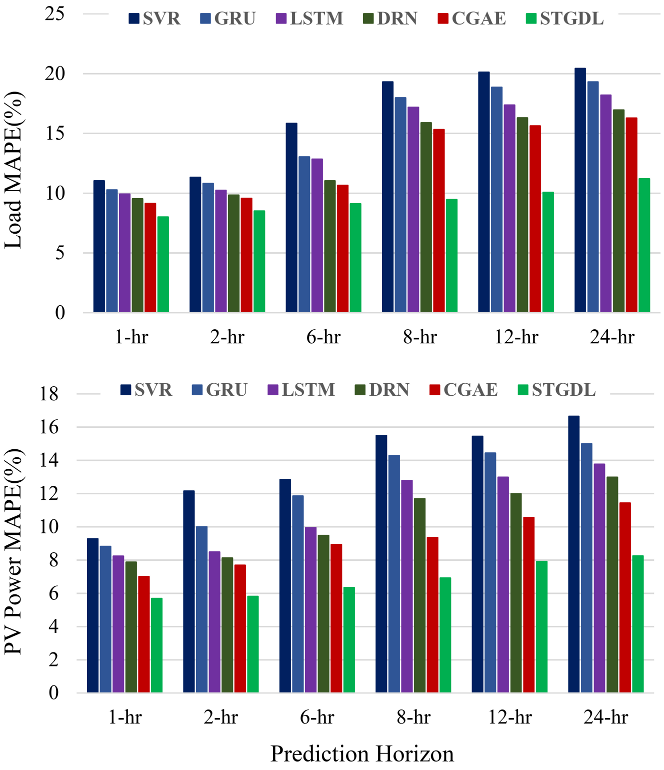


Figure 4.6. Prediction MAPE of the benchmarks for various time horizons

Chapter 5

Conclusions

In recent years, with the rapid growth in size and complexity of energy datasets, machine learning algorithms have gained increasing attention to recognize meaningful patterns and structures in the data. Data-driven algorithms provide accurate solutions to a large variety of classification, detection, prediction, and modeling problems in energy systems. In this domain, we introduce advanced deep learning frameworks as cutting-edge pattern recognition models for the prediction, modeling, and synthesis of power system measurements in real-world energy networks. Our research includes a novel graph convolutional network (GCN) for the spatiotemporal prediction of wind energy time series in large-scale wind sites. The wind sites are modeled as nodes of an undirected weighted graph where each edge reflects the correlation between the wind measurements at the corresponding sites. Interval neurons derived from the Rough set theory are incorporated into our graph convolution layers to address the uncertainties of wind time series while learning spatiotemporal wind patterns in a sparse GCN. In this category of studies, we developed an interval probability distribution learning (IPDL) model that incorporates the Rough set theory into restricted Boltzmann machines (RBMs) to capture the probabilistic patterns of wind data in an unsupervised fashion. Our IPDLs are stacked to create a novel deep belief network (DBN) that captures unsupervised interval knowledge from the wind data, hence, handling the uncertainties of the wind measurements. The computed deep probabilistic features are utilized by a Fuzzy Type II system to predict the day ahead wind energy of the wind sites in Colorado, United States. Our numerical comparisons show significant prediction accuracy improvement compared to very recent DBN and autoencoding approaches due to better generalization and robustness to measurement noise. We extended our energy prediction research to devise the convolutional graph autoencoder (CGAE), which is a deep generative framework to predict the solar irradiance in a large set of neighboring photovoltaic (PV) energy sites near Chicago, United States. CGAE is a new class of conditional variational autoencoders that consists of a GCN as an encoder to capture the spatiotemporal features of PV measurements at the solar sites as well as a rectified linear unit (ReLU) neural network as a decoder to map the latent spatiotemporal features computed by the encoder to the future values of PV energy. Data-driven time-varying load modeling using system-wide measurements is another major domain of our deep learning studies. We proposed a multimodal long short-term memory network (LSTM) that simultaneously extracts highly nonlinear temporal features from the load and voltage/power measurements of various buses in a power system. The computed patterns are utilized in a high-dimensional regression formulation to compute the parameters of several dynamic loads corresponding to the induction motors in a large-scale smart grid. Also, we present a deep generative version of this work, where we define a new conditional variational autoencoder with an encoding LSTM to capture the temporal load patterns, and a decoding ReLU neural network to decode LSTM's hidden features and estimate the load parameters in a probabilistic fashion. Our numerical results in terms of both deterministic and probabilistic error metrics show significantly better load identification performance compared to recent machine learning studies including extreme learning machines and gated recurrent units (GRUs). Another area of our research is the deep temporal dictionary learning (DTDL) for signal decomposition. In this domain, we devise a novel deep learning-based sparse coding optimization for energy disaggregation, i.e., decomposition of energy consumption signals of a residential customer into the appliances used. The idea is to learn a deep nonlinear dictionary of signature patterns corresponding to different appliances inside the latent space of an LSTM autoencoder. The autoencoder reconstructs the total consumption signal of the home, hence, capturing powerful time-dependent features of the observed energy. Our sparse optimization simultaneously learns a dictionary of signature patterns that can discriminate between different devices/appliances while finding the optimal parameters of LSTM to increase the discrimination accuracy. In this domain, we also presented a new deep learning-based optimization to decompose the observed net load signal of a set of homes into their unobserved behind-the-meter load and PV generation measurements. The problem is defined as a novel spatiotemporal dynamic graph dictionary learning problem, where a deep dictionary is captured for the spatiotemporal patterns of the graph corresponding to different houses in a wide area. Our optimization disaggregates the net load at each home into the load and PV generation inside the latent space of a new spatiotemporal graph autoencoder (ST-GAE). Our latest deep learning work is the deep graph probability density learning for power grid synthesis. Actual power network datasets are generally confidential; however, researchers need realistic datasets to improve power grid reliability and security. Therefore, in this study, we present a novel deep generative model to learn the joint probability density function (PDF) of nodes/edges of a power network. Each node represents a bus with power demand/supply data and each bus represents a line between the corresponding buses as well as its physical characteristics. The graph is encoded by a GRU that learns the sequence of observed nodes/edges in the network. The captured features are further used by ReLU neural networks to model the PDF of buses and lines. Sampling from the captured PDF, one can generate a large variety of realistic power grids that imitate the original power network.

Chapter 6

Publication List

[P1] **M. Khodayar** and J. Wang, ”Probabilistic Time-Varying Parameter Identification for Load Modeling: A Deep Generative Approach”, *IEEE Transactions on Industrial Informatics*, vol. 16, no.9, pp. 1-11, 2020. DOI: 10.1109/TII.2020.2971014.

[P2] **M. Khodayar**, J. Wang and Z. Wang, ”Energy Disaggregation via Deep Temporal Dictionary Learning”, *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 5, pp. 1-14, 2019. DOI: 10.1109/tnnls.2019.2921952.

[P3] M. Cui, **M. Khodayar**, C. Chen, X. Wang, Y. Zhang and M. Khodayar, ”Deep Learning-Based Time-Varying Parameter Identification for System-Wide Load Modeling”, *IEEE Transactions on Smart Grid*, vol. 10, no. 6, pp. 6102-6114, 2019. DOI: 10.1109/tsg.2019.2896493.

[P4] **M. Khodayar**, J. Wang and M. Manthouri, ”Interval Deep Generative Neural Network for Wind Speed Forecasting”, *IEEE Transactions on Smart Grid*, vol. 10, no. 4, pp. 3974-3989, 2018. DOI: 10.1109/tsg.2018.2847223.

[P5] **M. Khodayar** and J. Wang, ”Spatio-Temporal Graph Deep Neural Network for Short-Term Wind Speed Forecasting”, *IEEE Transactions on Sustainable Energy*, vol. 10, no. 2, pp. 670-681, 2018. DOI: 10.1109/tste.2018.2844102.

[P6] **M. Khodayar**, S. Mohammadi, M. Khodayar, J. Wang and G. Liu, ”Convolutional Graph Autoencoder: A Generative Deep Neural Network for Probabilistic Spatio-temporal Solar Irradiance Forecasting”, *IEEE Transactions on Sustainable Energy*, vo. 11, no.2, 2019. DOI:

10.1109/tste.2019.2897688.

[P7] **M. Khodayar**, Y. Zhang, J. Wang and Z. Wang, ”Deep Generative Graph Distribution Learning for Synthetic Power Grids”, *ArXiv: https://arxiv.org/abs/1901.09674.*, 2020.

[P8] **M. Khodayar**, G. Liu, J. Wang, O. Kaynak,and M. E. Khodayar ”Spatiotemporal Behind-the-Meter Load and PV Power Prediction via Deep Graph Dictionary Learning”, *IEEE Transactions on Neural Networks and Learning Systems*, Under Review.

[P9] **M. Khodayar**, G. Liu, J. Wang, and M. E. Khodayar ”Deep Learning in Power Systems Research: A Review”, *IEEE Transactions on Smart Grid*, Under Review.

BIBLIOGRAPHY

[1] IEEE benchmark systems. http://labs.ece.uw.edu/pstca/. Accessed: 2019-01-20.

[2] Pecan Street Dataset. http://https://www.pecanstreet.org/dataport/. Accessed: 2019-11-28. 110

[3] Polish grid. www.pserc.cornell.edu/matpower/. Accessed: 2019-01-21.

[4] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467* (2016). 53, 76

[5] Agoua, X. G., Girard, R., and Kariniotakis, G. Probabilistic model for spatio-temporal photovoltaic power forecasting. *IEEE Transactions on Sustainable Energy* (2018), 1–1. 62, 76

[6] Ahmadian, S., Malki, H., and Han, Z. Cyber attacks on smart energy grids using generative adverserial networks. In *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)* (2018), IEEE, pp. 942–946. 3, 11, 17

[7] Alessandrini, S., Delle Monache, L., Sperati, S., and Cervone, G. An analog ensemble for short-term probabilistic solar power forecast. *Applied energy 157* (2015), 95–110. 75

[8] Amjady, N., Keynia, F., and Zareipour, H. Short-term wind power forecasting using ridgelet neural network. *Electric Power Systems Research 81*, 12 (2011), 2099–2107. 20, 22, 24

[9] An, D., Yang, Q., Liu, W., and Zhang, Y. Defending against data integrity attacks in smart grid: A deep reinforcement learning-based approach. *IEEE Access 7* (2019), 110835–110845. 4, 15, 17

[10] Arif, A., Wang, Z., Wang, J., Mather, B., Bashualdo, H., and Zhao, D. Load modeling–a review. *IEEE Trans. Smart Grid* (2017). in press.

[11] Awad, Y. A., Koutrakis, P., Coull, B. A., and Schwartz, J. A spatio-temporal prediction model based on support vector machine regression: Ambient black carbon in three new england states. *Environmental research 159* (2017), 427–434. 62, 76

[12] AZAD, H. B., MEKHILEF, S., AND GANAPATHY, V. G. Long-term wind speed forecasting and general pattern recognition using neural networks. *IEEE Transactions on Sustainable Energy 5*, 2 (2014), 546–553. 21, 44, 48

[13] BAE, K. Y., JANG, H. S., AND SUNG, D. K. Hourly solar irradiance prediction based on support vector machine and its error analysis. *IEEE Transactions on Power Systems 32*, 2 (2017), 935–945. 58

[14] BARBOUNIS, T., AND THEOCHARIS, J. Locally recurrent neural networks for long-term wind speed and power prediction. *Neurocomputing 69*, 4-6 (2006), 466–496.

[15] BASTIAN, M., HEYMANN, S., JACOMY, M., ET AL. Gephi: an open source software for exploring and manipulating networks. *Icwsm 8*, 2009 (2009), 361–362. 76

[16] BAYINDIR, R., YESILBUDAK, M., COLAK, M., AND GENC, N. A novel application of naive bayes classifier in photovoltaic energy prediction. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)* (Dec 2017), pp. 523–527. 60

[17] BENGIO, Y., COURVILLE, A., AND VINCENT, P. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence 35*, 8 (2013), 1798–1828. 1, 2

[18] BHASKAR, K., AND SINGH, S. Awnn-assisted wind power forecasting using feed-forward neural network. *IEEE transactions on sustainable energy 3*, 2 (2012), 306–315.

[19] BIRCHFIELD, A. B., XU, T., GEGNER, K. M., SHETYE, K. S., AND OVERBYE, T. J. Grid structural characteristics as validation criteria for synthetic networks. *IEEE Transactions on power systems 32*, 4 (2017), 3258–3265.

[20] BIRCHFIELD, A. B., XU, T., AND OVERBYE, T. J. Power flow convergence and reactive power planning in the creation of large synthetic grids. *IEEE Transactions on Power Systems 33*, 6 (Nov 2018), 6667–6674.

[21] BOSCH, J. L., AND KLEISSL, J. Cloud motion vectors from a network of ground sensors in a solar power plant. *Solar Energy 95* (2013), 13–20. 58

[22] BRACALE, A., CARPINELLI, G., AND DE FALCO, P. A probabilistic competitive ensemble method for short-term photovoltaic power forecasting. *IEEE Transactions on Sustainable Energy 8*, 2 (2017), 551–560. 60

[23] BRACALE, A., CARPINELLI, G., DE FALCO, P., RIZZO, R., AND RUSSO, A. New advanced method and cost-based indices applied to probabilistic forecasting of photovoltaic generation. *Journal of Renewable and Sustainable Energy 8*, 2 (2016), 023505. 60

[24] BUHAN, S., AND ÇADIRCI, I. Multistage wind-electric power forecast by using a combination of advanced statistical methods. *IEEE Transactions on Industrial Informatics 11*, 5 (2015), 1231–1242. 19

[25] Cadenas, E., and Rivera, W. Short term wind speed forecasting in la venta, oaxaca, méxico, using artificial neural networks. *Renewable Energy 34*, 1 (2009), 274–278.

[26] Cai, L., Thornhill, N. F., Kuenzel, S., and Pal, B. C. Wide-area monitoring of power systems using principal component analysis and k-nearest neighbor analysis. *IEEE Transactions on Power Systems 33*, 5 (2018), 4913–4923. 1

[27] Cao, J., and Fan, Z. Deep learning-based online small signal stability assessment of power systems with renewable generation. In *2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)* (2018), IEEE, pp. 216–221. 2, 6, 7, 16

[28] Cao, Q., Ewing, B. T., and Thompson, M. A. Forecasting wind speed with recurrent neural networks. *European Journal of Operational Research 221*, 1 (2012), 148–154. 20, 21, 22, 24, 44

[29] Castro, L. M., Rodríguez-Rodríguez, J., and Martin-del Campo, C. Modelling of pv systems as distributed energy resources for steady-state power flow studies. *International Journal of Electrical Power & Energy Systems 115* (2020), 105505. 91

[30] Chan, S., Oktavianti, I., Puspita, V., and Nopphawan, P. Convolutional adversarial neural network (cann) for fault diagnosis within a power system: Addressing the challenge of event correlation for diagnosis by power disturbance monitoring equipment in a smart grid. In *2019 International Conference on Information and Communications Technology (ICOIACT)* (2019), IEEE, pp. 596–601. 3, 9, 16

[31] Chang, W.-Y., et al. A literature review of wind forecasting methods. *Journal of Power and Energy Engineering 2*, 04 (2014), 161. 19, 22

[32] Chen, J., Zhu, Q., Li, H., Zhu, L., Shi, D., Li, Y., Duan, X., and Liu, Y. Learning heterogeneous features jointly: A deep end-to-end framework for multi-step short-term wind power prediction. *IEEE Transactions on Sustainable Energy* (2019). 2, 7, 16

[33] Chen, K., Chen, K., Wang, Q., He, Z., Hu, J., and He, J. Short-term load forecasting with deep residual networks. *IEEE Transactions on Smart Grid* (2018). 114

[34] Chen, M.-R., Zeng, G.-Q., Lu, K.-D., and Weng, J. A two-layer nonlinear combination method for short-term wind speed prediction based on elm, enn, and lstm. *IEEE Internet of Things Journal 6*, 4 (2019), 6997–7010. 2, 16

[35] Chen, N., Qian, Z., Nabney, I. T., and Meng, X. Wind power forecasts using gaussian processes and numerical weather prediction. *IEEE Transactions on Power Systems 29*, 2 (2013), 656–665.

[36] Chen, S., Gooi, H., and Wang, M. Solar radiation forecast based on fuzzy logic and neural networks. *Renewable Energy 60* (2013), 195–201. 59

[37] CHEN, Y., WANG, Y., KIRSCHEN, D., AND ZHANG, B. Model-free renewable scenario generation using generative adversarial networks. *IEEE Transactions on Power Systems 33*, 3 (2018), 3265–3275. 3, 11, 17, 61

[38] CHOLLET, F., ET AL. Keras. https://github.com/fchollet/keras, 2015. 76

[39] CHU, Y., AND COIMBRA, C. F. Short-term probabilistic forecasts for direct normal irradiance. *Renewable Energy 101* (2017), 526–536. 59

[40] CIARAMELLA, A., STAIANO, A., CERVONE, G., AND ALESSANDRINI, S. A bayesian-based neural network model for solar photovoltaic power forecasting. In *International Workshop on Neural Networks* (2015), Springer, pp. 169–177. 60

[41] COCHRAN, J., DENHOLM, P., SPEER, B., AND MILLER, M. Grid integration and the carrying capacity of the us grid to incorporate variable renewable energy. Tech. rep., National Renewable Energy Lab.(NREL), Golden, CO (United States), 2015.

[42] CRISOSTO, C., HOFMANN, M., MUBARAK, R., AND SECKMEYER, G. One-hour prediction of the global solar irradiance from all-sky images using artificial neural networks. *Energies 11*, 11 (2018), 2906. 58, 62

[43] CROS, S., LIANDRAT, O., SÉBASTIEN, N., AND SCHMUTZ, N. Extracting cloud motion vectors from satellite images for solar power forecasting. In *Geoscience and Remote Sensing Symposium (IGARSS), 2014 IEEE International* (2014), IEEE, pp. 4123–4126. 58

[44] CUI, M., FENG, C., WANG, Z., AND ZHANG, J. Statistical representation of wind power ramps using a generalized Gaussian mixture model. *IEEE Trans. Sustain. Energy 9*, 1 (Jan. 2018), 261–272.

[45] CUI, M., KE, D., SUN, Y., GAN, D., ZHANG, J., AND HODGE, B.-M. Wind power ramp event forecasting using a stochastic scenario generation method. *IEEE Trans. Sustain. Energy 6*, 2 (Apr. 2015), 422–433.

[46] CUI, M., KHODAYAR, M., CHEN, C., WANG, X., ZHANG, Y., AND KHODAYAR, M. E. Deep learning-based time-varying parameter identification for system-wide load modeling. *IEEE Transactions on Smart Grid 10*, 6 (2019), 6102–6114. 2, 8, 16

[47] CUI, M., WANG, J., TAN, J., FLORITA, A., AND ZHANG, Y. A novel event detection method using PMU data with high precision. *IEEE Trans. Power Syst. 34*, 1 (Jan. 2019), 454–466.

[48] CUI, M., WANG, J., AND YUE, M. Machine learning based anomaly detection for load forecasting under cyberattacks. *IEEE Trans. Smart Grid* (2019). in press.

[49] CUI, M., WANG, Z., FENG, C., AND ZHANG, J. A truncated Gaussian mixture model for distributions of wind power ramping features. In *Proc. IEEE Power Energy Soc. Gen. Meeting* (Chicago, IL, USA, 2017), pp. 1–5.

[50] da Costa Lopes, F., Watanabe, E. H., and Rolim, L. G. B. A control-oriented model of a PEM fuel cell stack based on NARX and NOE neural networks. *IEEE Trans. Ind. Electron. 62*, 8 (2015), 5155–5163.

[51] Danziger, M. M., Shekhtman, L. M., Berezin, Y., and Havlin, S. Two distinct transitions in spatially embedded multiplex networks. *arXiv preprint arXiv:1505.01688* (2015).

[52] de Jesús, D. A. R., Mandal, P., Chakraborty, S., and Senjyu, T. Solar pv power prediction using a new approach based on hybrid deep neural network. In *2019 IEEE Power & Energy Society General Meeting (PESGM)* (2019), IEEE, pp. 1–5. 2, 8, 16

[53] De Kock, J., Van Der Merwe, F., and Vermeulen, H. Induction motor parameter estimation through an output error technique. *IEEE Trans. Energy Convers. 9*, 1 (1994), 69–76.

[54] Debnath, K. B., and Mourshed, M. Forecasting methods in energy planning models. *Renewable and Sustainable Energy Reviews 88* (2018), 297–325. 59, 60, 62

[55] Dejamkhooy, A., Ahmadpour, A., and Pourjafar, S. Non–intrusive appliance load disaggregation in smart homes using hybrid constrained particle swarm optimization and factorial hidden markov model. *Journal of Energy Management and Technology 3*, 4 (2019), 52–64. 93

[56] Doersch, C. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908* (2016). 72

[57] Doucoure, B., Agbossou, K., and Cardenas, A. Time series prediction using artificial wavelet neural network and multi-resolution analysis: Application to wind speed data. *Renewable Energy 92* (2016), 202–211. 19

[58] Duan, J., Shi, D., Diao, R., Li, H., Wang, Z., Zhang, B., Bian, D., and Yi, Z. Deep-reinforcement-learning-based autonomous voltage control for power grid operations. *IEEE Transactions on Power Systems* (2019). 4, 13, 14, 15, 17

[59] Duchesne, L., Karangelos, E., and Wehenkel, L. Machine learning of real-time power systems reliability management response. In *2017 IEEE Manchester PowerTech* (2017), IEEE, pp. 1–6. 2, 6, 16

[60] Duval, M., and DePabla, A. Interpretation of gas-in-oil analysis using new iec publication 60599 and iec tc 10 databases. *IEEE Electrical Insulation Magazine 17*, 2 (2001), 31–41. 7

[61] Elhamifar, E., and Sastry, S. Energy disaggregation via learning powerlets and sparse coding. In *Twenty-Ninth AAAI Conference on Artificial Intelligence* (2015). 94, 113

[62] Erdem, E., and Shi, J. Arma based approaches for forecasting the tuple of wind speed and direction. *Applied Energy 88*, 4 (2011), 1405–1414. 20

[63] ESEYE, A. T., ZHANG, J., ZHENG, D., MA, H., AND JINGFU, G. A double-stage hierarchical anfis model for short-term wind power prediction. In *2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA)(* (2017), IEEE, pp. 546–551. 20, 21, 22, 24

[64] FINAMORE, A. R., CALDERARO, V., GALDI, V., PICCOLO, A., CONIO, G., AND GRASSO, S. A day-ahead wind speed forecasting using data-mining model-a feed-forward nn algorithm. In *2015 International Conference on Renewable Energy Research and Applications (ICRERA)* (2015), IEEE, pp. 1230–1235. 20, 21, 22, 24, 44

[65] GAO, X., DENG, F., AND YUE, X. Data augmentation in fault diagnosis based on the wasserstein generative adversarial network with gradient penalty. *Neurocomputing* (2019). 11, 17

[66] GEGNER, K. M., BIRCHFIELD, A. B., XU, T., SHETYE, K. S., AND OVERBYE, T. J. A methodology for the creation of geographically realistic synthetic power flow models. In *Power and Energy Conference at Illinois (PECI), 2016 IEEE* (2016), pp. 1–6.

[67] GENSLER, A., HENZE, J., SICK, B., AND RAABE, N. Deep learning for solar power forecasting—an approach using autoencoder and lstm neural networks. In *2016 IEEE international conference on systems, man, and cybernetics (SMC)* (2016), IEEE, pp. 002858–002865. 2, 7, 8, 9, 16

[68] GILANIFAR, M., WANG, H., OZGUVEN, E. E., ZHOU, Y., AND ARGHANDEH, R. Bayesian spatiotemporal gaussian process for short-term load forecasting using combined transportation and electricity data. *ACM Transactions on Cyber-Physical Systems 4*, 1 (2019), 2. 93

[69] GIRVAN, M., AND NEWMAN, M. E. Community structure in social and biological networks. *Proceedings of the national academy of sciences 99*, 12 (2002), 7821–7826. 65

[70] GOLESTANEH, F., GOOI, H. B., AND PINSON, P. Generation and evaluation of space–time trajectories of photovoltaic power. *Applied energy 176* (2016), 80–91. 59

[71] GOLESTANEH, F., PINSON, P., AND GOOI, H. B. Very short-term nonparametric probabilistic forecasting of renewable energy generation—with application to solar energy. *IEEE Transactions on Power Systems 31*, 5 (2016), 3850–3863. 59

[72] GRIGG, C., WONG, P., ALBRECHT, P., ALLAN, R., BHAVARAJU, M., BILLINTON, R., CHEN, Q., FONG, C., HADDAD, S., KURUGANTY, S., ET AL. The ieee reliability test system-1996. a report prepared by the reliability test system task force of the application of probability methods subcommittee. *IEEE Transactions on power systems 14*, 3 (1999), 1010–1020. 6

[73] HAMMOND, D. K., VANDERGHEYNST, P., AND GRIBONVAL, R. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis 30*, 2 (2011), 129–150.

[74] HE, W., AND CHAI, Y. An empirical study on energy disaggregation via deep learning. In *2016 2nd International Conference on Artificial Intelligence and Industrial Engineering (AIIE 2016)* (2016/11), Atlantis Press. 2, 16

[75] HE, Y., DENG, J., AND LI, H. Short-term power load forecasting with deep belief network and copula models. In *2017 9th International conference on intelligent human-machine systems and cybernetics (IHMSC)* (2017), vol. 1, IEEE, pp. 191–194. 3, 8, 17

[76] HENRIET, S., ŞIMŞEKLI, U., DOS SANTOS, S., FUENTES, B., AND RICHARD, G. Independent-variation matrix factorization with application to energy disaggregation. *IEEE Signal Processing Letters 26*, 11 (2019), 1643–1647. 94

[77] HINTON, G. E., OSINDERO, S., AND TEH, Y.-W. A fast learning algorithm for deep belief nets. *Neural computation 18*, 7 (2006), 1527–1554. 22

[78] HISKENS, I. A. Nonlinear dynamic model evaluation from disturbance measurements. *IEEE Trans. Power Syst. 16*, 4 (2001), 702–710.

[79] HIYAMA, T., TOKIEDA, M., HUBBI, W., AND ANDOU, H. Artificial neural network based dynamic load modeling. *IEEE Trans. Power Syst. 12*, 4 (1997), 1576–1583.

[80] HONG, T., PINSON, P., FAN, S., ZAREIPOUR, H., TROCCOLI, A., AND HYNDMAN, R. J. Probabilistic energy forecasting: Global energy forecasting competition 2014 and beyond. *International Journal of Forecasting 32*, 3 (2016), 896–913. 60

[81] HOTTEL, H. C. A simple model for estimating the transmittance of direct solar radiation through clear atmospheres. *Solar energy 18*, 2 (1976), 129–134. 58

[82] HU, J., AND TAN, Y.-P. Nonlinear dictionary learning with application to image classification. *Pattern Recognition 75* (2018), 282–291. 113

[83] HU, Q., SU, P., YU, D., AND LIU, J. Pattern-based wind speed prediction based on generalized principal component analysis. *IEEE Transactions on Sustainable Energy 5*, 3 (2014), 866–874. 20, 21, 22, 24, 44

[84] HU, Q., ZHANG, S., YU, M., AND XIE, Z. Short-term wind speed or power forecasting with heteroscedastic support vector regression. *IEEE Transactions on Sustainable Energy 7*, 1 (2015), 241–249.

[85] HU, T., GUO, Q., LI, Z., SHEN, X., AND SUN, H. Distribution-free probability density forecast through deep neural networks. *IEEE transactions on neural networks and learning systems* (2019). 92

[86] HUANG, C.-J., AND KUO, P.-H. Multiple-input deep convolutional neural network model for short-term photovoltaic power forecasting. *IEEE Access 7* (2019), 74822–74834. 2

[87] HUANG, J., AND PERRY, M. A semi-empirical approach using gradient boosting and k-nearest neighbors regression for gefcom2014 probabilistic solar power forecasting. *International Journal of Forecasting 32*, 3 (2016), 1081–1086. 59

[88] HUANG, Q., HUANG, R., HAO, W., TAN, J., FAN, R., AND HUANG, Z. Adaptive power system emergency control using deep reinforcement learning. *IEEE Transactions on Smart Grid* (2019). 4, 14, 15, 17

[89] HUANG, X., HONG, S. H., YU, M., DING, Y., AND JIANG, J. Demand response management for industrial facilities: A deep reinforcement learning approach. *IEEE Access 7* (2019), 82194–82205. 4, 14, 17

[90] HUANG, Y., XU, Q., HU, C., SUN, Y., AND LIN, G. Probabilistic state estimation approach for ac/mtdc distribution system using deep belief network with non-gaussian uncertainties. *IEEE Sensors Journal 19*, 20 (2019), 9422–9430. 3, 10, 17

[91] HUNG, D. Q., MITHULANANTHAN, N., AND LEE, K. Y. Determining PV penetration for distribution systems with time-varying load models. *IEEE Trans. Power Syst. 29*, 6 (2014), 3048–3057.

[92] JANG, H. S., BAE, K. Y., PARK, H.-S., AND SUNG, D. K. Solar power prediction based on satellite images and support vector machine. *IEEE Trans. Sustain. Energy 7*, 3 (2016), 1255–1263. 58

[93] JIANG, H., ZHANG, Y., MULJADI, E., ZHANG, J. J., AND GAO, D. W. A short-term and high-resolution distribution system load forecasting approach using support vector regression with hybrid parameters optimization. *IEEE Transactions on Smart Grid 9*, 4 (2016), 3341–3350. 91

[94] JIANG, H., ZHANG, Y., MULJADI, E., ZHANG, J. J., AND GAO, D. W. A short-term and high-resolution distribution system load forecasting approach using support vector regression with hybrid parameters optimization. *IEEE Trans. Smart Grid 9*, 4 (2018), 3341–3350.

[95] JIANG, J., KONG, Q., PLUMBLEY, M., AND GILBERT, N. Deep learning based energy disaggregation and on/off detection of household appliances. *arXiv preprint arXiv:1908.00941* (2019). 2

[96] JIANG, Y., LONG, H., ZHANG, Z., AND SONG, Z. Day-ahead prediction of bihourly solar radiance with a markov switch approach. *IEEE Transactions on Sustainable Energy 8*, 4 (2017), 1536–1547. 58

[97] JIANG, Y., SONG, Z., AND KUSIAK, A. Very short-term wind speed forecasting with bayesian structural break model. *Renewable energy 50* (2013), 637–647. 20

[98] JIAO, R., ZHANG, T., JIANG, Y., AND HE, H. Short-term non-residential load forecasting based on multiple sequences lstm recurrent neural network. *IEEE Access 6* (2018), 59438–59448. 92, 114

[99] JIMENEZ, Y., CORTES, J., DUARTE, C., PETIT, J., AND CARRILLO, G. Non-intrusive discriminant analysis of loads based on power quality data. In *2019 IEEE Workshop on Power Electronics and Power Quality Applications (PEPQA)* (2019), IEEE, pp. 1–5. 1

[100] JONES, L., ZAVADIL, R., GRANT, W., ET AL. The future of wind forecasting and utility operations. *IEEE Power and Energy Magazine 3*, 6 (2005), 57–64.

[101] JUBAN, R., OHLSSON, H., MAASOUMY, M., POIRIER, L., AND KOLTER, J. Z. A multiple quantile regression approach to the wind, solar, and price tracks of gefcom2014. *International Journal of Forecasting 32*, 3 (2016), 1094–1102. 59

[102] JUNG, J., AND BROADWATER, R. P. Current status and future advances for wind speed and power forecasting. *Renewable and Sustainable Energy Reviews 31* (2014), 762–777.

[103] KARAKUŞ, O., KURUOĞLU, E. E., AND ALTINKAYA, M. A. One-day ahead wind speed/power prediction based on polynomial autoregressive model. *IET Renewable Power Generation 11*, 11 (2017), 1430–1439. 24, 34

[104] KARLSSON, D., AND HILL, D. J. Modelling and identification of nonlinear dynamic loads in power systems. *IEEE Trans. Power Syst. 9*, 1 (1994), 157–166.

[105] KAUR, D., LIE, T. T., NAIR, N. K., AND VALLÈS, B. Wind speed forecasting using hybrid wavelet transform-arma techniques. *Aims Energy 3*, 1 (2015), 13.

[106] KAWAGUCHI, K., KAELBLING, L. P., AND BENGIO, Y. Generalization in deep learning. *arXiv preprint arXiv:1710.05468* (2017).

[107] KEHE, W., YUE, Y., BOHAO, C., AND JINSHUI, W. Research of wind power prediction model based on rbf neural network. In *2013 International Conference on Computational and Information Sciences* (2013), IEEE, pp. 237–240. 20, 21, 22, 24

[108] KESHK, M., TURNBULL, B., MOUSTAFA, N., VATSALAN, D., AND CHOO, K.-K. R. A privacy-preserving framework based blockchain and deep learning for protecting smart power networks. *IEEE Transactions on Industrial Informatics* (2019). 2

[109] KEYHANI, A., LU, W., AND HEYDT, G. T. Composite neural network load models for power system stability analysis. In *Proc. IEEE PES Power Syst. Conf. and Expo.* (2004), pp. 1159–1163.

[110] KHODAYAR, M. Deep graph distribution learning synthetic dataset.

[111] KHODAYAR, M., KAYNAK, O., AND KHODAYAR, M. E. Rough deep neural architecture for short-term wind speed forecasting. *IEEE Transactions on Industrial Informatics 13*, 6 (2017), 2770–2779. 2, 7, 16, 21, 24, 44, 51, 58

[112] KHODAYAR, M., MOHAMMADI, S., KHODAYAR, M. E., WANG, J., AND LIU, G. Convolutional graph autoencoder: a generative deep neural network for probabilistic spatio-temporal solar irradiance forecasting. *IEEE Transactions on Sustainable Energy* (2019). 2, 3, 9, 16, 17, 114

[113] KHODAYAR, M., AND TESHNEHLAB, M. Robust deep neural network for wind speed prediction. In *2015 4th Iranian Joint Congress on Fuzzy and Intelligent Systems (CFIS)* (2015), IEEE, pp. 1–5. 58

[114] KHODAYAR, M., AND WANG, J. Spatio-temporal graph deep neural network for short-term wind speed forecasting. *IEEE Transactions on Sustainable Energy* (2018). 73

[115] KHODAYAR, M., WANG, J., AND MANTHOURI, M. Interval deep generative neural network for wind speed forecasting. *IEEE Transactions on Smart Grid 10*, 4 (2018), 3974–3989. 2, 10, 59

[116] KHODAYAR, M., WANG, J., AND WANG, Z. Energy disaggregation via deep temporal dictionary learning. *arXiv preprint arXiv:1809.03534* (2018). 62

[117] KHODAYAR, M., WANG, J., AND WANG, Z. Deep generative graph distribution learning for synthetic power grids. *arXiv preprint arXiv:1901.09674* (2019). 2, 12

[118] KHODAYAR, M., WANG, J., AND WANG, Z. Energy disaggregation via deep temporal dictionary learning. *IEEE transactions on neural networks and learning systems* (2019). 2, 16, 93, 94, 113

[119] KHODAYAR, M., ZHANG, Y., AND WANG, J. Deep generative graph distribution learning for synthetic power grids. 3, 11, 17

[120] KIANI, H. M., AND ZENG, X.-J. A function-on-function linear regression approach for short-term electric load forecasting. In *2019 IEEE Texas Power and Energy Conference (TPEC)* (2019), IEEE, pp. 1–5. 91

[121] KIM, D.-I., WANG, L., AND SHIN, Y.-J. Data driven method for event classification via regional segmentation of power systems. *IEEE Access 8* (2020), 48195–48204. 2, 6, 7, 9, 16

[122] KIPF, T. N., AND WELLING, M. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016). 73

[123] KNYAZKIN, V., CANIZARES, C. A., AND SODER, L. H. On the parameter estimation and modeling of aggregate power system loads. *IEEE Trans. Power Syst. 19*, 2 (2004), 1023–1031.

[124] KONG, W., DONG, Z. Y., HILL, D. J., LUO, F., AND XU, Y. Short-term residential load forecasting based on resident behaviour learning. *IEEE Transactions on Power Systems 33*, 1 (2017), 1087–1088. 92

[125] KONG, W., DONG, Z. Y., JIA, Y., HILL, D. J., XU, Y., AND ZHANG, Y. Short-term residential load forecasting based on lstm recurrent neural network. *IEEE Transactions on Smart Grid 10*, 1 (2017), 841–851. 92

[126] KRAMER, O., AND GIESEKE, F. Short-term wind energy forecasting using support vector regression. In *Soft Computing Models in Industrial and Environmental Applications, 6th International Conference SOCO 2011* (2011), Springer, pp. 271–280.

[127] KU, B.-Y., THOMAS, R. J., CHIOU, C.-Y., AND LIN, C.-J. Power system dynamic load modeling using artificial neural networks. *IEEE Trans. Power Syst. 9*, 4 (1994), 1868–1874.

[128] KUMAR, G. K., KAVATI, I., RAO, K. S., AND CHERUKU, R. Spatial co-location pattern mining using delaunay triangulation. In *Advances in Machine Learning and Data Science*. Springer, 2018, pp. 95–102.

[129] LAN, H., ZHANG, C., HONG, Y.-Y., HE, Y., AND WEN, S. Day-ahead spatiotemporal solar irradiation forecasting using frequency-based hybrid principal component analysis and neural network. *Applied Energy 247* (2019), 389–402. 93

[130] LARSON, D. P., NONNENMACHER, L., AND COIMBRA, C. F. Day-ahead forecasting of solar power output from photovoltaic plants in the american southwest. *Renewable Energy 91* (2016), 11–20. 58

[131] LAURET, P., DAVID, M., AND PEDRO, H. T. Probabilistic solar forecasting using quantile regression models. *Energies 10*, 10 (2017), 1591. 59, 62

[132] LAURET, P., VOYANT, C., SOUBDHAN, T., DAVID, M., AND POGGI, P. A benchmarking of machine learning techniques for solar radiation forecasting in an insular context. *Solar Energy 112* (2015), 446–457. 58, 62, 76

[133] LE CADRE, H., ARAVENA, I., AND PAPAVASILIOU, A. Solar pv power forecasting using extreme learning machine and information fusion. In *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning* (2015), pp. 1–6. 58, 62

[134] LEE, D., AND BALDICK, R. Short-term wind power ensemble prediction based on gaussian processes and neural networks. *IEEE Transactions on Smart Grid 5*, 1 (2013), 501–510. 20, 21, 22, 24, 44

[135] LEI, M., SHIYAN, L., CHUANWEN, J., HONGLING, L., AND YAN, Z. A review on the forecasting of wind speed and generated power. *Renewable and Sustainable Energy Reviews 13*, 4 (2009), 915–920. 19

[136] LI, B., ZHANG, J., HE, Y., AND WANG, Y. Short-term load-forecasting method based on wavelet decomposition with second-order gray neural network model combined with adf test. *IEEE Access 5* (2017), 16324–16331. 92

[137] LI, C., LI, Y., CAO, Y., MA, J., KUANG, Y., ZHANG, Z., LI, L., AND WEI, J. Credibility forecasting in short-term load forecasting and its application. *IET Gener. Transm. Distrib. 9*, 13 (2015), 1564–1571.

[138] LI, J., JOHN, R., COUPLAND, S., AND KENDALL, G. On nie-tan operator and type-reduction of interval type-2 fuzzy sets. *IEEE Transactions on Fuzzy Systems 26*, 2 (2017), 1036–1039. 38

[139] LI, L.-L., SUN, J., WANG, C.-H., ZHOU, Y.-T., AND LIN, K.-P. Enhanced gaussian process mixture model for short-term electric load forecasting. *Information Sciences 477* (2019), 386–398. 91, 92

[140] Li, S., Wang, H., Tian, Y., Shen, Y., and Aitouche, A. Wind speed forecasting based on fuzzy-neural network combination method. In *The 27th Chinese Control and Decision Conference (2015 CCDC)* (2015), IEEE, pp. 4811–4816. 24, 34

[141] Li, W., Deka, D., Chertkov, M., and Wang, M. Real-time faulted line localization and pmu placement in power systems through convolutional neural networks. *IEEE Transactions on Power Systems 34*, 6 (2019), 4640–4651. 2

[142] Li, X., Peng, M., He, H., and Liu, T. Dynamic load modeling for power system based on GD-FNN. In *Proc. 3rd Int. Conf. Digital Manuf. Autom.* (2012), pp. 339–342.

[143] Liao, H., Ding, S., Wang, M., and Ma, G. An overview on rough neural networks. *Neural Computing and Applications 27*, 7 (2016), 1805–1816. 22, 26

[144] Lin, K.-P., Pai, P.-F., and Ting, Y.-J. Deep belief networks with genetic algorithms in forecasting wind speed. *IEEE Access 7* (2019), 99244–99253. 2

[145] Liu, H., Tian, H.-q., Pan, D.-f., and Li, Y.-f. Forecasting models for wind speed using wavelet, wavelet packet, time series and artificial neural networks. *Applied Energy 107* (2013), 191–208. 20, 21, 22, 24, 25

[146] Liu, J., Qu, F., Hong, X., and Zhang, H. A small-sample wind turbine fault detection method with synthetic fault data using generative adversarial nets. *IEEE Transactions on Industrial Informatics 15*, 7 (2018), 3877–3888. 3, 11, 12, 17

[147] Liu, W., Liu, C., Lin, Y., Ma, L., Xiong, F., and Li, J. Ultra-short-term forecast of photovoltaic output power under fog and haze weather. *Energies 11*, 3 (2018), 528. 58

[148] Liu, Y., Qin, H., Zhang, Z., Pei, S., Jiang, Z., Feng, Z., and Zhou, J. Probabilistic spatiotemporal wind speed forecasting based on a variational bayesian deep learning model. *Applied Energy 260* (2020), 114259. 2

[149] Liu, Y., Qin, H., Zhang, Z., Pei, S., Wang, C., Yu, X., Jiang, Z., and Zhou, J. Ensemble spatiotemporal forecasting of solar irradiation using variational bayesian convolutional gate recurrent unit network. *Applied Energy 253* (2019), 113596. 93

[150] Liu, Z. Wind speed forecasting model based on fuzzy manifold support vector machine. *JOURNAL OF INFORMATION &COMPUTATIONAL SCIENCE 11*, 7 (2014), 2387–2395. 20, 21, 22, 24, 44

[151] Lu, X., Kuzmin, K., Chen, M., and Szymanski, B. K. Adaptive modularity maximization via edge weighting scheme. *Information Sciences 424* (2018), 55–68.

[152] Luo, P., Zhu, S., Han, L., and Chen, Q. Short-term photovoltaic generation forecasting based on similar day selection and extreme learning machine. In *Power & Energy Society General Meeting, 2017 IEEE* (2017), IEEE, pp. 1–5. 59

[153] Ma, J., He, R., and Hill, D. J. Load modeling by finding support vectors of load data from field measurements. *IEEE Trans. Power Syst. 21*, 2 (2006), 726–735.

[154] MA, X., JIN, Y., AND DONG, Q. A generalized dynamic fuzzy neural network based on singular spectrum analysis optimized by brain storm optimization for short-term wind speed forecasting. *Applied Soft Computing 54* (2017), 296–312. 24, 34, 44

[155] MAJUMDAR, A. Blind denoising autoencoder. *IEEE transactions on neural networks and learning systems 30*, 1 (2018), 312–317. 113

[156] MALDONADO, S., GONZÁLEZ, A., AND CRONE, S. Automatic time series analysis for electric load forecasting via support vector regression. *Applied Soft Computing 83* (2019), 105616. 91

[157] MARQUEZ, R., AND COIMBRA, C. F. Intra-hour dni forecasting based on cloud tracking image analysis. *Solar Energy 91* (2013), 327–336. 58

[158] MARRERO, L., GARCÍA-SANTANDER, L., CARRIZO, D., AND ULLOA, F. An application of load forecasting based on arima models and particle swarm optimization. In *2019 11th International Symposium on Advanced Topics in Electrical Engineering (ATEE)* (2019), IEEE, pp. 1–6. 91

[159] MARTIN, N., FRASCA, P., AND CANUDAS-DE WIT, C. A network reduction method inducing scale-free degree distribution. In *2018 European Control Conference (ECC)* (2018), IEEE, pp. 2236–2241.

[160] MATHE, J., MIOLANE, N., SEBASTIEN, N., AND LEQUEUX, J. Pvnet: A lrcn architecture for spatio-temporal photovoltaic powerforecasting from numerical weather prediction. *arXiv preprint arXiv:1902.01453* (2019). 3, 16

[161] MENG, H., BIANCHI-BERTHOUZE, N., DENG, Y., CHENG, J., AND COSMAS, J. P. Time-delay neural network for continuous emotional dimension prediction from facial expression sequences. *IEEE T. Cybern. 46*, 4 (2016), 916–929.

[162] MESTAV, K. R., LUENGO-ROZAS, J., AND TONG, L. Bayesian state estimation for unobservable distribution systems via deep learning. *IEEE Transactions on Power Systems 34*, 6 (2019), 4910–4920. 2

[163] MONFARED, M. A. S., JALILI, M., AND ALIPOUR, Z. Topology and vulnerability of the iranian power grid. *Physica A: Statistical Mechanics and its Applications 406* (2014), 24–33.

[164] MORAD, M., ABBAS, H. S., NAYEL, M., ELBASET, A. A., AND GALAL, A. Electrical energy consumption forecasting using gaussian process regression. In *2018 Twentieth International Middle East Power Systems Conference (MEPCON)* (2018), IEEE, pp. 292–297. 91, 92

[165] NASIR, Y. S., AND GUO, D. Multi-agent deep reinforcement learning for dynamic power allocation in wireless networks. *IEEE Journal on Selected Areas in Communications 37*, 10 (2019), 2239–2250. 4

[166] NOU, J., CHAUVIN, R., THIL, S., EYNARD, J., AND GRIEU, S. Clear-sky irradiance model for real-time sky imager application. *Energy Procedia 69* (2015), 1999–2008. 58

[167] OSÓRIO, G., MATIAS, J., AND CATALÃO, J. Short-term wind power forecasting using adaptive neuro-fuzzy inference system combined with evolutionary particle swarm optimization, wavelet transform and mutual information. *Renewable Energy 75* (2015), 301–307.

[168] OZKAN, M. B., AND KARAGOZ, P. A novel wind power forecast model: Statistical hybrid wind power forecast technique (shwip). *IEEE Transactions on Industrial Informatics 11*, 2 (2015), 375–387. 19

[169] PAGANI, G. A., AND AIELLO, M. The power grid as a complex network: a survey. *Physica A: Statistical Mechanics and its Applications 392*, 11 (2013), 2688–2700.

[170] PAN, Z., WANG, J., LIAO, W., CHEN, H., YUAN, D., ZHU, W., FANG, X., AND ZHU, Z. Data-driven ev load profiles generation using a variational auto-encoder. *Energies 12*, 5 (2019), 849. 4, 17

[171] PANDEY, S., AND KARYPIS, G. Structured dictionary learning for energy disaggregation. In *Proceedings of the Tenth ACM International Conference on Future Energy Systems* (2019), ACM, pp. 24–34. 93

[172] PAWLAK, Z. Rough set theory and its applications to data analysis. *Cybernetics & Systems 29*, 7 (1998), 661–688.

[173] PAWLAK, Z. *Rough sets: Theoretical aspects of reasoning about data*, vol. 9. Springer Science & Business Media, 2012. 22, 26

[174] PENG, G.-J. Joint and direct optimization for dictionary learning in convolutional sparse representation. *IEEE transactions on neural networks and learning systems* (2019). 94

[175] PEREIRA, J., AND SILVEIRA, M. Unsupervised anomaly detection in energy time series data using variational recurrent autoencoders with attention. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)* (2018), IEEE, pp. 1275–1282. 3, 17

[176] PFENNINGER, S., AND STAFFELL, I. Long-term patterns of european pv output using 30 years of validated hourly reanalysis and satellite data. *Energy 114* (2016), 1251–1265. 58

[177] PHILIPPOPOULOS, K., AND DELIGIORGI, D. Application of artificial neural networks for the spatial estimation of wind speed in a coastal region with complex topography. *Renewable Energy 38*, 1 (2012), 75–82. 20, 22

[178] PONOĆKO, J., AND MILANOVIĆ, J. V. Forecasting demand flexibility of aggregated residential load using smart meter data. *IEEE Transactions on Power Systems 33*, 5 (2018), 5446–5455. 92

[179] PULLEN, A., SAWYER, S., TESKE, S., AND JONES, J. Global wind energy outlook 2010. 19

[180] Qi, X., Luo, Y., Wu, G., Boriboonsomsin, K., and Barth, M. Deep reinforcement learning enabled self-learning control for energy efficient driving. *Transportation Research Part C: Emerging Technologies 99* (2019), 67–81. 4, 14, 17

[181] Qin, J., Han, X., Liu, G., Wang, S., Li, W., and Jiang, Z. Wind and storage cooperative scheduling strategy based on deep reinforcement learning algorithm. In *Journal of Physics: Conference Series* (2019), vol. 1213, IOP Publishing, p. 032002. 4, 14, 17

[182] Rafiei, M., Niknam, T., Aghaei, J., Shafie-Khah, M., and Catalão, J. P. Probabilistic load forecasting using an improved wavelet neural network trained by generalized extreme learning machine. *IEEE Transactions on Smart Grid 9*, 6 (2018), 6961–6971. 92

[183] Rouhani, A., and Abur, A. Real-time dynamic parameter estimation for an exponential dynamic load model. *IEEE Trans. Smart Grid 7*, 3 (2016), 1530–1536.

[184] Roy, Y., Banville, H., Albuquerque, I., Gramfort, A., Falk, T. H., and Faubert, J. Deep learning-based electroencephalography analysis: a systematic review. *Journal of neural engineering 16*, 5 (2019), 051001. 2

[185] Saber, A. Y., and Alam, A. R. Short term load forecasting using multiple linear regression for big data. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)* (2017), IEEE, pp. 1–6. 91

[186] Schmidhuber, J., and Hochreiter, S. Long short-term memory. *Neural Comput 9*, 8 (1997), 1735–1780.

[187] Sekhar, P., and Mishra, S. Storage free smart energy management for frequency control in a diesel-pv-fuel cell-based hybrid ac microgrid. *IEEE transactions on neural networks and learning systems 27*, 8 (2015), 1657–1671. 91

[188] Sengupta, M., Xie, Y., Lopez, A., Habte, A., Maclaurin, G., and Shelby, J. The national solar radiation data base (nsrdb). *Renewable and Sustainable Energy Reviews 89* (2018), 51–60. 62, 63

[189] Severoğlu, N., and Salor, Ö. Harmonic analysis in power systems using convolutional neural networks. In *2018 26th Signal Processing and Communications Applications Conference (SIU)* (2018), IEEE, pp. 1–4. 3, 9, 16

[190] Shepero, M., van der Meer, D., Munkhammar, J., and Widén, J. Residential probabilistic load forecasting: A method using gaussian process designed for electric load data. *Applied energy 218* (2018), 159–172. 91, 92

[191] Shi, J., Ding, Z., Lee, W.-J., Yang, Y., Liu, Y., and Zhang, M. Hybrid forecasting model for very-short term wind power forecasting based on grey relational analysis and wind speed distribution features. *IEEE Transactions on Smart Grid 5*, 1 (2013), 521–526. 20, 21, 22, 24, 44

[192] Shi, Z., Yao, W., Zeng, L., Wen, J., Fang, J., Ai, X., and Wen, J. Convolutional neural network-based power system transient stability assessment and instability mode prediction. *Applied Energy 263* (2020), 114586. 3, 9, 16

[193] Shukur, O. B., and Lee, M. H. Daily wind speed forecasting through hybrid kf-ann model based on arima. *Renewable Energy 76* (2015), 637–647.

[194] Siemens. PSSE 32.0.5 Program Operational Manual, 2010.

[195] Singh, A. K., Pal, B. C., et al. Report on the 68-bus, 16-machine, 5-area system. *IEEE PES Task Force on Benchmark Systems for Stability Controls. Ver 3* (2013). 6, 8

[196] Soltan, S., Loh, A., and Zussman, g. Columbia university synthetic power grid with geographical coordinates. Tech. rep., 1 2018.

[197] Soltan, S., Loh, A., and Zussman, G. A learning-based method for generating synthetic power grids. *IEEE Systems Journal 13*, 1 (2018), 625–634. 13

[198] Soltan, S., and Zussman, G. Generation of synthetic spatially embedded power grid networks. In *Power and Energy Society General Meeting (PESGM), 2016* (2016), IEEE, pp. 1–5.

[199] Soman, S. S., Zareipour, H., Malik, O., and Mandal, P. A review of wind power and wind speed forecasting methods with different time horizons. In *North American Power Symposium 2010* (2010), IEEE, pp. 1–8.

[200] Sun, M., Zhang, T., Wang, Y., Strbac, G., and Kang, C. Using bayesian deep learning to capture uncertainty for residential net load forecasting. *IEEE Transactions on Power Systems 35*, 1 (2019), 188–201. 2

[201] Sun, X., Ouyang, Z., and Yue, D. Short-term load forecasting based on multivariate linear regression. In *2017 IEEE Conference on Energy Internet and Energy System Integration (EI2)* (2017), IEEE, pp. 1–5. 91

[202] Sun, Z., Zhao, S., and Zhang, J. Short-term wind power forecasting on multiple scales using vmd decomposition, k-means clustering and lstm principal computing. *IEEE Access 7* (2019), 166917–166929. 2, 8, 16

[203] Szabó, Z. Information theoretical estimators toolbox. *The Journal of Machine Learning Research 15*, 1 (2014), 283–287. 76

[204] Tan, B., Yang, J., Tang, Y., Jiang, S., Xie, P., and Yuan, W. A deep imbalanced learning framework for transient stability assessment of power system. *IEEE Access 7* (2019), 81759–81769. 2, 7, 16

[205] Tan, M., Yuan, S., Li, S., Su, Y., Li, H., and He, F. Ultra-short-term industrial power demand forecasting using lstm based hybrid ensemble learning. *IEEE Transactions on Power Systems* (2019). 2, 7, 8, 16

[206] Tang, X., Dai, Y., Wang, T., and Chen, Y. Short-term power load forecasting based on multi-layer bidirectional recurrent neural network. *IET Generation, Transmission & Distribution 13*, 17 (2019), 3847–3854. 92

[207] Tascikaraoglu, A., and Sanandaji, B. M. Short-term residential electric load forecasting: A compressive spatio-temporal approach. *Energy and Buildings 111* (2016), 380–392.

[208] Tascikaraoglu, A., Sanandaji, B. M., Chicco, G., Cocina, V., Spertino, F., Erdinc, O., Paterakis, N. G., and Catalao, J. P. Compressive spatio-temporal forecasting of meteorological quantities and photovoltaic power. *IEEE Transactions on Sustainable Energy 7*, 3 (2016), 1295–1305. 62, 76

[209] Tastu, J., Pinson, P., and Madsen, H. Space-time scenarios of wind power generation produced using a gaussian copula with parametrized precision matrix. *Tech. Univ. Denmark, Tech. Rep.* (2013). 62, 76

[210] Tiwari, S., Sabzchgar, R., and Rasouli, M. Short term solar irradiance forecast using numerical weather prediction (nwp) with gradient boost regression. In *2018 9th IEEE International Symposium on Power Electronics for Distributed Generation Systems (PEDG)* (2018), IEEE, pp. 1–8. 60

[211] Van der Meer, D. W., Widén, J., and Munkhammar, J. Review on probabilistic forecasting of photovoltaic power production and electricity consumption. *Renewable and Sustainable Energy Reviews 81* (2018), 1484–1512. 59, 60, 61, 62, 75

[212] Vázquez-Canteli, J. R., and Nagy, Z. Reinforcement learning for demand response: A review of algorithms and modeling techniques. *Applied energy 235* (2019), 1072–1089. 4

[213] Venayagamoorthy, G. K., Sharma, R. K., Gautam, P. K., and Ahmadi, A. Dynamic energy management system for a smart microgrid. *IEEE transactions on neural networks and learning systems 27*, 8 (2016), 1643–1656. 91

[214] Voyant, C., Notton, G., Kalogirou, S., Nivet, M.-L., Paoli, C., Motte, F., and Fouilloy, A. Machine learning methods for solar radiation forecasting: A review. *Renewable Energy 105* (2017), 569–582. 58

[215] Wan, C., Lin, J., Song, Y., Xu, Z., and Yang, G. Probabilistic forecasting of photovoltaic generation: An efficient statistical approach. *IEEE Transactions on Power Systems 32*, 3 (2017), 2471–2472. 59

[216] Wan, C., Xu, Z., Pinson, P., Dong, Z. Y., and Wong, K. P. Probabilistic forecasting of wind power generation using extreme learning machine. *IEEE Transactions on Power Systems 29*, 3 (2013), 1033–1044.

[217] Wan, C., Xu, Z., Pinson, P., Dong, Z. Y., and Wong, K. P. Probabilistic forecasting of wind power generation using extreme learning machine. *IEEE Transactions on Power Systems 29*, 3 (2014), 1033–1044. 58, 62, 75

[218] WAN, C., ZHAO, J., SONG, Y., XU, Z., LIN, J., AND HU, Z. Photovoltaic and solar power forecasting for smart grid energy management. *CSEE Journal of Power and Energy Systems 1*, 4 (2015), 38–46. 58

[219] WANG, C., WANG, Z., WANG, J., AND ZHAO, D. Robust time-varying parameter identification for composite load modeling. *IEEE Trans. Smart Grid* (2017). in press.

[220] WANG, H., RUAN, J., WANG, G., ZHOU, B., LIU, Y., FU, X., AND PENG, J. Deep learning-based interval state estimation of ac smart grids against sparse cyber attacks. *IEEE Transactions on Industrial Informatics 14*, 11 (2018), 4766–4778. 2

[221] WANG, H., WANG, G., LI, G., PENG, J., AND LIU, Y. Deep belief network based deterministic and probabilistic wind speed forecasting approach. *Applied Energy 182* (2016), 80–93. 3, 12, 17

[222] WANG, J., AND HU, J. A robust combination approach for short-term wind speed forecasting and analysis–combination of the arima (autoregressive integrated moving average), elm (extreme learning machine), svm (support vector machine) and lssvm (least square svm) forecasts using a gpr (gaussian process regression) model. *Energy 93* (2015), 41–56.

[223] WANG, J., LI, P., RAN, R., CHE, Y., AND ZHOU, Y. A short-term photovoltaic power prediction model based on the gradient boost decision tree. *Applied Sciences 8*, 5 (2018), 2076–3417. 60

[224] WANG, J., ZHANG, F., LIU, F., AND MA, J. Hybrid forecasting model-based data mining and genetic algorithm-adaptive particle swarm optimisation: a case study of wind speed time series. *IET Renewable Power Generation 10*, 3 (2016), 287–298. 40, 44

[225] WANG, K., CHEN, J., AND SONG, Z. Fault detection based on variational autoencoders for complex nonlinear processes. In *2019 12th Asian Control Conference (ASCC)* (2019), IEEE, pp. 1352–1357. 17

[226] WANG, Z., AND WANG, J. Time-varying stochastic assessment of conservation voltage reduction based on load modeling. *IEEE Trans. Power Syst. 29*, 5 (2014), 2321–2328.

[227] WEI, F., WAN, Z., AND HE, H. Cyber-attack recovery strategy for smart grid based on deep reinforcement learning. *IEEE Transactions on Smart Grid* (2019). 4, 14, 15, 17

[228] WEI, X., SHEN, H., LI, Y., TANG, X., WANG, F., KLEINSTEUBER, M., AND MURPHEY, Y. L. Reconstructible nonlinear dimensionality reduction via joint dictionary learning. *IEEE transactions on neural networks and learning systems 30*, 1 (2018), 175–189. 94

[229] WEN, Q., GAO, J., SONG, X., SUN, L., XU, H., AND ZHU, S. Robuststl: a robust seasonal-trend decomposition algorithm for long time series. In *Proceedings of the AAAI Conference on Artificial Intelligence* (2019), vol. 33, pp. 5409–5416. 113

[230] WEN, S., WANG, Y., TANG, Y., XU, Y., LI, P., AND ZHAO, T. Real-time identification of power fluctuations based on lstm recurrent neural network: A case study on singapore power system. *IEEE Transactions on Industrial Informatics 15*, 9 (2019), 5266–5275. 2, 8, 16

[231] WEN, Y., SONG, M., AND WANG, J. A combined ar-knn model for short-term wind speed forecasting. In *2016 IEEE 55th Conference on Decision and Control (CDC)* (2016), IEEE, pp. 6342–6346. 20

[232] WU, W., LIAO, W., MIAO, J., AND DU, G. Using gated recurrent unit network to forecast short-term load considering impact of electricity price. *Energy Procedia 158* (2019), 3369–3374. 114

[233] XIE, G., CHEN, X., AND WENG, Y. An integrated gaussian process modeling framework for residential load prediction. *IEEE Transactions on Power Systems 33*, 6 (2018), 7238–7248. 92

[234] XIE, L., GU, Y., ZHU, X., AND GENTON, M. G. Short-term spatio-temporal wind power forecast in robust look-ahead power system dispatch. *IEEE Transactions on Smart Grid 5*, 1 (2013), 511–520.

[235] XU, F., TIAN, Y., WANG, Z., AND LI, J. One-day ahead forecast of pv output based on deep belief network and weather classification. In *2018 Chinese Automation Congress (CAC)*, IEEE, pp. 412–417. 3, 12, 17

[236] XU, H., SUN, H., NIKOVSKI, D., KITAMURA, S., MORI, K., AND HASHIMOTO, H. Deep reinforcement learning for joint bidding and pricing of load serving entity. *IEEE Transactions on Smart Grid 10*, 6 (2019), 6366–6375. 4, 17

[237] XU, T., BIRCHFIELD, A. B., GEGNER, K. M., SHETYE, K. S., AND OVERBYE, T. J. Application of large-scale synthetic power system models for energy economic studies. In *Proceedings of the 50th Hawaii International Conference on System Sciences* (2017).

[238] XU, Z., MO, W., WANG, Y., LUO, S., AND LIU, T. Transformer fault diagnosis based on deep brief sparse autoencoder. In *2019 Chinese Control Conference (CCC)* (2019), IEEE, pp. 7432–7435. 2, 16

[239] YAN, J., ZHANG, H., LIU, Y., HAN, S., LI, L., AND LU, Z. Forecasting the high penetration of wind power on multiple scales using multi-to-multi mapping. *IEEE Transactions on Power Systems 33*, 3 (2018), 3276–3284. 2, 7, 8, 9, 16

[240] YANG, W., ZHU, Y., AND LIU, Y. Fast assessment of short-term voltage stability of ac/dc power grid based on cnn. In *2019 IEEE PES Asia-Pacific Power and Energy Engineering Conference (APPEEC)* (2019), IEEE, pp. 1–4. 3, 9, 16

[241] YANG, Y., CHE, J., DENG, C., AND LI, L. Sequential grid approach based support vector regression for short-term electric load forecasting. *Applied energy 238* (2019), 1010–1021. 114

[242] YANG, Y., HAO, J., ZHENG, Y., HAO, X., AND FU, B. Large-scale home energy management using entropy-based collective multiagent reinforcement learning framework. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems* (2019), International Foundation for Autonomous Agents and Multiagent Systems, pp. 2285–2287. 4, 17

[243] YE, Y., PAPADASKALOPOULOS, D., AND STRBAC, G. Factoring flexible demand non-convexities in electricity markets. *IEEE Transactions on Power Systems 30*, 4 (2014), 2090–2099.

[244] YE, Y., QIU, D., SUN, M., PAPADASKALOPOULOS, D., AND STRBAC, G. Deep reinforcement learning for strategic bidding in electricity markets. *IEEE Transactions on Smart Grid* (2019). 4, 14, 15, 17

[245] YU, Z., NIU, Z., TANG, W., AND WU, Q. Deep learning for daily peak load forecasting–a novel gated recurrent neural network combining dynamic time warping. *IEEE Access 7* (2019), 17184–17194. 92

[246] ZANG, H., CHENG, L., DING, T., CHEUNG, K. W., LIANG, Z., WEI, Z., AND SUN, G. Hybrid method for short-term photovoltaic power forecasting based on deep convolutional neural network. *IET Generation, Transmission & Distribution 12*, 20 (2018), 4557–4567. 2

[247] ZHANG, C., ZHOU, J., LI, C., FU, W., AND PENG, T. A compound structure of elm based on feature selection and parameter optimization using hybrid backtracking search algorithm for wind speed forecasting. *Energy Conversion and Management 143* (2017), 360–376. 44, 46

[248] ZHANG, C.-Y., CHEN, C. P., GAN, M., AND CHEN, L. Predictive deep boltzmann machine for multiperiod wind speed forecasting. *IEEE Transactions on Sustainable Energy 6*, 4 (2015), 1416–1425. 21, 23, 34, 36, 43, 44, 48, 51

[249] ZHANG, J., WANG, Y., SUN, M., ZHANG, N., AND KANG, C. Constructing probabilistic load forecast from multiple point forecasts: A bootstrap based approach. In *2018 IEEE Innovative Smart Grid Technologies-Asia (ISGT Asia)* (2018), IEEE, pp. 184–189. 62

[250] ZHANG, K., ZHU, H., AND GUO, S. Dependency analysis and improved parameter estimation for dynamic composite load modeling. *IEEE Trans. Power Syst. 32*, 4 (2017), 3287–3297.

[251] ZHANG, S., WANG, Y., LIU, M., AND BAO, Z. Data-based line trip fault prediction in power systems using lstm networks and svm. *IEEE Access 6* (2017), 7675–7686. 2, 8, 16

[252] ZHANG, S., ZHANG, S., LI, S., DU, L., AND HABETLER, T. G. Visualization of multi-objective switched reluctance machine optimization at multiple operating conditions with t-sne. In *2019 IEEE Energy Conversion Congress and Exposition (ECCE)* (2019), IEEE, pp. 3793–3798. 1

[253] Zhang, W., Quan, H., Gandhi, O., Rodríguez-Gallegos, C. D., Sharma, A., and Srinivasan, D. An ensemble machine learning based approach for constructing probabilistic pv generation forecasting. In *Asia-Pacific Power and Energy Engineering Conference (APPEEC), 2017 IEEE PES* (2017), IEEE, pp. 1–6. 60

[254] Zhang, W., Wang, J., Wang, J., Zhao, Z., and Tian, M. Short-term wind speed forecasting based on a hybrid model. *Applied Soft Computing 13*, 7 (2013), 3225–3233. 20, 21, 22, 24

[255] Zhang, Y., and Wang, J. Gefcom2014 probabilistic solar power forecasting based on k-nearest neighbor and kernel density estimator. In *Power & Energy Society General Meeting, 2015 IEEE* (2015), IEEE, pp. 1–5. 59, 62, 75

[256] Zhang, Z., Jiang, W., Qin, J., Zhang, L., Li, F., Zhang, M., and Yan, S. Jointly learning structured analysis discriminative dictionary and analysis multiclass classifier. *IEEE transactions on neural networks and learning systems 29*, 8 (2017), 3798–3814. 94

[257] Zhao, J., Netto, M., and Mili, L. A robust iterated extended kalman filter for power system dynamic state estimation. *IEEE Trans. Power Syst. 32*, 4 (2017), 3205–3216.

[258] Zhao, J., Wang, Z., and Wang, J. Robust time-varying load modeling for conservation voltage reduction assessment. *IEEE Trans. Smart Grid 9*, 4 (2018), 3304–3312.

[259] Zhao, R., Wang, D., Yan, R., Mao, K., Shen, F., and Wang, J. Machine health monitoring using local feature-based gated recurrent unit networks. *IEEE Trans. Ind. Electron. 65*, 2 (2018), 1539–1548.

[260] Zhao, X., Wang, S., and Li, T. Review of evaluation criteria and main methods of wind power forecasting. *Energy Procedia 12* (2011), 761–769. 19

[261] Zheng, L., Hu, W., Zhou, Y., Min, Y., Xu, X., Wang, C., and Yu, R. Deep belief network based nonlinear representation learning for transient stability assessment. In *2017 IEEE Power & Energy Society General Meeting* (2017), IEEE, pp. 1–5. 3, 10, 12, 17

[262] Zheng, R., and Gu, J. Anomaly detection for power system forecasting under data corruption based on variational auto-encoder. 3, 17

[263] Zhou, B., Ma, X., Luo, Y., and Yang, D. Wind power prediction based on lstm networks and nonparametric kernel density estimation. *IEEE Access 7* (2019), 165279–165292. 2, 8

[264] Zhou, H., Zhang, Y., Yang, L., Liu, Q., Yan, K., and Du, Y. Short-term photovoltaic power forecasting based on long short term memory neural network and attention mechanism. *IEEE Access 7* (2019), 78063–78074. 2, 8, 16