Article:
Hu, T, Li, K, Ma, H et al. (2021) Quantile forecast of renewable energy generation based on Indicator Gradient Descent and deep residual BiLSTM. Control Engineering Practice, 114. 104863. ISSN: 0967-0661

https://doi.org/10.1016/j.conengprac.2021.104863

Quantile Forecast of Renewable Energy Generation Based on Indicator Gradient Descent and Deep Residual BiLSTM

Tianyu Hu[a], Kang Li[b], Huimin Ma[a], Hongbin Sun[c], Kailong Liu*[d]

[a]*School of Computer and Communication Engineering, University of Science and Technology Beijing, China*
[b]*School of Electronic and Electrical Engineering, University of Leeds, UK*
[c]*Department of Electrical Engineering, Tsinghua University, China*
[d]*WMG, University of Warwick, UK*
** corresponding author (kliu02@qub.ac.uk; Kailong.Liu@warwick.ac.uk)*

Abstract

Accurate generation forecasting can effectively accelerate the use of renewable energy in hybrid energy systems, contributing significantly to the delivery of the net-zero emission target. Recently, neural-network-based quantile forecast models have shown superior performance on renewable energy generation forecasting, partially because they have subtly embedded quantile forecast *evaluation metrics* into their loss functions. However, the non-differentiability of involved metrics has rendered their metric-embedded loss functions not everywhere-derivable, resulting in inapplicability of gradient-based training approaches. Instead, they have resorted to heuristic searches for Neural Network (NN) training, bringing low training efficiency and a rigid restriction on the size of the resultant NN. In this paper, the Indicator Gradient Descent (IGD) is proposed to overcome the non-differentiability of involved metrics, and several metric-embedded loss functions are innovatively customized combining IGD, enabling NNs to be trained efficiently in a 'gradient-descent-like' manner. Moreover, the deep Bidirectional Long Short-Term Memory (BiLSTM) is adopted to capture the periodicity of renewable generation (diurnal and seasonal patterns), and the residual technique is used to improve the training efficiency of the deep BiLSTM. Finally, a Deep Quantile Forecast Network (DQFN) based on IGD and deep residual BiLSTM is developed for wind and solar power quantile forecasting. Practical experiments in four cases have verified the effectiveness and efficiency of DQFN and IGD, where DQFN has achieved the lowest average proportion deviations (all below 1.7%) and the highest skill scores.

Keywords: Energy generation, Quantile forecasting, Renewable energy, Deep learning, Indicator gradient decent, BiLSTM

Abbreviations

NN Neural Network

IGD Indicator Gradient Descent

BiLSTM Bidirectional Long Short-Term Memory

DQFN Deep Quantile Forecast Network

MILNN Metrics-In-Loss NN

PI Prediction Interval

MO Metric-Oriented

NMO Non-Metric-Oriented

CWC Coverage Width-based Criterion

ELM Extreme Learning Machine

APD Average Proportion Deviation

SS Skill Score

GD Gradient Descent

PSO Particle Swarm Optimization

GPU Graphic Process Unit

IG Indicator Gradient

CDF Cumulative Distribution Function

LSTM Long Short-Term Memory

SMP Strictly Monotone Premises

PDPF Piecewisely Derivable Piecewise Function

DLQR Dynamic Linear Quantile Regression

DeepAR Deep Auto-Regressive recurrent network

MOrdReD Memory-based Ordinal Regression Deep neural network

BLSTM Bayesian LSTM

ARX Auto-Regressive including eXogenous covariate

GARCH Generalized AR Conditional Heteroskedasticity

DQR Direct Quantile Regression

Nomenclature

t Timestamp

T The total number of timestamps

$\mathbf{x}_t$ Explanatory information (covariates)

y_t The forecast target

$F_t(\cdot)$ The CDF of y_t

$\hat{y}_t$ The forecast of y_t

D The lag interval

$\mathbf{X}_t$	A series of $\mathbf{x}_t$ with lag interval D
h	The lead time
α	The nominal proportion
q_t^α	The quantile of y_t with nominal proportion α
$P(\cdot)$	The probability of an event
$\hat{q}_t^\alpha$	The forecast of q_t^α
$\hat{\mathbf{q}}_t$	All the forecasted quantiles for timestamp t
r	The total number of nominal proportions
$\hat{\mathbf{Q}}$	All the forecasted quantiles
$\mathcal{E}(\cdot)$	An evaluation function of quantiles
$\mathbf{y}$	All the observations of forecast target
H	The dimension of hidden state in LSTM
L	The number of modules in DQFN
$\mathbb{D}_{\text{SMP}}$	The feasible region of $\hat{\mathbf{Q}}$ under SMP
APD^α	The APD corresponding to α
$\varepsilon(\cdot)$	The indicator function
$S_t^{\alpha_i}$	An intermediate variable for obtaining SS
$\mathcal{L}(\cdot)$	A metric-embedded loss function
M	The number of metrics involved
Λ	IG
∇	Gradient
λ_*	Step size during iterative update
$\mathcal{L}_{\text{APD}}$	Customized loss from APD
$\mathcal{L}_{\text{SS}}$	Customized loss from SS
$\mathcal{L}_{\text{SMP}}$	The penalty loss from SMP
$\boldsymbol{\theta}$	All the trainable parameters of an MILNN
$\overline{\lvert\text{APD}\rvert}$	The average absolute APD over all α
ξ	The upper bound of $\overline{\lvert\text{APD}\rvert}$
β_*	Lagrangian multipliers
$\mathcal{L}_g(\cdot)$	Lagrangian function

1. Introduction

Uncertainty exists in almost all aspects of power systems, especially due to significant penetration of renewable energy. As a crucial component in the operation and control of hybrid energy systems, to develop advanced *generation forecast* models (e.g., for the forecasting of wind power, wind speed, solar power, and solar radiation, etc.) is one of the most efficient ways to mitigate the uncertainty of renewable energy, accelerate the use of renewable energy in the whole energy chain (Yan et al., 2016), support renewable power integration in manufacturing and power systems, and contribute to the delivery of the net-zero emission target (Yang et al., 2020; Xue et al., 2016). Existing forecast models can be categorized into two groups: deterministic forecast and probabilistic forecast. Deterministic approaches (i.e., point forecast) produce forecast results in the form of expectations (conditioned on the explanatory information) (Hu et al., 2020b; Liu et al., 2020; Liu et al., 2020). While for probabilistic forecast models, information on the distribution of forecasting targets can be obtained, such as Prediction Intervals (PIs) (Wan et al., 2013b; Khosravi et al., 2013; Wan et al., 2013a; Khosravi and Nahavandi, 2013; Yan et al., 2019), quantiles (Wan et al., 2017; Golestaneh et al., 2016; Sideratos and Hatziargyriou, 2012), and full distributions (Wan et al., 2013c; Wang et al., 2017; He et al., 2017; Hu et al., 2020a), offering greater flexibility in the design of cost-effective decision-making schemes.

Among the aforementioned probabilistic forecast models, much attention has been paid recently to NN-based quantile (and PI) forecast due to their high non-linearity and universal approximation ability. Depending on whether or not the quantile (or PI) forecast evaluation *metrics* are embedded into NN's loss functions or not, NN-based probabilistic forecast models can be further categorized into two groups: Metric-Oriented (MO) models and Non-Metric-Oriented (NMO) models. For NMO models, the generation and evaluation of quantiles or PIs are completely *separated*, i.e., the generation of quantiles or PIs has no association with the quality of generated quantiles or PIs (their quality is evaluated through metrics). Examples of NMO models include the delta technique (Khosravi et al., 2010a), Bayesian (Zhang and Luh, 2005), bootstrap (Wan et al., 2013c), and mean-variance estimation (Nix and Weigend, 1994). The quantiles (or PIs) of these models are all generated under assumptions that either NN parameters or forecasting targets follow particular known distribution families (e.g., normal (Wan et al., 2013c; Wang et al., 2017), t-distribution (Khosravi et al., 2010a)). However, as the actual distributions of NN parameters or forecast targets are often unknown, these distribution assumptions may result in inevitable mismatches between *models* and *practice*, rendering them problematic to apply. While for MO models, the objective of NN training is to make the final evaluation metrics (i.e., performances) more satisfactory. E.g., Khosravi et al. (2010a) adopted the delta technique to generate PIs, then the PI-error-based cost function, a combination of several metrics, was used as the loss function for NN training. Khosravi et al. (2014) adopted bootstrap method to construct PIs, and the NN was trained using Coverage Width-based Criterion (CWC, a quantile forecast evaluation metric) as the loss function. In Khosravi and Nahavandi (2013), Khosravi et al. (2010b), and Quan et al. (2013), PIs were directly generated by NNs with CWC as loss functions. In Wan et al. (2013b), Wan et al. (2013a), Wan et al. (2017), and Golestaneh et al. (2016), PIs or quantiles were directly generated by Extreme Learning Machines (ELMs) with a combination of metrics as their loss functions, which had considered both the reliability and sharpness of resultant quantiles (or PIs). Since evaluation metrics are embedded into MO models' loss functions, they can thus directly pursue the best quality of resultant quantiles or PIs (Wan et al., 2013b).

Nevertheless, the computational efficiency and scalability of the above MO models still need improvement. Frequently used quantile forecast evaluation metrics include the following two: (a) Average Proportion Deviation (APD), assessing the reliability of the model; (b) Skill Score (SS), assessing both the reliability and sharpness comprehensively. Though rich applications of APD and SS (especially in the renewable energy generation forecast field) have demonstrated their effectiveness in

model assessment, both of them are not everywhere-derivable due to the indicator function in their definitions. Consequently, the loss functions of the MO models in which APD or SS is embedded are not everywhere-derivable, either. The non-differentiability of these metric-embedded loss functions has made Gradient Descent (GD) and back propagation inapplicable to the training of NNs in these MO models. Instead, they have resorted to heuristic searches, e.g., Particle Swarm Optimization (PSO) (Wan et al., 2013b,a; Golestaneh et al., 2016; Sideratos and Hatziargyriou, 2012; Quan et al., 2013), simulated annealing (Khosravi et al., 2013; Khosravi and Nahavandi, 2013; Khosravi et al., 2010a,b), genetic algorithm (Wan et al., 2016), evolutionary algorithm (Khosravi et al., 2014), and bat algorithm (Kavousi-Fard et al., 2015), etc. Though heuristic searches have made the NN training implementable, their computational efficiency is very sensitive to the problem size, i.e., as the size of search space grows exponentially with the number of trainable parameters, solving large optimization problems using heuristic searches will become dramatically time-consuming. Indeed, the intolerance of heuristic searches to the problem size has directly constrained the widths and depths of candidate NNs, resulting in limited approximation ability, scalability, and generalizability. Moreover, the performances of heuristic searches can be significantly affected by their hyper-parameters, e.g., the probability of crossover and mutation in genetic algorithm, inertia weight and velocity limits in PSO, and the control of added noise, etc. Since the adjustment of these hyper-parameters relies on a case-by-case basis and no effective strategy has been proposed to solve this problem yet, a significant mismatch might exist between case studies and practice. Wan et al. (2017) has converted the optimization of metrics into a linear optimization problem tactfully, which can effectively avoid the shortcomings of heuristic searches. Even though this conversion is effective, it was only applicable for ELMs with limited model capacities and approximation ability (compared to NNs) due to their relatively shallow structures and the random assignment of their parameters.

The low computational efficiency and scalability of heuristic searches have motivated the development of a more efficient training approach for MO models in this study. As MO models are essentially NNs, an intuitive idea is to use the off-the-shelf GD-based optimizers, e.g., momentum, Adam, Nadam, etc. These GD-based optimizers possess high compatibility with Graphic Process Units (GPUs), facilitating their broad application in the deep learning domain (Bengio et al., 2017). Nevertheless, as mentioned above, the non-differentiability of involved metrics has made their gradient calculation problematic, ruling out all the off-the-shelf GD-based optimizers. To overcome the non-differentiability of involved metrics, in this work, a novel training approach is proposed to train MO quantile forecast models, i.e., the IGD. In IGD, two steps are taken for optimizing the APD and SS in a "GD-like" manner: 1) the proposal and utilization of Indicator Gradient (IG), aiming to eliminate the presence of impulse functions when differentiating the indicator function (which is contained in APD, SS, and NN's loss function); 2) two loss functions for APD and SS have been specially customized after fully considering the properties of IGD, ensuring that each iteration in IGD always leads to more satisfactory APD and SS. Moreover, IG follows the same chain rule with gradient, making it compatible with back propagation and thus applicable to all kinds of NNs.

Another noticeable trend in the renewable energy generation forecast field is the increasing adoption of deep learning models. Deep learning has attracted a great deal of interest in academia recently (Bengio et al., 2017), and it has also been applied to probabilistic forecast: in Wang et al. (2017), convolutional neural networks and the bootstrap method were combined for wind power PI forecast. In Flunkert et al. (2017), a Deep Auto-Regressive recurrent network (DeepAR) under normal distribution is proposed for the probabilistic forecast. In Zhu and Laptev (2017), a Bayesian LSTM (BLSTM) model is proposed for the probabilistic forecast. The parametric models such like Wang et al. (2017), Flunkert et al. (2017), and Zhu and Laptev (2017) need a preemptive assumption on the distribution of the forecast errors or forecast target (e.g., normal), and this might result in considerable mismatches between model and practice. Recently proposed Memory-based Ordinal Regression Deep (MOrdReD)

neural network (Orozco et al., 2018) is another deep-learning-based probabilistic forecast model that has converted the probabilistic forecast task into a multi-class classification one. Though MOrdReD is non-parametric, it is NMO. In this work, a deep-learning-based non-parametric MO quantile forecast model is developed, i.e., the DQFN, which takes in explanatory information and outputs forecasted quantiles in an end-to-end manner. Considering the periodicity of renewable energy generation, e.g., diurnal and seasonal patterns, the BiLSTM model is adopted as the building block in DQFN, since BiLSTM has shown effectiveness for extracting periodicity features (Liu et al., 2017). The DQFN is trained using the IGD approach, and substantive practical experiments in four cases have demonstrated the superiority of DQFN over the state-of-the-arts and the superiority of IGD over heuristic searches.

The main contributions of this work can be summarized as follows:

(1) An investigation on the differentiability of two most crucial quantile forecast evaluation metrics, i.e., APD and SS, is first presented in detail, which systematically revealed the obstacles that have resisted the application of deep learning in the non-parametric quantile forecast domain.

(2) The IGD, a GPU-compatible training approach for (NN-based) MO quantile forecast models, is proposed. With the proposal of IG and customized losses, the IGD has overcome the non-differentiability of evaluation metrics and metric-embedded loss functions, and it can be applied to any NN-based MO quantile forecast models. Substantive case studies using practical data have verified the superiority of IGD in terms of effectiveness and efficiency over heuristic searches.

(3) A deep-learning-based non-parametric MO quantile forecast network is developed, i.e., DQFN. The DQFN is built based on deep residual BiLSTM, and it is trained using IGD with GPU acceleration. Practical experiments have verified the superiority of DQFN in terms of forecast performance over state-of-the-art models.

The rest of this paper is organized as follows: Section 2 introduces the evaluation metrics of quantile forecast and their derivability; Section 3 presents the IGD and customized losses; Section 4 details the framework of DQFN. Section 5 provides the results of the practical experiments. Finally, conclusions are drawn in Section 6.

2. Evaluation Metrics of Quantile Forecast and Their Derivability

2.1. Problem Formulation of Quantile Forecast

Given a set of information-target pairs as $\{(\mathbf{x}_t, y_t)\}_{t=1}^{T}$, where t is the timestamp, T is the total number of timestamps, $\mathbf{x}_t$ denotes available explanatory information (covariate) for quantile forecast at time t (such as historical wind/solar power and weather conditions in wind/solar power forecast scenarios), and y_t is the forecast target at time t. Point forecast essentially aims to build a mapping from historical information to the future target, which can be formulated as:

$$\mathbf{x}_{t-D+1}, \mathbf{x}_{t-D+2}, \cdots, \mathbf{x}_t \rightarrow \hat{y}_{t+h} \tag{1}$$

where D denotes the lag interval, h denotes the lead time, and $\hat{y}_{t+h}$ denotes the forecast of y_{t+h}. As y_{t+h} is a random variable, point forecast yields too little information on its distribution and uncertainty. If denote $F_{t+h}(\cdot)$ as the Cumulative Distribution Function (CDF) of y_{t+h}, the quantile of y_{t+h} with nominal proportion α (denoted as q_{t+h}^{α}) can be obtained through the inverse function of CDF:

$$q_{t+h}^{\alpha} = \left(F_{t+h}\right)^{-1}(\alpha), \text{ thus } P\left(y_{t+h} \leq q_{t+h}^{\alpha}\right) = \alpha \tag{2}$$

Quantile forecast models aim to estimate the quantiles accurately, so that much more information on the distribution of y_{t+h} can be obtained than point forecast. If denote explanatory variable series

$[\mathbf{x}_{t-D+1}, \mathbf{x}_{t-D+2}, \cdots, \mathbf{x}_t]$ anew as $\mathbf{X}_t$ for concise, quantile forecast aims to build a mapping as follows:

$$\mathbf{X}_t \rightarrow [\hat{q}_{t+h}^{\alpha_1}, \hat{q}_{t+h}^{\alpha_2}, \cdots, \hat{q}_{t+h}^{\alpha_r}] \tag{3}$$

where r is the total number of nominal proportions considered, $\hat{q}_{t+h}^{\alpha}$ denotes the forecast of q_{t+h}^{α}. If write the quantile forecast results at time t anew as $\hat{\mathbf{q}}_{t+h} = [\hat{q}_{t+h}^{\alpha_1}, \hat{q}_{t+h}^{\alpha_2}, \cdots, \hat{q}_{t+h}^{\alpha_r}]$, then the forecasted quantiles across all timestamps can be denoted as $\hat{\mathbf{Q}} = [\hat{\mathbf{q}}_1, \hat{\mathbf{q}}_2 \cdots, \hat{\mathbf{q}}_T]$, $\hat{\mathbf{Q}} \in \mathbb{R}^{(T \times r)}$. If denote $\mathbf{y} = [y_1, y_2, \cdots, y_T]$ as all the observations (realizations) of forecast targets, then a quantile forecast evaluation metric can be regarded as a measure of the mismatch between $\hat{\mathbf{Q}}$ and $\mathbf{y}$, i.e., $\mathcal{E}(\hat{\mathbf{Q}}, \mathbf{y})$, where $\mathcal{E}(\cdot)$ is the evaluation function that maps forecasted quantiles into metrics.

2.2. Quantile Forecast Evaluation Metrics and Their Derivability

2.2.1. Strictly Monotone Premises

Before introducing evaluation metrics, the Strictly Monotone Premises (SMP) is introduced first. The SMP defines a reasonable relationship among all the forecasted quantiles, and it is also a prerequisite for the validity of evaluation metrics. The SMP is defined as follows:

$$\forall i, j \in [1, r], i, j \in N^+ \text{: if } \alpha_i < \alpha_j, \text{ then } \hat{q}_t^{\alpha_i} < \hat{q}_t^{\alpha_j} \tag{4}$$

Any point satisfying SMP in the space $\mathbb{R}^{(T \times r)}$ is a feasible point of $\hat{\mathbf{Q}}$. The union of all the feasible points is the feasible region (denoted as $\mathbb{D}_{\text{SMP}}$), which is an *open set*.

2.2.2. Average Proportion Deviation

Reliability is the most crucial evaluation perspective of the probabilistic forecast. APD is the key evaluation metric from the perspective of reliability (Wan et al., 2017). Its definition is as follows:

$$\text{APD}^{\alpha_i} = \frac{1}{T} \sum_{t=1}^{T} \varepsilon \left(\hat{q}_t^{\alpha_i} - y_t \right) - \alpha_i \tag{5}$$

where APD^{α_i} denotes the APD corresponding to nominal proportion α_i, $\varepsilon(\cdot)$ is the indicator function defined as follows:

$$\varepsilon(x) = \begin{cases} 1, \text{ if } x \geq 0 \\ 0, \text{ if } x < 0 \end{cases} \tag{6}$$

The closer APD^{α_i} is to 0, the better forecasted quantiles are in terms of reliability. Obviously, APD^{α_i} is a piecewise function of $\hat{q}_t^{\alpha_i}$. For convenient narration, define the *Piecewisely Derivable Piecewise Function (PDPF)* as follows:

Definition 1. *(Piecewisely Derivable Piecewise Function)*

For a piecewise function $W(\cdot)$ defined on an open set $\mathbb{D}_P$, if it can be written into the following form:

$$W(\mathbf{x}) = \sum_{i=1}^{N_W} f_i(\mathbf{x}) \cdot \sigma_i(\mathbf{x}) \tag{7}$$

where N_W is the number of pieces, and $f_i(\cdot)$ is a derivable function defined on $\mathbb{D}_P$. $\sigma_i(\cdot)$ is defined as follows:

$$\sigma_i(\mathbf{x}) = \begin{cases} 1, \text{ if } \mathbf{x} \in S_i \\ 0, \text{ if } \mathbf{x} \notin S_i \end{cases} \tag{8}$$

where S_i is $f_i(\cdot)$'s active region, and it is a continuous set. $\{S_i\}_{i=1}^{N_W}$ forms a partition of $\mathbb{D}_P$.

Then, $W(\cdot)$ is a PDPF.

Obviously, PDPF is derivable at all the points inside the regions bounded by the pieces (i.e., excluding points on the inter-piece boundaries). Moreover, if a PDPF is also derivable at all the inter-piece boundaries, then it is essentially an everywhere-derivable function. Combining (5), (6), (7), and (8), one can obtain that APD^{α_i} is a PDPF of $\hat{q}_t^{\alpha_i}$, which equivalently means that APD^{α_i} is also a PDPF of $\hat{\mathbf{Q}}$ on $\mathbb{D}_{\mathrm{SMP}}$ (in higher dimensions, of course). Besides, it is not hard to prove that APD^{α_i} is not derivable on the inter-piece boundaries because of $\varepsilon(\cdot)$. Thus, APD^{α_i} is a PDPF of $\hat{\mathbf{Q}}$ on $\mathbb{D}_{\mathrm{SMP}}$ but with non-derivable points inside $\mathbb{D}_{\mathrm{SMP}}$.

Moreover, the average absolute APD over all the nominal proportions can be considered as an overview of reliability (denoted as $\overline{|\mathrm{APD}|}$):

$$\overline{|\mathrm{APD}|} = \sum_{i=1}^{r} \frac{|\,\mathrm{APD}^{\alpha_i}\,|}{r} \tag{9}$$

2.2.3. Skill Score

Under the guarantee of satisfying reliability, the sharpness of forecasted quantiles is also a pursuit (Pinson and Kariniotakis, 2010). SS (Wan et al., 2017) is a comprehensive quantile forecast evaluation metric that has fused both reliability and sharpness:

$$\mathrm{SS} = \frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{r} S_t^{\alpha_i} \tag{10}$$

where

$$S_t^{\alpha_i} = \left[\varepsilon\left(\hat{q}_t^{\alpha_i} - y_t\right) - \alpha_i\right] \cdot \left(y_t - \hat{q}_t^{\alpha_i}\right) \tag{11}$$

One can infer from (11) that SS is a PDPF of $\hat{\mathbf{Q}}$, since $S_t^{\alpha_i}$ can be rewritten as follows:

$$S_t^{\alpha_i} = \begin{cases} (1-\alpha_i)\cdot\left(y_t - \hat{q}_t^{\alpha_i}\right), \text{ if } \hat{q}_t^{\alpha_i} \geq y_t \\ -\alpha_i \cdot \left(y_t - \hat{q}_t^{\alpha_i}\right), \text{ if } \hat{q}_t^{\alpha_i} < y_t \end{cases} \tag{12}$$

Moreover, $S_t^{\alpha_i}$ is obviously not derivable versus $\hat{q}_t^{\alpha_i}$ when $\hat{q}_t^{\alpha_i} = y_t$. Thus, similar to APD, SS is also a PDPF of $\hat{\mathbf{Q}}$ on $\mathbb{D}_{\mathrm{SMP}}$ with non-derivable points inside $\mathbb{D}_{\mathrm{SMP}}$. Since SS is negatively defined but positively oriented, higher SS implies better quantile forecast.

3. Indicator Gradient Descent and Customized Losses

3.1. Challenges: The Difficulty of Training Neural-network-based Metric-oriented Models

Most NN-based MO quantile forecast models are developed to map explanatory information into forecasted quantiles, and their loss functions are constructed based on several quantile forecast evaluation metrics. One may as well call these NNs as Metrics-In-Loss NNs (MILNNs). In general, a suitable collocation of an NN's *loss function* and *training algorithm* should meet the following two requirements:

Requirement $\mathcal{A}$: Minimizing the loss function must be *practicable* (implementable) for the training algorithm. Namely, a function that is problematic to minimize should not serve as the loss function.

Requirement $\mathcal{B}$: Minimizing the loss function must be *favorable* to obtain a more satisfactory NN (which should have a better quantile forecast performance in this scenario).

It is worth mentioning that Requirement $\mathcal{B}$ is the exact basis of MO models, since they have directly embedded the quantile forecast evaluation metrics into the loss functions of their NNs. Namely,

$$\mathcal{L}\left[\mathcal{E}_1(\hat{\mathbf{Q}}, Y), \mathcal{E}_2(\hat{\mathbf{Q}}, Y), \cdots, \mathcal{E}_M(\hat{\mathbf{Q}}, Y)\right] \tag{13}$$

where M is the number of involved metrics, $\mathcal{L}(\cdot)$ is a metric-embedded loss function that fuses various metrics into a single-value loss, which can be a linear combination (Wan et al., 2013b,a) or a more complex form (such as CWC (Khosravi and Nahavandi, 2013; Khosravi et al., 2010b; Quan et al., 2013)). By embedding metrics into $\mathcal{L}(\cdot)$ directly, MO models can achieve better evaluation performance. However, as are discussed in Sections 2.2.2 and 2.2.3, two crucial evaluation metrics of quantile forecast, i.e., APD^{α_i} and SS, are not everywhere-derivable. Thus, any MILNN whose loss function has APD^{α_i} or SS embedded cannot be directly optimized by GD (i.e., Requirement $\mathcal{A}$ is not satisfied). Existing studies have resorted to heuristic searches, which, however, suffers from low computational efficiency, scalability, and generalizability.

3.2. Solutions: Indicator Gradient Descent and Customized Losses

Since powerful NNs (especially deep neural nets) usually possess numerous trainable parameters, training them through heuristic searches is too time-consuming or even impracticable. As GD-based training approaches have proven quite efficient and suitable for deep NNs due to their high compatibility with GPUs, a "GD-like" training approach, i.e., the IGD, is proposed in this work for training MILNNs. IGD takes two steps to overcome the non-differentiability of metric-embedded loss functions, through which both Requirement $\mathcal{A}$ and $\mathcal{B}$ in Section 3.1 can be satisfied. The two steps are as follows:

Step #1: IG is proposed and utilized to overcome the non-differentiability of evaluation metrics, such that the training of an MILNN is implementable (and compatible with GPUs). By taking Step #1, Requirement $\mathcal{A}$ can be satisfied.

Step #2: Through an elaborate constructing of $\mathcal{L}(\cdot)$, MILNNs' loss function has been specially customized based on the metrics to ensure that each iteration in IGD always leads to better MILNNs (which would achieve a more satisfying quantile forecast performance). By taking Step #2, Requirement $\mathcal{B}$ can be satisfied.

Steps #1 and #2 are detailed in Sections 3.3 and 3.4, respectively.

3.3. Step #1: Indicator Gradient and Indicator Gradient Descent Satisfy Requirement $\mathcal{A}$

Since both APD and SS are not derivable versus $\hat{\mathbf{Q}}$ at their inter-piece boundaries, optimizing them using GD directly is infeasible. In this work, IG and IGD are proposed to overcome their non-differentiability, which finally makes MILNNs trainable in a "GD-like" manner. IG (denoted as Λ) is defined by the following rules:

Definition 2. *(IG)*

i) IG of $\varepsilon(\cdot)$ is always 0

$$\Lambda_x \varepsilon(x) \equiv 0 \tag{14}$$

ii) For a function $F(\cdot)$, if it is derivable at $\mathbf{x}$, then its IG at $\mathbf{x}$ is identical to the gradient:

$$\Lambda_{\mathbf{x}} F(\mathbf{x}) \equiv \nabla_{\mathbf{x}} F(\mathbf{x}) \tag{15}$$

iii) IG follows the chain rule, the same as the gradient method when differentiating the compound functions. Namely, given two functions as $\mathbf{f} = F(\mathbf{x})$ and $z = Z(\mathbf{f})$, the compound function of z corresponding to $\mathbf{x}$ can be written as $z = Z[F(\mathbf{x})]$. Then, the IG of z versus $\mathbf{x}$ is:

$$\Lambda_{\mathbf{x}} z = \Lambda_{\mathbf{x}} \mathbf{f} \cdot \Lambda_{\mathbf{f}} z \tag{16}$$

The fatal limitation of GD is that the gradient at a non-derivable point does not exist. As introduced in Sections 2.2.2 and 2.2.3, both APD and SS are PDPFs with non-derivable points inside $\mathbb{D}_{\text{SMP}}$, thus they cannot be directly optimized using GD. In contrast, as is introduced in Theorems 1 and 2 in the Appendix A, the IG of a PDPF always exists regardless of its derivability. As both APD and SS are PDPFs, optimizing APD and SS using IG directly is now practical and implementable, which satisfies Requirement $\mathcal{A}$ in Section 3.1. Besides, the relationship between IG and gradient for an everywhere-derivable PDPF is presented in Theorem 3 in the Appendix A.

3.4. Step #2: Customized Losses Satisfy Requirement $\mathcal{B}$

Though IG has met Requirement $\mathcal{A}$ in Section 3.3, Requirement $\mathcal{B}$ still needs to be satisfied, which means that training MILNNs using IGD must lead to a better resultant quantile forecast performance, namely:

$$\forall m \in \{1, 2, \cdots, M\},\ \lim_{\lambda \to 0} \left[\mathcal{E}_m(\hat{\mathbf{Q}}_j, \mathbf{y}) \text{ is better than } \mathcal{E}_m(\hat{\mathbf{Q}}_{j-1}, \mathbf{y})\right] \tag{17}$$

where j denotes the sequence number during the iterative update of an MILNN trained by IGD (the update of this MILNN's parameters, of course). $\hat{\mathbf{Q}}_j$ denotes the forecasted quantiles from this MILNN after the j-th update using IGD. λ denotes the step size of each update. Then for a sufficiently small λ, each update of this MILNN implemented by IGD should lead to more satisfactory metrics. To achieve this goal, losses from the aforementioned APD and SS are further customized as follows.

3.4.1. Loss Corresponding to Average Proportion Deviation

Loss from all the APDs corresponding to different nominal proportions is customized as follows:

$$\mathcal{L}_{\text{APD}} = \sum_{i=1}^{r} \left(\text{APD}^{\alpha_i} \cdot \sum_{t=1}^{T} \hat{q}_t^{\alpha_i} \right) \tag{18}$$

where $\mathcal{L}_{\text{APD}}$ denotes the loss of MILNNs from APD. The partial IG of $\mathcal{L}_{\text{APD}}$ on $\hat{q}_t^{\alpha_i}$ is:

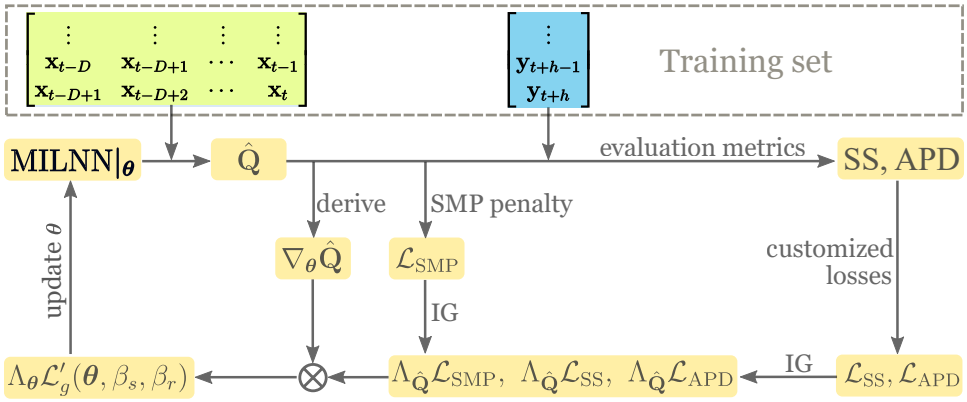
$$\begin{aligned} \Lambda_{\hat{q}_t^{\alpha_i}} \mathcal{L}_{\text{APD}} &= \Lambda_{\hat{q}_t^{\alpha_i}} \text{APD}^{\alpha_i} \cdot \sum_{t=1}^{T} \hat{q}_t^{\alpha_i} + \text{APD}^{\alpha_i} \cdot \Lambda_{\hat{q}_t^{\alpha_i}} \left(\sum_{t=1}^{T} \hat{q}_t^{\alpha_i} \right) \\ &= \Lambda_{\hat{q}_t^{\alpha_i}} \left[\frac{1}{T} \sum_{t=1}^{T} \varepsilon \left(\hat{q}_t^{\alpha_i} - y_t \right) - \alpha_i \right] \cdot \sum_{t=1}^{T} \hat{q}_t^{\alpha_i} + \text{APD}^{\alpha_i} \cdot 1 \\ &= \frac{1}{T} \Lambda_{\hat{q}_t^{\alpha_i}} \varepsilon \left(\hat{q}_t^{\alpha_i} - y_t \right) \cdot \sum_{t=1}^{T} \hat{q}_t^{\alpha_i} + \text{APD}^{\alpha_i} \end{aligned} \tag{19}$$

According to (14), one has:

$$\begin{aligned} \Lambda_{\hat{q}_t^{\alpha_i}} \mathcal{L}_{\text{APD}} &= \frac{1}{T} \Lambda_{\hat{q}_t^{\alpha_i}} \varepsilon \left(\hat{q}_t^{\alpha_i} - y_t \right) \cdot \sum_{t=1}^{T} \hat{q}_t^{\alpha_i} + \text{APD}^{\alpha_i} \\ &= 0 + \text{APD}^{\alpha_i} = \text{APD}^{\alpha_i} \end{aligned} \tag{20}$$

Thus, IGD of $\mathcal{L}_{\text{APD}}$ on $\hat{q}_t^{\alpha_i}$ essentially follows the following step:

$$\hat{q}_t^{\alpha_i} \leftarrow \ \hat{q}_t^{\alpha_i} - \lambda \cdot \Lambda_{\hat{q}_t^{\alpha_i}} \mathcal{L}_{\text{APD}} = \hat{q}_t^{\alpha_i} - \lambda \cdot \text{APD}^{\alpha_i} \tag{21}$$


Training set
MILNN|θ
Q̂
evaluation metrics
SS, APD
derive
SMP penalty
update θ
∇θQ̂
ℒSMP
customized losses
IG
IG
ΛθℒSMP, ΛQ̂ℒSS, ΛQ̂ℒAPD
ℒSS, ℒAPD

3.5. Training Metrics-in-loss Neural Networks with Indicator Gradient Descent

3.5.1. Penalty Loss from Strictly Monotone Premises

Before determining the loss function of MILNNs eventually, a penalty loss is formulated for the guarantee of SMP:

$$\mathcal{L}_{\text{SMP}} = \frac{1}{T \cdot r} \sum_{t=1}^{T} \sum_{i=0}^{r} \left(q_t^{\alpha_i} - q_t^{\alpha_{i+1}}\right) \cdot \varepsilon \left(q_t^{\alpha_i} - q_t^{\alpha_{i+1}}\right) \tag{26}$$

where $\mathcal{L}_{\text{SMP}}$ denotes the penalty loss from SMP.

3.5.2. Training Algorithm for Metrics-in-loss Neural Networks

Since SMP is the prerequisite for the validity of evaluation metrics, and also reliability is the most crucial evaluation perspective of the probabilistic forecast, the training of MILNNs is formulated into the following optimization problem:

$$\begin{aligned} &\min_{\boldsymbol{\theta}} \mathcal{L}_{\text{SS}} \\ s.t.\ &\mathcal{L}_{\text{SMP}} \leq 0 \\ &\overline{|\text{APD}|} \leq \xi \end{aligned} \tag{27}$$

where $\boldsymbol{\theta}$ denotes all the trainable parameters of the MILNN. ξ is a threshold which $\overline{|\text{APD}|}$ should be within, e.g., 2%. (27) can be implemented using a Dual Indicator Gradient Descent (DIGD) approach similar to the dual gradient descent, in which firstly Lagrangian function $\mathcal{L}_g(\boldsymbol{\theta}, \beta_s, \beta_r)$ is defined as:

$$\mathcal{L}_g(\boldsymbol{\theta}, \beta_s, \beta_r) = \mathcal{L}_{\text{SS}} + \beta_s \cdot \mathcal{L}_{\text{SMP}} + \beta_r \cdot \left(\overline{|\text{APD}|} - \xi\right) \tag{28}$$

where β_s and β_r are Lagrangian multipliers, and $\beta_s \geq 0$, $\beta_r \geq 0$. Since minimizing $(\overline{|\text{APD}|} - \xi)$ is equivalent to minimizing $\mathcal{L}_{\text{APD}}$ (both aiming to improve the reliability), (28) is rewritten as (29) for making the best use of the properties of $\mathcal{L}_{\text{APD}}$:

$$\mathcal{L}'_g(\boldsymbol{\theta}, \beta_s, \beta_r) = \mathcal{L}_{\text{SS}} + \beta_s \cdot \mathcal{L}_{\text{SMP}} + \beta_r \cdot \mathcal{L}_{\text{APD}} \tag{29}$$

Then, the IG of $\mathcal{L}'_g(\boldsymbol{\theta}, \beta_s, \beta_r)$ versus $\boldsymbol{\theta}$ can be obtained as follows:

$$\begin{aligned} \Lambda_{\boldsymbol{\theta}} \mathcal{L}'_g(\boldsymbol{\theta}, \beta_s, \beta_r) &= \Lambda_{\boldsymbol{\theta}} \mathcal{L}_{\text{SS}} + \beta_s \cdot \Lambda_{\boldsymbol{\theta}} \mathcal{L}_{\text{SMP}} + \beta_r \cdot \Lambda_{\boldsymbol{\theta}} \mathcal{L}_{\text{APD}} \\ &= \Lambda_{\boldsymbol{\theta}} \hat{\mathbf{Q}} \cdot \Lambda_{\hat{\mathbf{Q}}} \mathcal{L}_{\text{SS}} + \beta_s \cdot \Lambda_{\boldsymbol{\theta}} \hat{\mathbf{Q}} \cdot \Lambda_{\hat{\mathbf{Q}}} \mathcal{L}_{\text{SMP}} + \beta_r \cdot \Lambda_{\boldsymbol{\theta}} \hat{\mathbf{Q}} \cdot \Lambda_{\hat{\mathbf{Q}}} \mathcal{L}_{\text{APD}} \\ &= \nabla_{\boldsymbol{\theta}} \hat{\mathbf{Q}} \cdot \Lambda_{\hat{\mathbf{Q}}} \mathcal{L}_{\text{SS}} + \beta_s \cdot \nabla_{\boldsymbol{\theta}} \hat{\mathbf{Q}} \cdot \Lambda_{\hat{\mathbf{Q}}} \mathcal{L}_{\text{SMP}} + \beta_r \cdot \nabla_{\boldsymbol{\theta}} \hat{\mathbf{Q}} \cdot \Lambda_{\hat{\mathbf{Q}}} \mathcal{L}_{\text{APD}} \end{aligned} \tag{30}$$

The $[\boldsymbol{\theta}, \beta_s, \beta_r]$ are updated alternately following the procedure in Algorithm 1. For a more intuitive presentation, the update of $\boldsymbol{\theta}$ is graphically illustrated in Figure 1. Based on IGD and customized losses, the training of MILNNs is now implementable and effective (favorable). Moreover, it is worth mentioning that IGD and customized losses can be applied to any NN configurations.

4. Deep Quantile Forecast Network

Despite all the MILNNs in the literature, a novel candidate MILNN based on deep learning is proposed for quantile forecast in this work, i.e., the DQFN. The DQFN is an end-to-end NN that maps $\mathbf{X}_t$ to $\hat{\mathbf{q}}_{t+h}$ directly, and it is built following a deep residual architecture with BiLSTM as the basic building block. The structure of DQFN is shown in Figure 2.

Algorithm 1: The training of MILNNs based on DIGD.

Hyper-parameters: The size of mini-batch M_b; $[\lambda_{\boldsymbol{\theta}}, n_{\boldsymbol{\theta}}, \lambda_\beta, \xi]$.
Input: Initialized $\boldsymbol{\theta}$ and training set $\{\mathbf{X}_t, y_{t+h}\}_{t=1}^{T}$.
Output: $\boldsymbol{\theta}$ after training.

$\beta_r \leftarrow 10^3$, $\beta_s \leftarrow 10^5$;
while maximum iteration has not been reached **do**
  for $i = 1, \cdots, n_{\boldsymbol{\theta}}$ **do**
    Sample a mini-batch $\{\mathbf{X}_j, y_{j+h}\}_{j=1}^{M_b}$;
    Input $\{\mathbf{X}_j\}_{j=1}^{M_b}$ to MILNN and get $\{\hat{\mathbf{q}}_{j+h}\}_{j=1}^{M_b}$;
    Get APD and SS using $\{\hat{\mathbf{q}}_{j+h}\}_{j=1}^{M_b}$ and $\{y_{j+h}\}_{j=1}^{M_b}$ through (5) and (10);
    Get $\mathcal{L}_{\text{APD}}$, $\mathcal{L}_{\text{SS}}$, and $\mathcal{L}_{\text{SMP}}$ through (18), (23), and (26), respectively;
    $\mathcal{L}'_g(\boldsymbol{\theta}, \beta_s, \beta_r) = \mathcal{L}_{\text{SS}} + \beta_s \cdot \mathcal{L}_{\text{SMP}} + \beta_r \cdot \mathcal{L}_{\text{APD}}$;
    Get $\Lambda_{\boldsymbol{\theta}} \mathcal{L}'_g(\boldsymbol{\theta}, \beta_s, \beta_r)$ through (30);
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \lambda_{\boldsymbol{\theta}} \cdot \Lambda_{\boldsymbol{\theta}} \mathcal{L}'_g(\boldsymbol{\theta}, \beta_s, \beta_r)$;
  Sample a mini-batch $\{\mathbf{X}_j, y_{j+h}\}_{j=1}^{M_b}$;
  Input $\{\mathbf{X}_j\}_{j=1}^{M_b}$ to MILNN and get $\{\hat{\mathbf{q}}_{j+h}\}_{j=1}^{M_b}$;
  Get APD using $\{\hat{\mathbf{q}}_{j+h}\}_{j=1}^{M_b}$ and $\{y_{j+h}\}_{j=1}^{M_b}$ through (5);
  Get $\overline{|\text{APD}|}$ through (9);
  $\beta_r \leftarrow \max\Big[0, \beta_r + \lambda_\beta \cdot \Big(\overline{|\text{APD}|} - \xi\Big)\Big]$;
  Get $\mathcal{L}_{\text{SMP}}$ using $\{\hat{\mathbf{q}}_{j+h}\}_{j=1}^{M_b}$ through (26);
  $\beta_s \leftarrow \max(0, \beta_s + \lambda_\beta \cdot \mathcal{L}_{\text{SMP}})$;

4.1. Bidirectional Long Short-term Memory

As a recurrent neural network, the introduction of memory units and adaptive gating mechanism in LSTM (Hochreiter and Schmidhuber, 1997; Liu et al., 2021) has enabled the capturing of long-term dependence and makes it suitable for time series problems. Given $\mathbf{X}_t = [\mathbf{x}_{t-D+1}, \mathbf{x}_{t-D+2}, \cdots, \mathbf{x}_t]$ as the input series, the information flow in (vanilla) LSTM can be described as follows: for $k = t-D+1, \cdots, t$:

$$\mathbf{i}_k = \rho\left(\mathbf{w}_i \mathbf{x}_k + \mathbf{u}_i \mathbf{h}_{k-1} + \mathbf{b}_i\right) \tag{31}$$

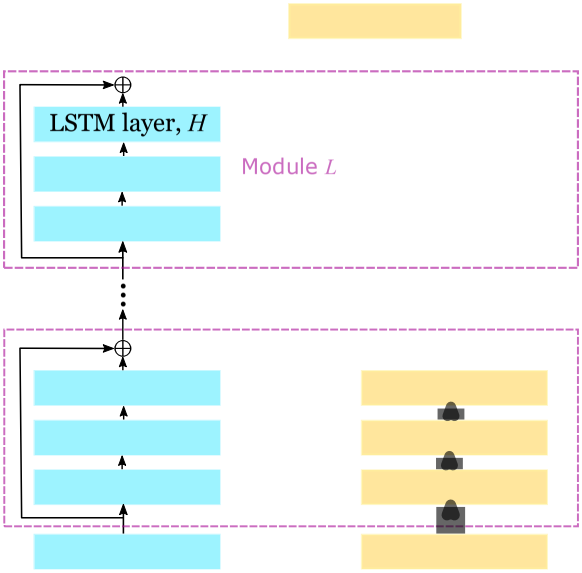
$$\mathbf{f}_k = \rho\left(\mathbf{w}_f \mathbf{x}_k + \mathbf{u}_f \mathbf{h}_{k-1} + \mathbf{b}_f\right) \tag{32}$$

$$\mathbf{c}_k = \mathbf{f}_k \otimes \mathbf{c}_{k-1} + \mathbf{i}_k \otimes \tanh\left(\mathbf{w}_c \mathbf{x}_k + \mathbf{u}_c \mathbf{h}_{k-1} + \mathbf{b}_c\right) \tag{33}$$

$$\mathbf{h}_k = \mathbf{o}_k \otimes \tanh\left(\mathbf{c}_k\right) \tag{34}$$

where $\mathbf{i}_k$ is the input gate, $\mathbf{f}_k$ is the forget gate, $\mathbf{o}_k$ is the output gate, $\mathbf{c}_k$ is the memory unit, $\mathbf{h}_k$ is the hidden state, $\mathbf{w}_*$ and $\mathbf{u}_*$ are weights, $\mathbf{b}_*$ are biases, ρ represents the element-wise sigmoid function, and $\otimes$ represents element-wise multiplication.

Considering the periodicity of renewable energies (diurnal and seasonal patterns), a bidirectional feature learning strategy is adopted to strengthen the LSTM further, i.e., BiLSTM, which is motivated by that BiLSTM has shown effectiveness for extracting periodicity features (Liu et al., 2017). In BiLSTM, another LSTM is built which marches in the opposite direction of the first LSTM, as is shown in Figure 2.



$$\left[\mathbf{x}_{t-D+1},\quad \mathbf{x}_{t-D+2},\ \cdots,\mathbf{x}_{t-1},\quad \mathbf{x}_{t}\right]$$

Table 1: Details of four cases

Cases	Case #1	Case #2	Case #3	Case #4
Scenario	Wind power	Wind power	Solar power	Solar power
Resolution	5 minutes	10 minutes	5 minutes	10 minutes (after merging)
Location	Waterloo Wind Farm, Australia	Guangzhou, China	Royalla Solar Farm, Australia	Univ. of Queensland, Australia
Capacity	111 MW	49.6 MW	20 MW	433 kW
Period	2015/3/1 to 2017/3/1	2016/1/1 to 2017/1/1	2016/5/1 to 2017/8/15, 5:00-19:00	2011/9/1 to 2017/9/1, 5:00-19:00
Lead time	5 minutes	10 minutes	5 minutes	10 minutes
$\mathbf{X}_t$	historical power series, time of the day, day of the year			

5. Practical Experiments

DQFN and IGD were tested through practical experiments in four cases, which demonstrated their superiority in forecasting performance and time-efficiency over state-of-the-art models.

5.1. Four Cases

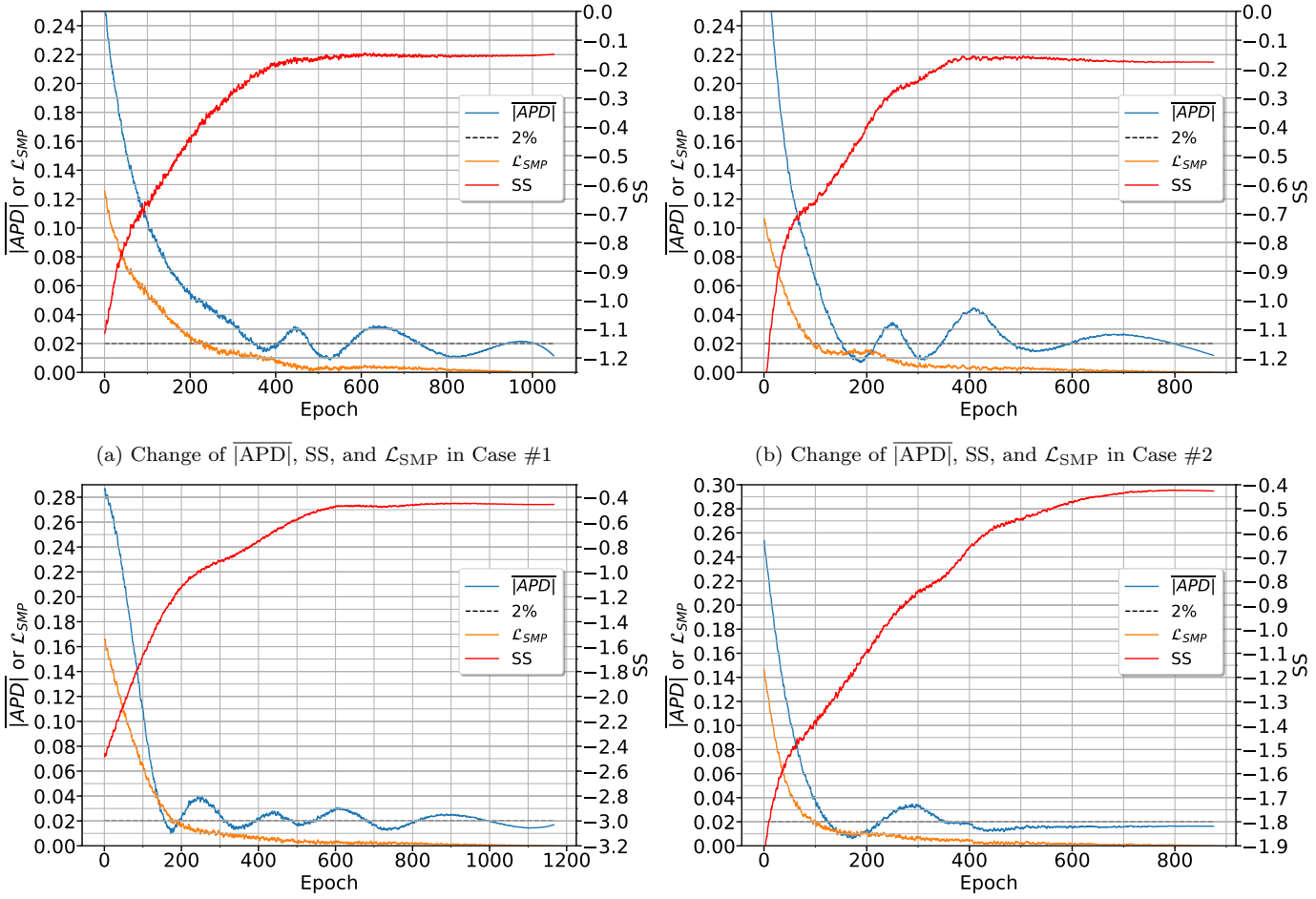
All four cases were based on real-world datasets, which are detailed in Table 1. For market operators, the very-short-term forecast (e.g., 5-min or 10-min ahead) is crucial for optimal dispatch schemes and system operation, since renewable energy is causing increasing uncertainty and fluctuations in power systems. Both the datasets in Cases #1 and #3 are from the national electricity market of Australia (Australian Energy Market Operator, 2016). In the national electricity market, all generators submit their offers and are dispatched every 5 minutes, thus 5-min lead time is of significant importance for generators, consumers, and retailers to offer a reasonable bid (including amount and price). Therefore, in Cases #1 and #3, 5-min ahead forecast is implemented. Besides the 5-min lead time, 10-min ahead wind and solar power forecast have also been considered important for power system operation (Wan et al., 2017; Golestaneh et al., 2016). Thus, in Cases #2 and #4, 10-min ahead forecast was implemented. The datasets in Cases #2 and #4 are from China Southern Power Grid and the University of Queensland (Solar, 2017), respectively. It is worth mentioning that the original resolution of the dataset in Case #4 is one-minute, and it is merged into ten-minute-resolution. Each dataset was divided into three parts: the first 40%, middle 20%, and last 40% as the training, validation, and testing sets, respectively. Moreover, time of the day and day of the year are included in $\mathbf{X}_t$ for considering diurnal and seasonal effects in the same way as introduced in Hu et al. (2020a). Namely, given N_{sec} as the total amount of seconds counting from 00 : 00 to the timestamp of y_{t+k}, then time of the day could be designed into a pair of variables as follows:

$$\left\{\sin\Big(\frac{2\pi N_{sec}}{24\times 3600}\Big), \cos\Big(\frac{2\pi N_{sec}}{24\times 3600}\Big)\right\} \tag{36}$$

Day of the year could be obtained similarly by substituting N_{sec} with the total amount of days between January 1st and the date of y_{t+k} and correspondingly substituting 24×3600 with the total amount of days in this year.

5.2. Structure Determination of Deep Quantile Forecast Networks

The structure of DQFN in each case was determined through a grid search: D was chosen from {4, 16, 64, 256}, L was chosen from {2, 4, 8, 16}, H was chosen from {8, 16, 32, 64}, and the width of the fully-connected layer was chosen from {10, 50, 100, 500}, during which the combination that



(a) Change of $\overline{|\text{APD}|}$, SS, and $\mathcal{L}_{\text{SMP}}$ in Case #1

(b) Change of $\overline{|\text{APD}|}$, SS, and $\mathcal{L}_{\text{SMP}}$ in Case #2

(c) Change of $\overline{|\text{APD}|}$, SS, and $\mathcal{L}_{\text{SMP}}$ in Case #3

(d) Change of $\overline{|\text{APD}|}$, SS, and $\mathcal{L}_{\text{SMP}}$ in Case #4

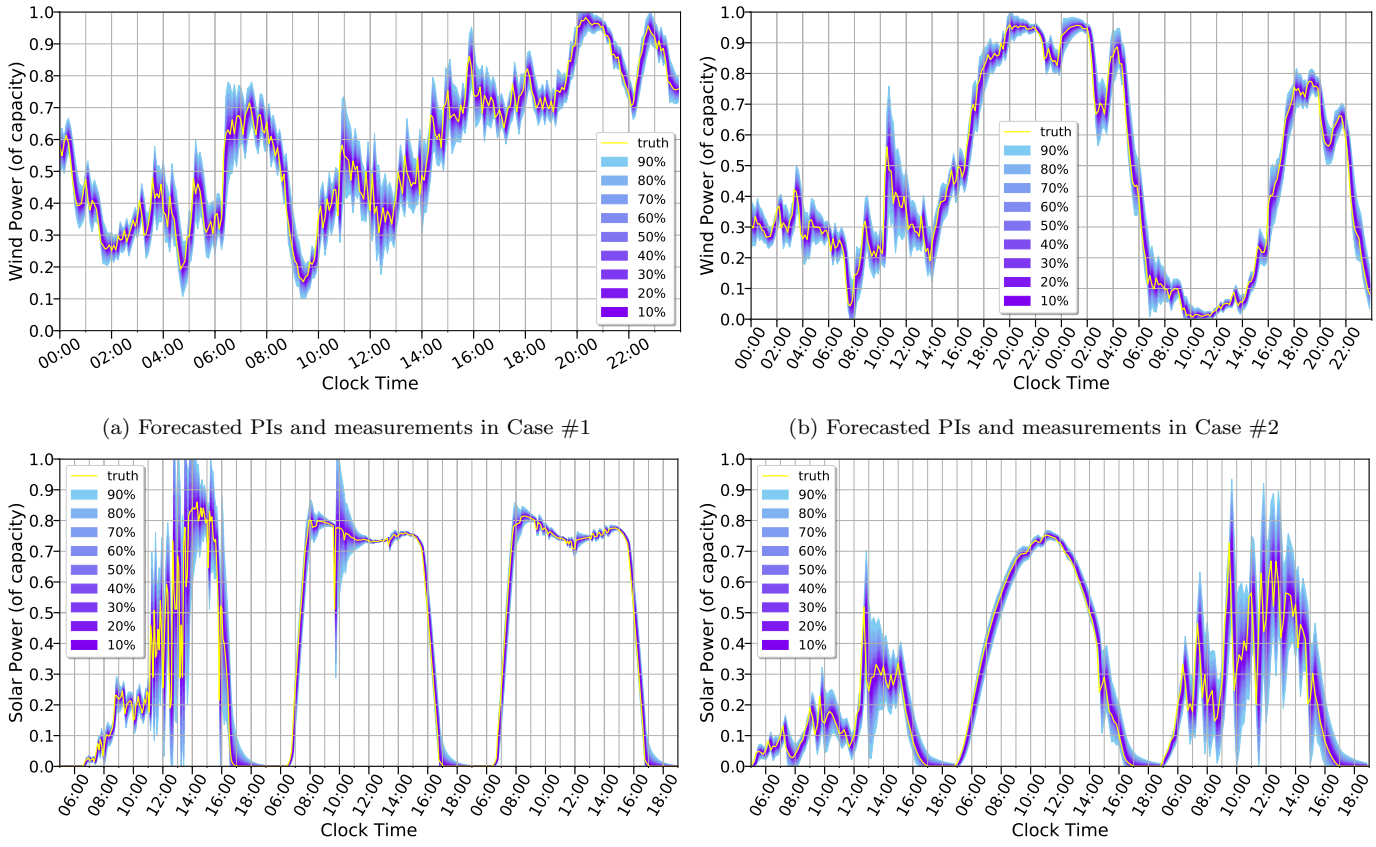
Figure 3: Change of $\overline{|\text{APD}|}$, SS, and $\mathcal{L}_{\text{SMP}}$ for DQFN in the four cases.

had achieved the highest SS on the validation set after training was chosen as the optimal structure of this DQFN, and the reason choosing SS to observe is that SS is a more comprehensive metric than APD (APD only focuses on reliability). Moreover, $[\lambda_{\boldsymbol{\theta}}, n_{\boldsymbol{\theta}}, \lambda_\beta, \xi]$ was set as $[0.001,\ 50,\ 10,\ 2\%]$ in all the cases based on experience in order to reduce the search space of hyper-parameters.

5.3. Quantile Forecast Results of Deep Quantile Forecast Network

The quantiles' nominal proportions that DQFN aims to forecast range from 5% to 95% in 5% steps. The change of $\overline{|\text{APD}|}$, SS, and $\mathcal{L}_{\text{SMP}}$ during the training of DQFN using IGD in the four cases is shown in Figure 3. Figure 3 shows that: 1) the $\mathcal{L}_{\text{SMP}}$ convergences to 0 very fast in all cases, which is caused by the very large initial value of β_s in Algorithm 1. 2) The $\overline{|\text{APD}|}$ tends to vibrate around 2% up and down (except Case #4) similarly with the negative feedback regulation, which has demonstrated the effectiveness of the constraint on $\overline{|\text{APD}|}$ in (27).

To intuitively illustrate the quantile forecast performance of DQFN, PIs formulated from forecasted quantiles in the four cases together with corresponding real measurements (truth) are plotted in Figure 4a-4d, respectively. Figure 4 shows that real measurements can be enclosed by the forecasted PIs in all four cases, indicating the satisfying superiority and generalization ability of DQFN. Moreover, Figure 4 also shows that PIs corresponding to lower nominal proportions are perfectly covered by those corresponding to higher proportions, so that the quantile cross has been avoided, which has verified the effectiveness of $\mathcal{L}_{\text{SMP}}$ in (27).



(a) Forecasted PIs and measurements in Case #1

(b) Forecasted PIs and measurements in Case #2

(c) Forecasted PIs and measurements in Case #3

(d) Forecasted PIs and measurements in Case #4

Figure 4: PIs from the forecasted quantiles of DQFN and measurements in four cases.

5.4. Performance Comparison with Benchmarking Models

Several cutting-edge quantile forecast models belonging to either MO or NMO families were implemented for performance comparison. They are the follows:

1) A probabilistic generalization of persistence following double-censored normal distribution. Its expectation is the last observation, and its variance is obtained by exponential smoothing of recent squared residuals whose forgetting factor is set by finding the highest SS on the validation set.
2) Auto-Regressive including eXogenous covariate (ARX) following double-censored normal distribution whose parameters are obtained by recursive least square estimation. The D of ARX was set by finding the highest SS on the validation set.
3) ARX(D)-GARCH($p,\ q$) (Taylor et al., 2009) carried out following a sliding-window process under double-censored normal distribution, where the orders of GARCH terms and AR Conditional Heteroskedasticity (ARCH) terms are denoted as q and p, respectively. The $\{D,\ p,\ q\}$ were set by finding the highest SS on the validation set.
4) Direct Quantile Regression (DQR) (Wan et al., 2017), an ELM-based MO quantile forecast model whose solver is linear programming.
5) BLSTM (Zhu and Laptev, 2017), a deep-neural-network-based parametric model following double-censored normal distribution.
7) DeepAR (Flunkert et al., 2017), another deep-neural-network-based parametric model following double-censored normal distribution.
7) An MO quantile forecast model proposed in Golestaneh et al. (2016) based on ELM and PSO

Table 2: Performance comparison of all the models

Models	Case #1			Case #2			Case #3			Case #4		
	$\overline{\lvert\text{APD}\rvert}$[A]	SS[A]	time	$\overline{\lvert\text{APD}\rvert}$	SS	time	$\overline{\lvert\text{APD}\rvert}$	SS	time	$\overline{\lvert\text{APD}\rvert}$	SS	time
Persistence	4.90	-17.5	**0.16ms**	5.31	-21.2	**0.16ms**	5.79	-52.6	**0.15ms**	5.78	-46.9	**0.15ms**
ARX	3.13	-16.6	0.17ms	3.49	-19.8	0.17ms	3.97	-51.3	0.16ms	5.25	-47.4	0.16ms
GARCH	2.93	-16.7	23.96ms	3.64	-19.7	24.63ms	3.07	-46.5	26.93ms	4.75	-47.5	29.14ms
BLSTM	3.16	-16.1	4.72ms	3.00	-18.3	5.12ms	2.48	-51.5	6.13ms	3.79	-48.6	6.51ms
ELM-PSO	3.15	-15.2	1.41s	2.31	-18.1	1.50s	3.05	-46.9	2.78s	4.42	-48.6	2.70s
DeepAR	2.83	**-14.9**	1.50ms	2.21	-19.4	1.62ms	3.00	-46.1	2.51ms	4.02	-44.6	3.06ms
DLQR	2.08	-15.7	1.37ms	1.75	-17.8	0.79ms	2.45	-50.7	1.18ms	3.22	-44.9	1.12ms
DQR	1.42	-15.6	8.68ms	1.47	-18.2	9.40ms	2.35	-46.2	9.89ms	3.11	-43.2	9.52ms
DQFN	**1.15**	**-14.9**	10.96ms	**1.18**	**-17.6**	12.25ms	**1.69**	**-45.8**	11.63ms	**1.63**	**-42.6**	10.93ms

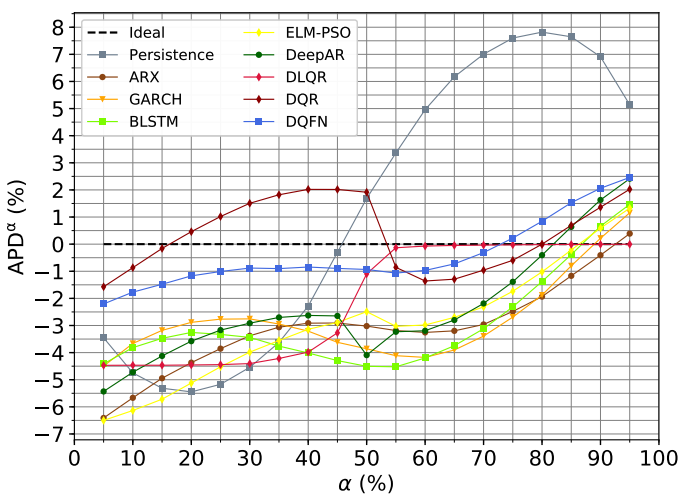
A. $\overline{\lvert\text{APD}\rvert}$ and SS are presented in the form of "%".

(ELM-PSO).

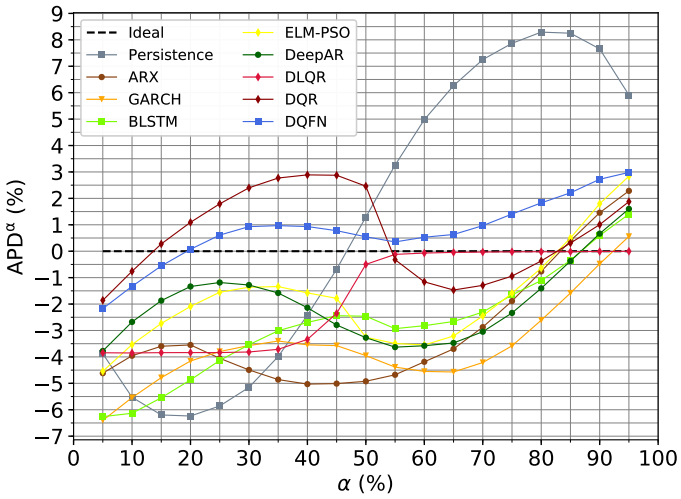
8) Dynamic Linear Quantile Regression (DLQR) carried out using the R package with `quantreg` library and the model `dynrq`.

Amongst them, parametric models include persistence, ARX, GARCH, BLSTM, and DeepAR, while non-parametric models include DQR, ELM-PSO, DLQR, and DQFN. The BLSTM and DeepAR were trained with Adam(Kingma and Ba, 2014) (where $\alpha = 0.0001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$). Their performances on testing sets are shown in Table 2, which demonstrates that: 1) DQFN has performed best in terms of both reliability and sharpness, i.e., the DQFN has achieved the least $\overline{\lvert\text{APD}\rvert}$ and the highest SS in all the cases. It is worth mentioning that for Case #1, both DeepAR and DQFN have achieved the highest SS as -14.9%, while in the other three cases, the highest SS is achieved by DQFN solely. 2) The $\overline{\lvert\text{APD}\rvert}$ of DQFN is all lower than 2% in all the cases, which has verified the effectiveness of the constraint on $\overline{\lvert\text{APD}\rvert}$ in (27). 3) The MO models have performed better than the NMO ones overall, i.e., the DQFN and DQR are the best two among all the models, and ELM-PSO has also performed fairly well compared to the persistence, ARX, GARCH, and BLSTM. This is because embedding metrics into loss functions has enabled MO models to directly pursue the best performance of resultant quantiles. 4) The two MO benchmarking models, i.e., ELM-PSO and DQR, are both based on shallow-structure (three-layer) ELMs, and this has limited their approximation ability and generalizability. As a result, they are unable to perform as well as deep-structure ones, i.e., DeepAR and DQFN, respectively (even though DeepAR is NMO). 5) From the perspective of parametric and non-parametric, the non-parametric models have performed better than parametric ones overall (except that DeepAR has performed better than ELM-PSO due to its deep structure as mentioned above), which indicates that the core assumption of parametric methods, i.e., the forecast target will follow a particular known distribution family, may be inconsistent with practice. Moreover, among the parametric models, the performances of deep-learning-based ones, i.e., BLSTM and DeepAR, are obviously better than the else (especially on reliability).

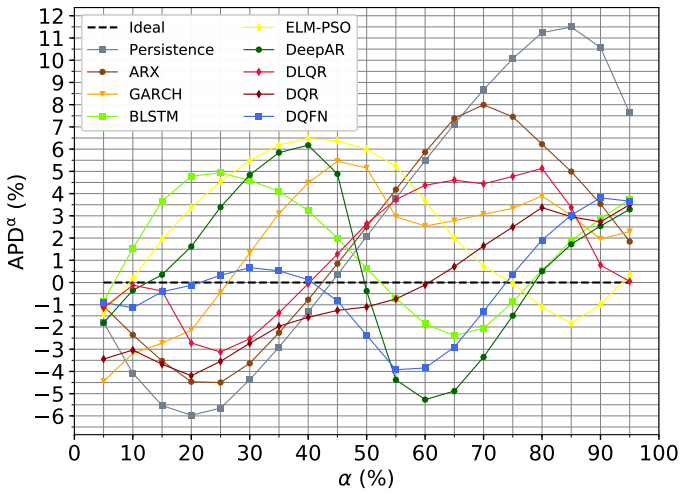
Besides, the reliability of all the models in all the cases is summarized in four reliability diagrams, as is shown in Figure 5. Figure 5 reveals that DQFN has performed quite well on reliability (In fact, Table 2 shows that the $\overline{\lvert\text{APD}\rvert}$ of DQFN is lower than 1.7% in all the cases, which is quite acceptable). While for other models, non-negligible bias can be observed: 1) In Case #1, persistence tends to overestimate the quantiles with high nominal proportions and underestimate the ones with low nominal proportions; ARX, GARCH, BLSTM, ELM-PSO, DeepAR, and DLQR tend to underestimate most quantiles; DQR tends to overestimate the quantiles in [20%, 50%] and [85%, 95%], and underestimate the else; DQFN also tends to underestimate most quantiles slightly. 2) In Case #2, things are quite similar to Case #1, except that DQFN tends to sightly overestimate most quantiles this time. 3) In



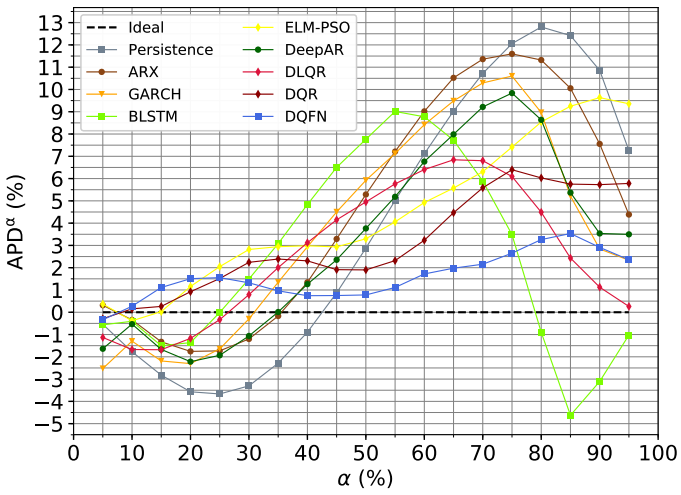
(a) APD of DQFN for Case #1



(b) APD of DQFN for Case #2



(c) APD of DQFN for Case #3



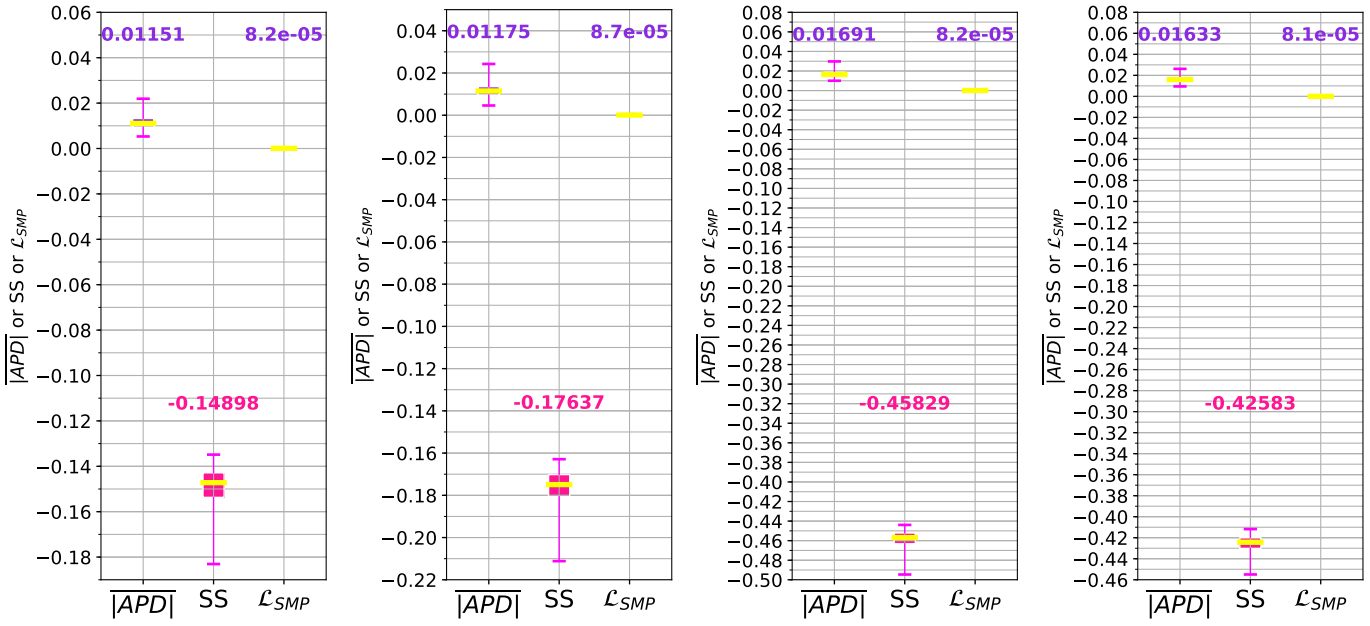
(d) APD of DQFN for Case #4

Figure 5: Reliability diagrams.

Case #3, persistence, ARX, GARCH, DLQR, and DQR tend to overestimate the quantiles with high nominal proportions and underestimate the ones with low nominal proportions; BLSTM, ELM-PSO, and DeepAR tend to underestimate the quantiles with high nominal proportions and overestimate those with low nominal proportions; DQFN tends to slightly overestimate or underestimate those quantiles only with high nominal proportions. 4) In Case #4, three MO models, i.e., DQFN, DQR, and ELM-PSO, tend to overestimate most quantiles; BLSTM tends to overestimate the quantiles in [30%, 75%] and underestimate the else; Other models all tend to overestimate the quantiles with high nominal proportions and underestimate the ones with low nominal proportions. On the whole, the over- or underestimation of DQFN is the slightest among all the models, while the over- or underestimation of the persistence is the severest.

5.5. Time-efficiency of Deep Quantile Forecast Network

All tests were implemented on the same computer with a 3.6 GHz central processing unit, 32 GB memory, and a graphics card (NVIDIA RTX3090) with 24 GB graphics memory. For every model in every test, the total running time (including training and rolling forecasting) averaged at every timestamp is presented in Table 2. Table 2 shows that DQFN has a satisfactory time-efficiency (with



(a) Convergence for Case #1 (b) Convergence for Case #2 (c) Convergence for Case #3 (d) Convergence for Case #4

Figure 6: Convergence analyses for IGD in the four cases. The purple & pink numbers above denote average values of $\overline{|\text{APD}|}$, SS, or $\mathcal{L}_{\text{SMP}}$, respectively.

GPU), i.e., the computation time of DQFN was much shorter than the heuristic-search-based ELM-PSO, and it was only very slightly longer than the computation time of the linear-programming-based DQR, even though the deep-NN-based DQFN has far more training parameters than DQR. Although DQFN is not the fastest one (the fastest one is, beyond doubt, persistence), DQFN's time-efficiency sill has a great potential for practice. This is because the running times of DQFN averaged at every timestamp are only 10.96ms, 12.25ms, 11.63ms, and 10.93ms in these four tests, respectively, and they are indeed much shorter than the corresponding lead times of these tests (5 minutes, 10 minutes, 5 minutes, and 10 minutes, respectively).

5.6. Convergence Analyses of Indicator Gradient Descent

For investigating the convergence property of IGD, the training of the four DQFN instances presented in Figure 3 was re-implemented by another five hundred times each, and the distributions of the eventual $\overline{|\text{APD}|}$, SS, and $\mathcal{L}_{\text{SMP}}$ are demonstrated by box-plots in Figure 6a, 6b, 6c, and 6d, respectively, which demonstrates that: 1) The IGD has performed satisfactorily on convergence, i.e., the variances of the eventual $\overline{|\text{APD}|}$, SS, and $\mathcal{L}_{\text{SMP}}$ were quite low, indicating that DQFNs have been trained to a satisfactory stage mostly. 2) The eventual average $\mathcal{L}_{\text{SMP}}$ is quite low in each case, i.e., 8.2×10^{-5}, 8.7×10^{-5}, 8.2×10^{-5}, and 8.1×10^{-5}, which indicates that the very large initial value of β_s in Algorithm 1 has made the constraint on $\mathcal{L}_{\text{SMP}}$ in (27) highly effective. Thus, the quantile cross has been avoided for very most of the time. 3) It is worth mentioning that sometimes the eventual $\overline{|\text{APD}|}$ is above 2%, which may be caused by that the training of DQFNs has been trapped into local minimums. For overcoming this, the strategies in Adam and Nadam can be introduced into Algorithm 1 to enhance IGD further in future work.

5.7. The Comparison between Long Short-term Memory and Bidirectional Long Short-term Memory

Since the quantile forecast task is based on time series, the utilization and effectiveness of BiLSTM in this paper seem somewhat counter-intuitive. Thus, more numerical tests and comparisons between

Table 3: Comparison Test of LSTM and BiLSTM on Case #1

Model	$\overline{\lvert\text{APD}\rvert}^{\text{A}}$	SS^{A}	time
Double-left DQFN	1.96	-17.9	10.74ms
Double-right DQFN	1.92	-16.3	11.02ms
DQFN	1.15	-14.9	10.96ms

A. $\overline{\lvert\text{APD}\rvert}$ and SS are presented in the form of "%".

LSTM and BiLSTM are carried out to verify the validity of using BiLSTM in this time series forecast task. Namely, the Double-left DQFN (where all the LSTMs in Figure 2 marches towards the left) and Double-right DQFN (where all the LSTMs in Figure 2 marches towards the right) were implemented, and their performances on Case #1 are shown in Table 3.

Table 3 shows that the Double-left DQFN is the worst, which is not surprising. Interestingly, the DQFN performs better than the Double-right DQFN, which implies that BiLSTM is more suitable than LSTM for this task. Indeed, this was not the first comparison test between BiLSTM and LSTM on time series tasks. I.e., Cui et al. (2018) and Siami-Namini et al. (2019) had also carried out detailed comparisons between LSTM and BiLSTM on time series tasks, and their results showed that BiLSTM outperforms LSTM on corresponding time series tasks. However, an in-depth analysis of this phenomenon was missing in these studies.

It may be "attention" that makes BiLSTM better than LSTM. Namely, even the LSTM has a "memory cell", it is still unavoidable that it forgets the information along with time, which can be recognized as "catastrophic forgetting". Therefore, at time spot t, the LSTM has forgotten most of the information at $t - D + 1$, even with a so-called gating mechanism. Consequently, the LSTM has paid much more attention to recent time spots than long from the past, limiting its ability to deal with long time series. While for BiLSTM, the other LSTM marching oppositely has provided strong attention to the information that is long ago from now, relieving the "catastrophic forgetting" to some extend.

5.8. Summary of Practical Experiments

Summarizing both forecasting performance and time-efficiency, one can conclude that DQFN (and IGD) has achieved superior performances over state-of-the-art benchmarking models. Namely, 1) embedding APD and SS into the loss function has led DQFN to pursue the best evaluation performance directly, which thus made it superior to NMO models, i.e., persistence, ARX, GARCH, BLSTM, DeepAR, and DLQR; 2) the deep architecture and high capacity of DQFN have made it superior to existing MO models which have relatively small model capacities, i.e., DQR and ELM-PSO; 3) the satisfactory compatibility of IGD with GPU has made DQFN much more time-efficient than the heuristic-search-based MO model, i.e., ELM-PSO.

The two tasks in this paper, i.e., wind and solar power forecast, are widely studied forecasting problems in the literature. They are of great importance for the operation and control of energy internet and smart-grids. The authors believe that the case studies have shown the superiority of DQFN and IGD in a fair manner.

Of course, it would be better to use a open benchmark problem for comparison. By far, an open benchmark problem or task for quantile forecast where both reliability and sharpness are considered has not been found. Therefore, on the one hand, publishing this dataset to make it an open benchmark for researchers after getting permission from the data provider is under consideration. On the other hand, finding a suitable open benchmark task for quantile forecast where both reliability and sharpness are considered is still being focused.

5.9. Discussion

The proposed indicator gradient, customized losses, and the indicator gradient descent algorithm is a general solution to all the quantile forecast problems, i.e., not limited to the power system field cases presented in this paper. They together have bridged the gap between deep learning and non-parametric quantile forecast, such that various deep neural networks with different architectures can be directly applied to quantile forecast, e.g., deep convolutional neural networks, deep recurrent neural networks, graph neural networks, and even recently proposed transformers, i.e., not limited to DQFN. Besides, quantile forecast, an essential branch of the probabilistic forecast, is the fore-end of optimal control approaches. Namely, the forecasting results of the probabilistic forecast are a crucial input for the subsequent optimization or control tasks, e.g., the stochastic programming, predictive control, and robust control of the micro-grids or energy internet, which has been discussed by Chen et al. (2020), Hua et al. (2019), and Chen et al. (2018) in detail.

Although IGD has been proven a well-defined and effective training approach, the APD of DQFN is still not close to zero enough (APD is indeed vibrating around the threshold, 2%), which means a non-ignorable mismatch exists between real and forecasted quantiles. This mismatch may result from three aspects: 1) Bayes error. For a classification or regression task, Bayes error is the lowest possible error that can be obtained based on offered information, which gives a statistical upper bound on achievable performance. Here, Bayes error can theoretically be lowered by extending explanatory variables with extra useful information. 2) The training of DQFN may fall into local minimums, which results from the non-convexity of neural networks. 3) Under-fitting might occur during the training of DQFN, which results from the fact that IGD is still a first-order optimization algorithm with limited searching efficiency.

6. Conclusions

As a crucial component in the operation and control of hybrid energy systems, to develop advanced forecast models is one of the most efficient ways to accelerate the use of renewable energy in the whole energy chain and contribute to the delivery of the net-zero emission target. In this work, an investigation on the differentiability of two most crucial quantile forecast evaluation metrics, i.e., APD and SS, is first presented in detail, which systematically revealed the obstacles that have resisted the application of deep learning in the non-parametric quantile forecast domain. Then, IGD, a GPU-compatible training approach for NN-based MO quantile forecast models, is proposed. With IG and customized losses, the IGD has overcome the non-differentiability of SS, APD, and metric-embedded loss functions, and it can be applied to any NN-based MO quantile forecast models. Substantive case studies using practical data have verified the superiority of IGD in terms of effectiveness and efficiency over heuristic searches. Moreover, a deep-learning-based non-parametric MO quantile forecast network is developed, i.e., DQFN. The DQFN is built based on deep residual BiLSTM, and it is trained using IGD with GPU acceleration. Practical experiments have verified the superiority of DQFN in terms of forecast performance over state-of-the-art models.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (No. U20B2062), the Beijing Municipal Science & Technology Project (No. Z191100007419001), the Beijing National Research Center for Information Science and Technology, the key Laboratory of Opto-Electronic Information Processing, CAS (No. JGA202004027), the EPSRC under grant EP/R030243/1, UK, and the HVM Catapult Grant under No. 8248 CORE, UK.

Appendix A. Theorems on the Properties of Indicator Gradient

Theorem 1. *Given $W(\cdot)$ as a PDPF on $\mathbb{D}_P$ and any one point on its inter-piece boundaries as $\mathbf{x}_b$, then the IG of $W(\cdot)$ at $\mathbf{x}_b$ always exists, regardless of whether $W(\cdot)$ is derivable at $\mathbf{x}_b$ or not.*

Proof. Assume $\mathbf{x}_b$ belongs to region S_I corresponding to function $f_I(\cdot)$, then IG of $W(\cdot)$ at $\mathbf{x}_b$ can be directly obtained as:

$$\begin{aligned}[\Lambda_{\mathbf{x}} W(\mathbf{x})]|_{\mathbf{x}_b} &= \left\{\sum_{i=1}^{N_W} [\Lambda_{\mathbf{x}} f_i(\mathbf{x}) \cdot \sigma_i(\mathbf{x}) + \Lambda_{\mathbf{x}} \sigma_i(\mathbf{x}) \cdot f_i(\mathbf{x})]\right\}\Bigg|_{\mathbf{x}_b} \\ &= \left\{\sum_{i=1}^{N_W} [\Lambda_{\mathbf{x}} f_i(\mathbf{x}) \cdot \sigma_i(\mathbf{x})]\right\}\Bigg|_{\mathbf{x}_b} = [\Lambda_{\mathbf{x}} f_I(\mathbf{x})]|_{\mathbf{x}_b}\end{aligned} \tag{A.1}$$

According to Definition 1, $f_I(\cdot)$ is derivable on $\mathbb{D}_P$, which means $[\nabla_{\mathbf{x}} f_I(\mathbf{x})]|_{\mathbf{x}_b}$ exists. Then according to (15), one has:

$$[\Lambda_{\mathbf{x}} f_I(\mathbf{x})]|_{\mathbf{x}_b} = [\nabla_{\mathbf{x}} f_I(\mathbf{x})]|_{\mathbf{x}_b} \tag{A.2}$$

Combining (A.1) and (A.2), one has:

$$[\Lambda_{\mathbf{x}} W(\mathbf{x})]|_{\mathbf{x}_b} = [\nabla_{\mathbf{x}} f_I(\mathbf{x})]|_{\mathbf{x}_b} \tag{A.3}$$

□

Theorem 2. *If $W(\cdot)$ is a PDPF on $\mathbb{D}_P$, then for any $\mathbf{x}_J$ inside the pieces of $W(\cdot)$ (i.e., not on boundaries), the IG of $W(\cdot)$ always exists, and it is identical to the gradient of $W(\cdot)$ at $\mathbf{x}_J$.*

Proof. Assume $\mathbf{x}_J$ is inside region S_J corresponding to function $f_J(\cdot)$, then the gradient of $W(\cdot)$ at $\mathbf{x}_J$ can be derived as:

$$[\nabla_{\mathbf{x}} W(\mathbf{x})]|_{\mathbf{x}_J} = \left\{\sum_{i=1}^{N_W} [\nabla_{\mathbf{x}} f_i(\mathbf{x})\sigma_i(\mathbf{x}) + \nabla_{\mathbf{x}} \sigma_i(\mathbf{x}) f_i(\mathbf{x})]\right\}\Bigg|_{\mathbf{x}_J} = [\nabla_{\mathbf{x}} f_J(\mathbf{x})\sigma_J(\mathbf{x}) + \nabla_{\mathbf{x}} \sigma_J(\mathbf{x}) f_J(\mathbf{x})]|_{\mathbf{x}_J} \tag{A.4}$$

Since $\mathbf{x}_J$ is inside S_J, one has:

$$[\sigma_J(\mathbf{x})]|_{\mathbf{x}_J} = 1 \tag{A.5}$$

$$[\nabla_{\mathbf{x}} \sigma_J(\mathbf{x})] = 0 \tag{A.6}$$

Combining (A.4), (A.5) and (A.6), one has:

$$[\nabla_{\mathbf{x}} W(\mathbf{x})]|_{\mathbf{x}_J} = [\nabla_{\mathbf{x}} f_J(\mathbf{x})]|_{\mathbf{x}_J} \tag{A.7}$$

The IG of $W(\cdot)$ at $\mathbf{x}_J$ can be obtained as:

$$\begin{aligned}[\Lambda_{\mathbf{x}} W(\mathbf{x})]|_{\mathbf{x}_J} &= \left\{\sum_{i=1}^{N_W} [\Lambda_{\mathbf{x}} f_i(\mathbf{x})\sigma_i(\mathbf{x}) + \Lambda_{\mathbf{x}} \sigma_i(\mathbf{x}) f_i(\mathbf{x})]\right\}\Bigg|_{\mathbf{x}_J} = \left\{\sum_{i=1}^{N_W} [\Lambda_{\mathbf{x}} f_i(\mathbf{x})\sigma_i(\mathbf{x})]\right\}\Bigg|_{\mathbf{x}_J} \\ &= [\Lambda_{\mathbf{x}} f_J(\mathbf{x})]|_{\mathbf{x}_J} = [\nabla_{\mathbf{x}} f_J(\mathbf{x})]|_{\mathbf{x}_J}\end{aligned} \tag{A.8}$$

Comparing (A.7) and (A.8), one has:

$$[\nabla_{\mathbf{x}} W(\mathbf{x})]|_{\mathbf{x}_J} = [\Lambda_{\mathbf{x}} W(\mathbf{x})]|_{\mathbf{x}_J} \tag{A.9}$$

□

Theorem 3. *If $W(\cdot)$ is an everywhere-derivable PDPF on $\mathbb{D}_P$, then $\Lambda_{\mathbf{x}} W(\mathbf{x}) = \nabla_{\mathbf{x}} W(\mathbf{x})$ for any $\mathbf{x}$ in $\mathbb{D}_P$.*

Proof. For an everywhere-derivable PDPF $W(\cdot)$, its IG can be directly written as:

$$\begin{aligned} \Lambda_{\mathbf{x}} W(\mathbf{x}) &= \sum_{i=1}^{N_W} \left[\Lambda_{\mathbf{x}} f_i(\mathbf{x}) \cdot \sigma_i(\mathbf{x}) + \Lambda_{\mathbf{x}} \sigma_i(\mathbf{x}) \cdot f_i(\mathbf{x})\right] = \sum_{i=1}^{N_W} \left[\Lambda_{\mathbf{x}} f_i(\mathbf{x}) \cdot \sigma_i(\mathbf{x})\right] \\ &= \sum_{i=1}^{N_W} \left[\nabla_{\mathbf{x}} f_i(\mathbf{x}) \cdot \sigma_i(\mathbf{x})\right] = \nabla_{\mathbf{x}} W(\mathbf{x}) \end{aligned} \tag{A.10}$$

□

References

Australian Energy Market Operator, 2016. AEMO 5 Minute Wind Power Data. URL: http://www.aemo.com.au/Electricity/National-Electricity-Market-NEM/Data/Market-Management-System-MMS/Generation-and-Load.

Bengio, Y., Goodfellow, I., Courville, A., 2017. Deep learning. volume 1. Citeseer.

Chen, Y., Deng, C., Li, D., Chen, M., 2020. Quantifying cumulative effects of stochastic forecast errors of renewable energy generation on energy storage soc and application of hybrid-mpc approach to microgrid. International Journal of Electrical Power & Energy Systems 117, 105710. doi:10.1016/j.ijepes.2019.105710.

Chen, Y., Deng, C., Yao, W., Liang, N., Xia, P., Cao, P., Dong, Y., Zhang, Y.a., Liu, Z., Li, D., Chen, M., Peng, P., 2018. Impacts of stochastic forecast errors of renewable energy generation and load demands on microgrid operation. Renewable Energy 133. doi:10.1016/j.renene.2018.09.110.

Cui, Z., Ke, R., Wang, Y., 2018. Deep bidirectional and unidirectional lstm recurrent neural network for network-wide traffic speed prediction. ArXiv abs/1801.02143.

Flunkert, V., Salinas, D., Gasthaus, J., 2017. Deepar: Probabilistic forecasting with autoregressive recurrent networks. arXiv preprint arXiv:1704.04110 .

Golestaneh, F., Pinson, P., Gooi, H.B., 2016. Very Short-Term Nonparametric Probabilistic Forecasting of Renewable Energy Generation; With Application to Solar Energy. Power Systems, IEEE Transactions on PP, 1–14. doi:10.1109/TPWRS.2015.2502423.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778.

He, Y., Liu, R., Li, H., Wang, S., Lu, X., 2017. Short-term power load probability density forecasting method using kernel-based support vector quantile regression and copula theory. Applied energy 185, 254–266.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural computation 9, 1735–1780.

Hu, T., Guo, Q., Li, Z., Shen, X., Sun, H., 2020a. Distribution-free probability density forecast through deep neural networks. IEEE Transactions on Neural Networks and Learning Systems 31, 612–625. doi:10.1109/TNNLS.2019.2907305.

Hu, T., Wu, W., Guo, Q., Sun, H., Shi, L., Shen, X., 2020b. Very short-term spatial and temporal wind power forecasting: A deep learning approach. CSEE Journal of Power and Energy Systems 6, 434–443. doi:10.17775/CSEEJPES.2018.00010.

Hua, H., Qin, Y., Hao, C., Cao, J., 2019. Stochastic optimal control for energy internet: A bottom-up energy management approach. IEEE Transactions on Industrial Informatics 15, 1788–1797. doi:10.1109/TII.2018.2867373.

Kavousi-Fard, A., Khosravi, A., Nahavandi, S., 2015. A new fuzzy-based combined prediction interval for wind power forecasting. IEEE Transactions on Power Systems 31, 18–26.

Khosravi, A., Nahavandi, S., 2013. Combined nonparametric prediction intervals for wind power generation. IEEE Transactions on Sustainable Energy 4, 849–856. doi:10.1109/TSTE.2013.2253140.

Khosravi, A., Nahavandi, S., Creighton, D., 2010a. Construction of optimal prediction intervals for load forecasting problems. IEEE Transactions on Power Systems 25, 1496–1503.

Khosravi, A., Nahavandi, S., Creighton, D., 2013. Prediction Intervals for Short-Term Wind Farm Power Generation Forecasts. IEEE Transactions on Sustainable Energy 4, 602–610. doi:10.1109/TSTE.2012.2232944.

Khosravi, A., Nahavandi, S., Creighton, D., Atiya, A.F., 2010b. Lower upper bound estimation method for construction of neural network-based prediction intervals. IEEE transactions on neural networks 22, 337–346.

Khosravi, A., Nahavandi, S., Srinivasan, D., Khosravi, R., 2014. Constructing optimal prediction intervals by using neural networks and bootstrap method. IEEE transactions on neural networks and learning systems 26, 1810–1815.

Kingma, D., Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 .

Liu, K., Li, Y., Hu, X., Lucu, M., Widanage, W.D., 2020. Gaussian process regression with automatic relevance determination kernel for calendar aging prediction of lithium-ion batteries. IEEE Transactions on Industrial Informatics 16, 3767–3777. doi:10.1109/TII.2019.2941747.

Liu, K., Shang, Y., Ouyang, Q., Widanage, W.D., 2021. A data-driven approach with uncertainty quantification for predicting future capacities and remaining useful life of lithium-ion battery. IEEE Transactions on Industrial Electronics 68, 3170–3180. doi:10.1109/TIE.2020.2973876.

Liu, K., Wei, Z., Yang, Z., Li, K., 2020. Mass load prediction for lithium-ion battery electrode clean production: A machine learning approach. Journal of Cleaner Production , 125159URL: http://www.sciencedirect.com/science/article/pii/S0959652620352033, doi:https://doi.org/10.1016/j.jclepro.2020.125159.

Liu, Y., Zheng, H., Feng, X., Chen, Z., 2017. Short-term traffic flow prediction with conv-lstm, in: 2017 9th International Conference on Wireless Communications and Signal Processing (WCSP), IEEE. pp. 1–6.

Nix, D.A., Weigend, A.S., 1994. Estimating the mean and variance of the target probability distribution, in: Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94), IEEE. pp. 55–60.

Orozco, B.P., Abbati, G., Roberts, S., 2018. Mordred: Memory-based ordinal regression deep neural networks for time series forecasting. arXiv preprint arXiv:1803.09704 .

Pinson, P., Kariniotakis, G., 2010. Conditional prediction intervals of wind power generation. IEEE Transactions on Power Systems 25, 1845–1856.

Quan, H., Srinivasan, D., Khosravi, A., 2013. Short-term load and wind power forecasting using neural network-based prediction intervals. IEEE transactions on neural networks and learning systems 25, 303–315.

Siami-Namini, S., Tavakoli, N., Namin, A.S., 2019. The performance of lstm and bilstm in forecasting time series, in: 2019 IEEE International Conference on Big Data (Big Data), pp. 3285–3292. doi:10.1109/BigData47090.2019.9005997.

Sideratos, G., Hatziargyriou, N.D., 2012. Probabilistic wind power forecasting using radial basis function neural networks. IEEE Transactions on Power Systems 27, 1788–1796. doi:10.1109/TPWRS.2012.2187803.

Solar, U., 2017. UQ SOLAR Photovoltaic Data. URL: http://solar.uq.edu.au/user/reportPower.php?pa=2-2{\&}dtra=day{\&}etp=n.

Taylor, J.W., McSharry, P.E., Buizza, R., 2009. Wind power density forecasting using ensemble predictions and time series models. IEEE Transactions on Energy Conversion 24, 775–782. doi:10.1109/TEC.2009.2025431.

Wan, C., Lin, J., Wang, J., Song, Y., Dong, Z., 2017. Direct Quantile Regression for Nonparametric Probabilistic Forecasting of Wind Power Generation. IEEE Transactions on Power Systems 32, 2767–2778. doi:10.1109/TPWRS.2016.2625101.

Wan, C., Niu, M., Song, Y., Xu, Z., 2016. Pareto optimal prediction intervals of electricity price. IEEE Transactions on Power Systems 32, 817–819.

Wan, C., Xu, Z., Pinson, P., 2013a. Direct interval forecasting of wind power. IEEE Transactions on Power Systems 28, 4877–4878. doi:10.1109/TPWRS.2013.2258824.

Wan, C., Xu, Z., Pinson, P., Dong, Z., Wong, K., 2013b. Optimal prediction intervals of wind power generation. IEEE Transactions on Power Systems 29, 1166–1174. doi:10.1109/TPWRS.2013.2288100.

Wan, C., Xu, Z., Pinson, P., Dong, Z., Wong, K., 2013c. Probabilistic Forecasting of Wind Power Generation Using Extreme Learning Machine. IEEE Transactions on Power Systems 29, 1033–1044. doi:10.1109/TPWRS.2013.2287871.

Wang, H., Li, G., Wang, G., Peng, J., Jiang, H., Liu, Y., 2017. Deep learning based ensemble approach for probabilistic wind power forecasting. Applied energy 188, 56–70.

Xue, Y., Yu, C., Li, K., Wen, F., Ding, Y., Wu, Q., Yang, G., 2016. Adaptive ultra-short-term wind power prediction based on risk assessment. CSEE Journal of Power and Energy Systems 2, 59–64. doi:10.17775/CSEEJPES.2016.00036.

Yan, J., Li, K., Bai, E., Deng, J., Foley, A.M., 2016. Hybrid probabilistic wind power forecasting using temporally local gaussian process. IEEE Transactions on Sustainable Energy 7, 87–95. doi:10.1109/TSTE.2015.2472963.

Yan, J., Li, K., Bai, E., Zhao, X., Xue, Y., Foley, A.M., 2019. Analytical iterative multistep interval forecasts of wind generation based on tlgp. IEEE Transactions on Sustainable Energy 10, 625–636. doi:10.1109/TSTE.2018.2841938.

Yang, Z., Mourshed, M., Liu, K., Xu, X., Feng, S., 2020. A novel competitive swarm optimized rbf neural network model for short-term solar power generation forecasting. Neurocomputing .

Zhang, L., Luh, P.B., 2005. Neural network-based market clearing price prediction and confidence interval estimation with an improved extended kalman filter method. IEEE Transactions on Power Systems 20, 59–66.

Zhu, L., Laptev, N., 2017. Deep and confident prediction for time series at uber, in: Data Mining Workshops (ICDMW), 2017 IEEE International Conference on, IEEE. pp. 103–110.