Exploiting more robust and efficacious deep learning techniques for modeling wind power with speed

Chen, Hao; Staupe-Delgado, Reidar

Published in:
Energy Reports

DOI:
10.1016/j.egyr.2021.11.151

Publication date:
2022

Document Version
Publisher's PDF, also known as Version of record

Available online at www.sciencedirect.com
ScienceDirect
Energy Reports 8 (2022) 864–870
www.elsevier.com/locate/egyr

2021 8th International Conference on Power and Energy Systems Engineering (CPESE 2021), 10–12 September 2021, Fukuoka, Japan

Exploiting more robust and efficacious deep learning techniques for modeling wind power with speed

Hao Chen[b,*], Reidar Staupe-Delgado[a,b]

[a] Department of Social Sciences and Business, Roskilde University (RUC), Denmark
[b] Department of Technology and Safety, UiT The Arctic University of Norway, Tromsø 9019, Norway

Received 30 October 2021; accepted 8 November 2021
Available online 27 November 2021

Abstract

Sound analyses of the nonlinear relationship between wind speed and power generation are crucial for the advancement of wind energy optimization. As an emerging artificial intelligence technology, deep learning has received growing attention from energy researchers for its outstanding ability to provide complex mappings. However, deep neural networks involve complex configurations, making it challenging to utilize them in practice. This paper assesses and presents a number of model-control techniques, categorized as model-oriented and data-oriented, to achieve more robust and efficacious deep neural networks for applications in the nonlinear modeling of wind power with wind speed. These carefully refined models are also compared with polynomials, simple neural networks, and not optimized deep networks with annual data of an Arctic wind farm. The results show that deep networks with sufficient parameter tunings, training optimizations, and modeling exhibit superior performance and generalization, thus possessing considerable advantages in wind energy engineering.

Peer-review under responsibility of the scientific committee of the 2021 8th International Conference on Power and Energy Systems Engineering, CPESE, 2021.

Keywords: Arctic; Deep learning; Modeling control; Neural networks; Nonlinear model; Wind energy

1. Introduction

Wind power is characterized by volatility, randomness, and intermittency. The large-scale grid connection of wind power, therefore, poses certain challenges for the safe and stable operation of the power grid. For grid planning and dispatching, improving the modeling accuracy of wind power can protect the economic scheduling and power balance and can reduce the allocation of energy storage equipment capacity [1]. For wind power parks, accurate models can provide a reliable reference for power generation plans and thus improve production efficiency [2,3]. In particular, short-term prediction of wind power with a forecast horizon from 0 to 24 h is vital in determining the daily operation plan of the grid, and can be produced with numerical weather prediction model data, measured meteorological wind, and observed power as the main inputs.

* Corresponding author.
E-mail address: hao.chen@uit.no (H. Chen).

https://doi.org/10.1016/j.egyr.2021.11.151

Peer-review under responsibility of the scientific committee of the 2021 8th International Conference on Power and Energy Systems Engineering, CPESE, 2021.

Abbreviations

PR	Polynomial Regression
NN	three-layer Neural Network
DN	Deep Network with three hidden layers
MDN	Model-oriented Deep Network
DDN	Data-oriented Deep Network
ODN	Overall optimized Deep Network
RRSE	Root Relative Squared Error
R^2	Coefficient of determination
QR	Qualification Rate
QR75	Qualification Rate of predictive accuracy over 75% to the total test samples
QR90	Qualification Rate of predictive accuracy over 90% to the total test samples

Since wind speed and wind power inherently constitute time series, classical time-series forecasting models have been widely used in wind power modeling. However, newer machine learning algorithms have proven to be effective methods given the nonlinear relations between wind power and wind speed, direction, and other predictors used in wind power planning. In 2006, Hinton and Salakhutdinov successfully trained multilayer neural networks (more than two hidden layers) for the first time as the convolutional network and achieved excellent results on several datasets [4], which also signified the birth of deep learning. Mishra and colleagues compared five common deep learning models, viz. Deep Feed Forward (DFF), Deep Convolutional Network (DCN), Recurrent Neural Network (RNN), Attention mechanism (Attention), and Long Short-Term Memory Networks (LSTM) in modeling monthly wind power and found the DCN and Attention performed best [5]. Zhu and colleagues used a temporal convolutional network to forecast wind power with datasets from the renewable energy laboratory of the United States and demonstrated its edge over traditional learning methods [6].

Although deep learning received considerable attention from researchers in wind power modeling [7], there are still the following issues to be addressed. Firstly, deep learning is a complex neural network model, and the modeling process requires precise skills and parameter tuning, which is harder for energy researchers to consistently deliver. Secondly, deep learning performs poorly in univariate autoregression and sometimes even inferiorly to benchmark statistics. Finally, deep learning requires substantial computational resources and is time-consuming, so currently, it is not very widespread in wind power operations.

In the present study, we have distilled and categorized some techniques, divided by model- and data-oriented, that make deep learning more effective, rapid, and stable, and applied them to wind power modeling. The performance of the boosted models was also compared statistically with the benchmark polynomial fitting, shallow neural networks, and deep networks. This study can assist researchers in the energy field to bypass the tedious and complex mathematical concepts underlying deep learning so that they can quickly acquire the relevant skills and apply them to energy research.

2. Wind data description

Theoretically, the output power of a single wind turbine is given by:

$$P_w = C_P A \rho V^3/2 \tag{1}$$

where P_w is the output power (kW), C_p is the rotor coefficient of performance, ρ is the air mass density (kg/m^3), A is the swept blade area (m^2), and V is the wind speed (m/s). It can be seen that the power generated by wind turbines is proportional to wind speed with a cubic relationship, and wind direction mainly affects A. In a shorter time period, air density does not vary much and serves as a secondary factor.

Inspired by Eq. (1), the power of the entire wind park can be regarded as a function of the third polynomial of wind speed, and if considering wind direction, the power P is expressed by:

$$P = f(u, v, u^2, v^2, u^3, v^3) \tag{2}$$

where f is a nonlinear function, u is defined as horizontal wind speed $u=V\times sin\ \theta$, and v is as vertical wind speed $v=V\times cos\ \theta$, V is the observed wind speed of the site, and θ is its direction angle. The core of this article lies in the proper non-explicit representation of the function f (.) by deep learning approaches.

In this research, a wind power park in Northern Norway, inside the Arctic Circle, is the target. The wind speed and direction data are measured by the site wind mast, and the power data are from the Norwegian Water Resources and Energy Directorate (NVE). The two sets are both with hourly temporal resolution and from 0:00 1st January 2017 to 23:00 31st December 2017.

3. Methodology

As a highly representative part of machine learning, especially neural networks, deep learning has achieved unprecedented breakthroughs in many applications with originality and great success.

Compared with shallow neural networks, the advantages of deep learning are: 1. Better approximation of complex nonlinear functions through more layers with nonlinear structure; 2. The use of layer-by-layer learning and fine-tuning can extract and express high-level abstract concepts while avoiding being trapped in a local optimum; 3. The network architecture is flexible and suitable for various learning tasks [8].

Generally, neural networks perform learning with the stochastic gradient descent algorithm. The loss function of the training data is firstly estimated, and its derivative is calculated and propagated to update the weights of the network with the back-propagation algorithm.

While deep learning has a solid mathematical foundation for modeling, training, optimization, prediction, etc., (these mathematical foundations can be found in deep learning by I Goodfellow et al. [9] due to space constraints of this paper), deep learning engineering is typically hands-on. That is, the configuration of deep learning models is often associated with trial-and-error and heuristic approaches. The reason is the lack of accurate and fast rules for configuring a network for a given problem, and also the high-dimensionality of the space of hyperparameters that must be tuned [10]. On the basis of know-how and practical experience in modeling, we divide the techniques for constructing wind power models with more robust and efficacious deep learning into two categories: model-oriented and result-oriented.

For model-oriented techniques, they mainly include: 1. Taxonomy of deep networks; 2. Networks topology; 3. Art of renunciation; 4. Model ensemble; 5. Multiple models ensemble.

For data-oriented techniques, they primarily cover: 1. Input preprocessing; 2. Training parameters; 3. Weights adjustment; 4. Gradient control; 5. Adding noise.

A brief description of these techniques is described as follows.

3.1. Model-oriented techniques

Taxonomy of deep networks: according to tasks of learning and networks structure, deep learning can be categorized into: Deep Multilayer Perceptrons (DMLPs) for various types of nonlinear mappings; Convolutional Neural Networks (CNNs) mainly for image recognitions; Recurrent Neural Networks (RNNs) for sequential problems; Generative Adversarial Networks (GANs) for generating data; and various network transformations [9].

Networks topology: the basic networks consist of input, hidden, and output layers. Deep networks are all about building more complex, diverse networks based on these three layers. Designing network structures in accordance with tasks is one of the key missions of deep learning. Luckily, emerging APIs for deep learning offer a convenient and efficient way to configure a deep network with few lines of code. Some of the most popular APIs are Theano, Tensorflow, Keras, Pytorch, MXNet, and PaddlePaddle.

Art of renunciation: Deep neural networks have strong nonlinear learning capabilities, but the trained model is likely to behave poorly on a test set unless the problem of overfitting to the training dataset and insufficient generalization is properly addressed. In practice, convenient measures to avoid overfitting are dropout and early stopping. The former implies stochastically excluding a subset of nodes and their connections from the network during epochs of training [11]. This action makes network behavior less dependent on specific neurons, less prone to adapting to the particularities of the training samples, and makes the model more robust and general. Early stopping means to halt the training process when performance on the test dataset has reached its maximum and starts to decrease, which is determined by monitoring the loss functions computed separately on the training set and the test set during training epochs.

Deep learning is a flexible and nonlinear technology, meaning that the results of each training vary with the initial weights and noise in the data (i.e., the results are with high variance). Ensemble learning can reduce this variance.

Model ensemble: it is defined as using a statistical method to reach the ensemble effects inside one model. It includes resampling input data and so-called snapshot ensembles that cleverly obtain multiple trained models by dynamically cycling the learning rate in one training [12]. The former is an ensemble learning on the multiple subsets, made by resampling techniques like k-fold cross-validation and bootstrap aggregation of the training set. The latter is designed for coping with huge computational costs in a single training by systematically and aggressively changing the learning rate with the stochastic gradient descent algorithm in training to establish different networks with varying weights. Each time the model converges, the weights and their related networks are saved in the snapshot ensemble, and the current local minimum escapes by initiating bigger learning rates.

Multiple models ensemble: the method harnesses various combination approaches to ensemble sub-learners. It requires each learner to have contributions to the final model and reduces its variance, which means their prediction errors have small correlations. The combination strategies include simple averaging, dynamic weightings, stacking, and more complicated and intelligent strategies.

3.2. Data-oriented techniques

Inputs preprocessing: this technique refers to the treatment of missing and outlier values of the input data and scaling of variables with different scales, including normalization and standardization. The data scaling leads to more stable and faster learning since it helps balance out the updating of weights, without which exploding gradients might happen.

Training parameters: there are several tuning parameters in configuring the deep networks, such as loss function, learning rate, batch size, and activation function, etc. First, the loss function is the target of a stochastic gradient descent algorithm by which networks are learned. For regression problems, Mean Squared Error (MSE) is a statistically preferred loss function under maximum likelihood inference, assuming that the output is normally distributed [13]. Alternatively, the Mean Absolute Error (MAE), more stable with big outliers, can also be used in regression. Usually, an extra term called regularization is added to the loss function to escape overfitting. Second, the learning rate is a small positive value, usually between 0.0 and 1.0, in determining the size of the learning or updating weights step. If it is very small, the slow reduction in error will lead to long-time training. On the contrary, the big one makes the loss function miss the minimum values. Luckily, due to the introduction of momentum and adaptation in the learning rate, Adaptive Moment Estimation (Adam) and Root Mean Square Propagation (RMSprop) increasing training option with rapid speed and sound convergence [10]. Third, batch size, a crucial learning dynamic, is the number of examples used in each error gradient estimation. It directly influences learning speed and stability. Smaller batch sizes are noisy and generalized. In contrast, larger batch sizes reduce computational time but might skip the minima. Fourth, the activation function, typically chosen as nonlinear, activates the weighted input of the node and outputs it. It allows nodes to learn sophisticated data information. Specifically, the hyperbolic tangent activation function is usually better than the logistic sigmoid [9]. However, in deep networks, these functions make errors decrease sharply with their propagation through the network as the number of layers increases. The solution is to use the rectified linear activation function, ReLU for short, which exhibits several desirable properties.

Weights adjustment: deep networks learn a group of weights that can provide the best map inputs to outputs. A network with big weights might signal an unstable network (small changes may lead to big changes), likely learned noises in the inputs, which is a sign of overfitting. The networks that maintain small weights, called weight regularization, are an ideal choice for improving generalization. Practically, a weight constraint, set with a predefined threshold or minimum and maximum range, is used in checking the magnitude of the weights.

Gradient control: The weights can be updated very significantly or very slightly during the training procedure, which affects the efficiency and effectiveness of the training. The weights can be updated very significantly or very slightly, which affects the efficiency and effectiveness of the training network. A frequent solution is to alter error derivatives, namely gradients, and there are two main methods, which are gradient cropping and scaling. The former is cropping to restrict gradient values to a specific range when the gradient exits the expected scope. The latter is a normalization of the gradient vector so that the scalar value equals a defined value.

Adding noise: Training deep networks on a relatively small dataset may result in memorizing all the data, which causes overfitting. The addition of noise to the learning process is actually a regulation, improving the robustness

and generalization of networks. Noise can be added to the input, weights, gradients, batch size, and almost any other learning parameter. The most popular noise type is white noise, generated with a pseudo-random number generator, with a mean of zero and a standard deviation of one. In particular, according to a study [14], training with weights adding noise requires pairing with early stopping to avoid overfitting.

4. Modeling procedure

To leverage these deep learning tricks in wind power modeling, we build five individual models, three baseline models: polynomial regression (PR), a three-layer neural network (NN), and a deep network with three hidden layers (DN). Three optimized models with the same topology of DN: model-oriented deep network (MDN), data-oriented deep network (DDN), and overall optimized deep network (ODN). The MDN is reinforced with dropout, early stopping, *10*-fold cross-validation, and snapshot ensemble. The DDN is enhanced with RMSprop and dynamic batch size scheme, ReLU in the second hidden layer, weight regularization, gradient scaling, and adding tiny noise in weights. The ODN is strengthened with all tricks in MDN and DDN, and the final model is reached by sub-model average ensembles.

Given that modeling wind power with wind speed is essentially a regression problem in which MSE is the loss function, Root Mean Square Error (RMSE) is naturally chosen as a metric to evaluate model performance. To percentage this metric, Root Relative Squared Error (RRSE), as an extension of RMSE, is used as our evaluation metric. The coefficient of determination (R^2) is also popular for accessing regression performance. The Qualification Rate (QR) examines the proportion of predictive accuracy over 75% to the total test samples. Since only RMSE is negative-oriented, which means a smaller value is related to better performance. So, practically, we alternatively use (1-RMSE). The three metrics are expressed in (3), (4), and (5).

$$RRSE = \sqrt{\sum_{i=1}^{n}(P_{mi} - P_i)^2} \Big/ \sqrt{\sum_{i=1}^{n}(P_i - P_{i,average})^2} \tag{3}$$

$$R^2 = 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}} \tag{4}$$

$$QR = \frac{1}{n}\sum_{i=1}^{n} \begin{cases} 1, \left(1 - \frac{|P_{mi} - P_i|}{Cap}\right) \geqslant Q \\ 0, \left(1 - \frac{|P_{mi} - P_i|}{Cap}\right) < Q \end{cases} \tag{5}$$

where P_{mi} is modeled power, P_i is observed power, $P_{i,average}$ is the average of P_i, Cap is the designed capacity of the site. Q is the quantile percentage; we choose 75% and 90% and represent them with QR75 and QR90, respectively. While SS_{res} is the sum of squares of regression residuals and SS_{tot} is the total sum of squares.

The whole year's wind park data are divided into a training set (70%) and a testing set (30%). Using the above-mentioned model to learn on the training set and calculate the metrics on the testing set, and then analyzing differences in the results.

5. Results and discussions

Firstly, the fitting curve of PR on the training set is explicit as follows Eq. (6) and Fig. 1:

$$P = -.00003u^3 + .2168u^2 + .0234u - .00003v^3 + .2237v^2 - .008v + 1.6672 \tag{6}$$

It is intuitively seen that although the cubic item appears in Eq. (1), the quadratic terms of the horizontal and vertical wind speeds mainly contribute to the linear polynomial fit for wind power with u and v. The power is mathematically approximated as a 3D paraboloid with respect to u and v.

Secondly, for clarity, we pick 100 consecutive samples in the testing set and plot the observed power and the powers modeled with PR, NN, DN, ODN algorithms in Fig. 2, respectively. It is obvious that except for PR (which gives too large projections near the power peaks), the other neural network-based algorithms closely follow trends in the observed power and provide simulated results that are closer to the measured values. Ultimately, Table 1 shows the performance of the six models. Analogously, the last five approaches provide reasonably satisfactory modeling results except for PR, which delivers worse 1-RRSE and R^2, but its QRs are close to the other baseline models.

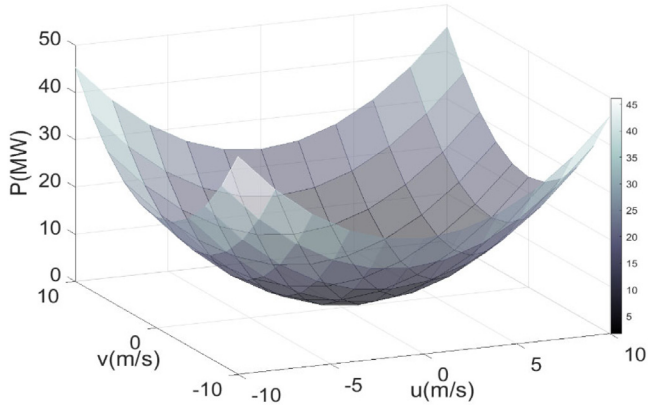

Fig. 1. The polynomial fitting 3D curve of wind power.

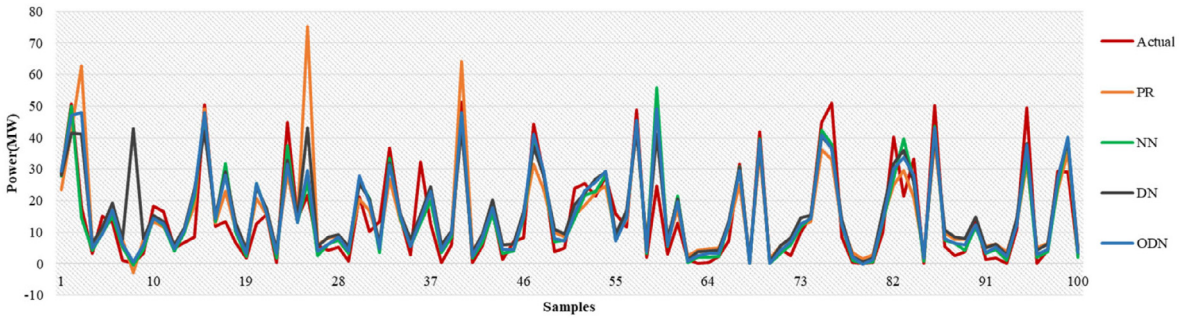

Fig. 2. The comparison of observed and modeled wind power with 100 samples.

Table 1. The performance of six wind power models.

Metrics	PR	NN	DN	MDN	DDN	ODN
1-RRSE	0.3785	0.5556	0.5335	0.5971	0.5976	0.6074
R^2	0.6230	0.8041	0.7941	0.8385	0.8391	0.8458
QR75	0.9136	0.9387	0.9471	0.9513	0.9505	0.9716
QR90	0.6880	0.7420	0.6492	0.7519	0.7576	0.7736

The deep network DN without optimization performs slightly worse than the NN, but the former does significantly worse on projected QRs, especially the one with the higher quantile. The MDN, DDN, and ODN with optimization improve more significantly in all metrics, especially QR90 and except QR75, than all benchmark models. And very similar performance is observed concerning MDN and DDN.

Furthermore, to further investigate the superiority of ODN combining both MDN and DDN, the performance improvements of ODN in comparison with other models are displayed in Table 2. ODN outperforms the first five models in all metrics. Among them, the largest improvements are 1-RRSE and QR90. In particular, the 1-RRSE and QR90 of the ODN are increased by nearly 14% and 19%, respectively, compared to the unoptimized DN. An exception to R^2, ODN also achieves around 2% metric improvement over MDN and DNN.

6. Conclusions

This study elaborates on the techniques of deep neural networks and demonstrated their robustness and effectiveness in wind power modeling by examining an Arctic wind farm as a case study. As per the results of the study, the following conclusions can be drawn:

Table 2. The performance improvement of ODN compared with others.

Metrics	ODN v. PR	ODN v. NN	ODN v. DN	ODN v. MDN	ODN v. DDN
1-RRSE	60.4593%	9.3208%	13.8616%	1.7216%	1.6409%
R^2	35.7714%	5.1957%	6.5220%	0.8756%	0.8095%
QR75	6.3426%	3.4976%	2.5828%	2.1314%	2.2132%
QR90	12.4455%	4.2571%	19.1685%	2.8853%	2.1102%

1. Modeling the power of a whole wind farm with wind speed is a complex nonlinear problem, and the use of the linear regression-based polynomial fit does not provide satisfactory results.
2. Simple three-layer neural networks and deep multilayer networks can assist in basic simulations of the nonlinear relationship between wind speed and power. However, there is still a potential for enhancing unoptimized networks.
3. Both model-oriented and data-oriented deep learning tricks increase the network's ability and generalization, which leads to more precise and robust predictions.
4. Adequate deployment of deep learning techniques cannot only train the networks more effectively, allowing the loss function to find more suitable minima but also provides predictions that are very close to the observations considered valuable for practical wind engineering applications.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The supporting data are available on reasonable request from the authors.

Acknowledgments

This study is supported by the Department of Technology and Safety, UiT The Arctic University of Norway. The measured wind data are the property of Troms Kraft AS. H.C thanks Dr. Yngve Birkelund for organizing data and Dr. Stian Normann Anfinsen for his comments on learning methods. The contribution of Reidar Staupe-Delgado was funded by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 897656.

References

[1] Chang W-Y. A literature review of wind forecasting methods. J Power Energy Eng 2014;2(04):161.
[2] Renani ET, Elias MFM, Rahim NA. Using data-driven approach for wind power prediction: A comparative study. Energy Convers Manage 2016;118:193–203.
[3] Ferreira M, Santos A, Lucio P. Short-term forecast of wind speed through mathematical models. Energy Rep 2019;5:1172–84.
[4] Hinton GE, Salakhutdinov RR. Reducing the dimensionality of data with neural networks. Science 2006;313(5786):504–7.
[5] Mishra S, Bordin C, Taharaguchi K, Palu I. Comparison of deep learning models for multivariate prediction of time series wind power generation and temperature. Energy Rep 2020;6:273–86.
[6] Zhu R, Liao W, Wang Y. Short-term prediction for wind power based on temporal convolutional network. Energy Rep 2020;6:424–9.
[7] Deng X, Shao H, Hu C, Jiang D, Jiang Y. Wind power forecasting methods based on deep learning: A survey. CMES Comput Model Eng Sci 2020;122(1):273–302.
[8] Gulli A, Pal S. Deep learning with keras. Packt Publishing Ltd; 2017.
[9] Goodfellow I, Bengio Y, Courville A, Bengio Y. Deep learning, no. 2. MIT Press Cambridge; 2016.
[10] Brownlee J. Better deep learning: train faster, reduce overfitting, and make better predictions. Machine Learning Mastery; 2018.
[11] Baldi P, Sadowski PJ. Understanding dropout. Adv Neural Inf Process Syst 2013;26:2814–22.
[12] Huang G, Li Y, Pleiss G, Liu Z, Hopcroft JE, Weinberger KQ. Snapshot ensembles: Train 1, get m for free. 2017, arXiv preprint arXiv:1704.00109.
[13] Krzanowski W. Principles of multivariate analysis. OUP Oxford; 2000.
[14] Graves A. Practical variational inference for neural networks. In: Advances in neural information processing systems. Citeseer; 2011, p. 2348–56.