More than accuracy

end-to-end wind power forecasting that optimises the energy system

Wahdany, Dariush; Schmitt, Carlo; Cremer, Jochen L.

DOI
10.1016/j.epsr.2023.109384

Publication date
2023

Document Version
Final published version

Published in
Electric Power Systems Research

Citation (APA)
Wahdany, D., Schmitt, C., & Cremer, J. L. (2023). More than accuracy: end-to-end wind power forecasting that optimises the energy system. *Electric Power Systems Research*, *221*, Article 109384. https://doi.org/10.1016/j.epsr.2023.109384

Electric Power Systems Research 221 (2023) 109384

Contents lists available at ScienceDirect

Electric Power Systems Research

journal homepage: www.elsevier.com/locate/epsr

More than accuracy: end-to-end wind power forecasting that optimises the energy system

Dariush Wahdany [a,b], Carlo Schmitt [a], Jochen L. Cremer [b,*]

[a] Institute for High Voltage Equipment and Grids, Digitalization and Energy Economics, RWTH Aachen University, Schinkelstraße 6, 52062 Aachen, Germany
[b] Department of Electrical Sustainable Energy, TU Delft, Mekelweg 5, 2628 CD Delft, Netherlands

ARTICLE INFO

Keywords:
Sampling
Database generation
Security assessment
Machine learning
Power system operation

ABSTRACT

Weather forecast models are essential for sustainable energy systems. However, forecast accuracy may not be the best metric for developing forecast models. A more or less conservative forecast may be preferred over pure accuracy. For example, forecasting accurately in times of energy-deprived situations may be more important than in times of excess wind power generation. This is not accounted for when learning a wind power forecast without system knowledge. Wind power forecasts directly impact system energy schedules. Therefore, to optimise system costs, our paper proposes a neural network structure for wind power forecasts directly considering varying energy system conditions. To train this neural network optimally, this paper models the system dispatch and their costs as an optimisation problem. Then, this paper connects the neural network with the optimisation model. In this connection, the implicit function theorem and the Karush–Kuhn–Tucker (KKT) conditions provide the system cost gradients during the training of the neurons. A case study using onshore and offshore weather data from Germany and The Netherlands showed forecast errors of system costs reduced by up to 10% with high wind capacity. Wind curtailment was reduced by more than 20% for individual cases but increased overall. Modified cases of the optimal power flow (OPF) with tighter line constraints led to a higher advantage of the proposed method. The approach led to more consistent forecast performance and reduced error variance by up to 70%.

1. Introduction

The renewable energy transition challenges power system control, power transportation and storage, market mechanisms and affordability. The energy output of most renewable energy (RE) sources cannot be arbitrarily controlled but instead relies on exogenic factors such as wind speed or solar radiation. With the increasing prevalence of RE, it becomes increasingly important to accurately predict these exogenic meteorological conditions. As electric energy cannot be stored in the grid itself and large-scale storage capacity is limited, electric power generation and consumption should ideally balance at all times. To ensure this energy balance, in Germany, compensations costs for wind power curtailment (caused by market or grid conditions) amounted to €373.7 m for the year 2016, more than the cost of re-dispatch or reserve plant costs, making up 43% of the total congestion management costs [1]. Furthermore, up to 11% or €41.1 m are caused by prediction inaccuracy of RE injections [1]. Hence, an accurate prediction of the uncertain RE power injection is required to reduce the cost of electric power generation.

Weather predictions are outputs from statistical or physical models [2] that aim at optimising the prediction accuracy. Physical models are mainly based on the weather research and forecasting (WRF) model, requiring an initial atmospheric state. Subsequently, the Monte Carlo approach samples the uncertain state from an assumed normal distribution of wind or pressure. Then, the WRF model considers the sampled state and physical models for atmospheric physics, including cloud parametrisation, land-surface models, atmosphere–ocean coupling and broad radiation [3]. Compared to these physical models, statistical models are trained from weather observation data with machine learning (ML) to learn the relationship between previous weather features and future weather. There, various ML approaches exist to reduce noise in data, handle outliers, detect meaningful features, memorise patterns and learn nonlinear relationships [4], for example, to predict solar power using ensemble architectures [5] or generative adversial networks [6]. Such models are especially relevant in decentralised scenarios e.g. for smart grids [7]. A primary ML model type is the feed-forward neural network (NN), which is a non-linear function of layered (connected) activating functions (e.g., sigmoid or tanh). When applied to weather prediction, effective NN models consider ‘pooling’ layers such as ‘MaxPool’ or ‘AveragePool’ layers that reduce the data

* Corresponding author.
E-mail addresses: dariush.wahdany@rwth-aachen.de (D. Wahdany), c.schmitt@iaew.rwth-aachen.de (C. Schmitt), j.l.cremer@tudelft.nl (J.L. Cremer).

https://doi.org/10.1016/j.epsr.2023.109384
Received 2 November 2022; Received in revised form 17 March 2023; Accepted 2 April 2023
Available online 12 April 2023

Nomenclature

Neural network

γ	Training step size
$\hat{y}$	Predicted output
$\nabla_w L(y, \hat{y})$	Training loss gradient
$L(y, \hat{y})$	Training loss function
$P(w)$	Predicting model
$p(x, w)$	Prediction function
w	Weights and biases
x	Observation input data
y	Observed output

Optimal power flow

α_g	Linear generation costs
χ	Optimisation problem variables
$\mathcal{Z}$	Line reactances
ϕ	Phase angle variable
θ	Constant optimisation problem parameters
B	Set of buses
c	Optimisation objective
$f_{\hat{y},\theta}(\chi)$	Optimisation problem objective function
G	Set of generators
$g_{\hat{y},\theta}(\chi)$	Optimisation problem constraint functions
M	System base
p^G	Power injection variable
p^{GMax}	Generator limits
v	Active line flow variable
v^{LMax}	Line Limits

End-to-end learning

$\nabla_w p(x, w)$	Wind power prediction gradient
$\nabla_{\hat{y}} s(\hat{y}, \theta)$	System cost gradient
A, D, G, Q, b, h, z	Generic optimisation problem descriptors
$s(\theta; \hat{y})$	Optimised system costs
$S(p)$	Solution mapping

Wind training

σ	RBF standard deviation
$\Phi(X)$	RBF
m	RBF centre

Hyper parameter optimisation

n	# combinations
R	Maximum epochs
r	Minimum epochs

complexity. There, a particular NN model type is a convolutional neural network (CNN) that is effective for predicting a time series of parameters such as the future weather condition [4,8–10]. The strength of CNNs is that they are time-shift invariant and resistant to noise. NNs for wind prediction also consider some recurrent components such as gated recurrent units (GRUs) layers for memorising patterns over the time domain [8]. Specialised activation functions such as the double radial basis function (RBF) can increase noise resistance and improve learning of the highly nonlinear correlations in weather predictions [9]. The model output, i.e. the weather prediction, is generic for their application purpose. In power systems, weather prediction is, for example, applied to optimise energy production schedules, market bidding, or

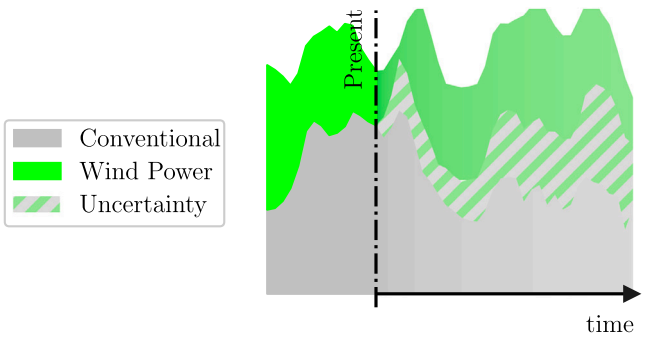
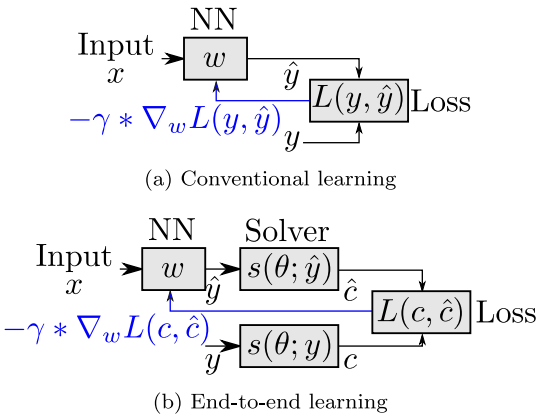


Fig. 1. Deciding on energy generation schedules for conventional power plants involves predicting wind power to match demand. The prediction typically aims at minimising the error and the uncertainty homogeneously without consideration for system cost. This work trains the prediction model directly to minimise sub-optimal scheduling cost.

security congestion management. However, the physical and statistical models aim to predict accurately, which does not directly match the aforementioned power system objectives. This mismatch of the prediction model and power system objectives can lead to suboptimal results, as mentioned above. Addressing this mismatch is actively and recently researched in operations research and machine learning. For example, [11] proposed learning a decision-focused loss function that improves the prediction model to relate better with the objective. This paper aims to design the prediction models with the intended objective of the application for wind power prediction. This paper examples this aim on wind power forecasts so that the forecasts can reduce wind curtailment costs.

Recent end-to-end approaches to training ML models consider an application model directly 'in the training loop'. The idea is to model the application task, i.e. an optimal power flow (OPF) optimisation problem, for which the prediction ML model will be applied directly during the training of the ML model [12], instead of afterwards as most conventional training approaches. The ML training loss function includes variables from that (convex-) optimisation where the solution must fulfil the Karush–Kuhn–Tucker (KKT) conditions [13,14] and can consider equality (and inequality) constraints [15] and even some combinatorics [16], or folded optimisation also to consider nonconvex problems [17]. The implicit function theorem can be applied to compute the loss gradients from the optimised solution to each ML model parameter. Then the approach updates the parameters with descending gradients. In other words, end-to-end learning directly considers the final application task and learning from observation data as in conventional ML. Directly considering this task loss has the advantage of learning better prediction models for a specific task instead of a generic prediction model. For example, the task can be to schedule energy generation by predicting energy loads [18] or by nowcast photovoltaic power outputs from multiple different input data modalities [19]. However, the disadvantage is that each training 'loop'/iteration includes solving an optimisation problem, which renders the efficient formulations of the convex optimisations necessary, e.g., as a disciplined parametrised program (DPP) [20].

This paper investigates whether end-to-end learning can reduce wind curtailment costs and proposes a specialised NN structure for wind power prediction that efficiently and directly models the task of applying the prediction to system energy optimisation during end-to-end training. The motivation for this investigation is that predicting wind power accurately during times of high energy costs may reduce energy system costs more than predicting accurately during low-cost hours. Furthermore, the proposed specialised NN structure learns systematically as a mapping of how to predict wind power at what times (e.g., when and where to over or underestimate the prediction). The two contributions are, therefore:



(a) Conventional learning

(b) End-to-end learning

Fig. 2. Overview of conventional vs end-to-end learning. Blue marks the flow of information from the gradient. In (a), learning occurs concerning errors in the intermediary value $\hat{y}$, while in (b), learning is conducted with respect to the optimisation output error $\hat{c} = s(\theta, \hat{y})$.

1. The proposed NN structure is at the same time trainable end-to-end and specially designed for wind power prediction consisting of GRU layers, 'pooling' layers, RBF activation functions, and adaptive normalisation.
2. The proposed algorithm for training this NN has two steps. The initial step trains the NNs for wind power prediction. The second training step models the OPF problem as convex optimisation end-to-end in the (training-) loop. These two steps improve the NN wind power predictions for energy system cost minimisation subject to constraints of the power network.

The approach is investigated on historical weather data sets from on- and offshore locations in Germany and Netherlands, where the IEEE-6 bus power system represents a power system. The approach was implemented in the open-source Python module *CvxpyLayers* [21] to connect the proposed NNs learning with the optimisation. The case studies focus on the economic impact on the system level, the curtailed wind, sustainability and computational training efforts.

The remaining paper is structured as follows. Section 2 describes the first contribution, an effective wind power prediction end-to-end learning approach, Section 3 describes the second contribution, the proposed NNs structure for wind power prediction. Section 4 is the case study and Section 5 concludes this paper.

2. End-to-end learning for wind power

Wind power prediction is often the first step in a sequence of steps. The second step can be the scheduling of energy resources. An issue with this sequence is that it results in suboptimal energy schedules, which is what this paper addresses through end-to-end learning (see Fig. 2). In the first step, the prediction of wind power injection can be considered as a time-series modelling problem, where the day-ahead forecast $\hat{y}$ is predicted from the observation data x available at the present time of prediction as shown in Fig. 1. This predicting model P can be a NN that considers the prediction function described by parameters w (weights and biases). Such a NN is then defined as

$$P(w) : \hat{y} = p(x, w). \tag{1}$$

To obtain a prediction, the forward-pass through the 'trained' function $\hat{y} = p(x)$ processes the input x. Conventionally, as shown in Fig. 2(a), this function is trained in a 'supervised' way using a training data set (X, Y) of i actual (x, y) observed combinations. There, the training loss function L typically quantifies the deviations of y and $\hat{y}$

$$L(y, \hat{y}) = \frac{1}{n} \sum_i (y[i] - \hat{y}[i])^2 \quad y, \hat{y} \in \mathbb{R}^n, \tag{2}$$

where $n \in \mathbb{N}$ denotes the dimension of the output values. This NN model is trained considering the actual value of the predicted variable $\hat{y}$. The loss function $L(y, \hat{y})$ is therefore based on the difference between the actual wind power y and the predicted wind power $\hat{y}$. A gradient-descent-based optimiser will update the parameters w according to the loss gradient concerning the parameters $\nabla_w L(y, \hat{y})$. The update formula is, therefore

$$w_{t+1} = w_t - \gamma * \nabla_w L(y, \hat{y}) \tag{3}$$

where γ is the step size. Therefore, the training aims to reduce the (absolute) difference of y and $\hat{y}$.

In the second step, the predicted output of wind power injections $\hat{y}$ are often used in some further applications within energy systems, for example, the optimisation of an energy scheduling plan described through the general optimisation

$$\min_{\chi} \quad f_{\hat{y},\theta}(\chi) \tag{4a}$$

$$\text{subject to} \quad g_{\hat{y},\theta}(\chi) \leq 0, \tag{4b}$$

where $g(\chi)$ can include all power network equality and inequality constraints, such as AC (or DC) power flow equations considering network connectivity, energy node balances, and capacity limits. There, the parameters θ are stationary parameters defining the functions f and g, such as the topology of the power network or parameters defining the physical limits of transmission lines.

An issue with running these two steps in a sequence is that the objectives (the optimisation problem objective and the training loss) of these two steps are not aligned. In the first step, the NN learns to minimise a loss function based on previous observations. However, this aim is unfortunately not directly aligned to the second step objective for the scheduling optimisation Eq. (4a). As a result of the missing alignment of the two objectives, in the second step, the optimisation determines suboptimal energy scheduling plans as the intermediate prediction was not aligned with its objective. This sub-optimality can result in unnecessary high wind curtailment and inefficiencies.

The proposed end-to-end approach is integrative, not sequential, to address the aforementioned sub-optimality issue. As shown in Fig. 2(b), the NN directly considers the objective function of optimisation Eq. (4a) as training loss function. The proposed approach considers directly in the training loop the optimisation objective $c = f_{y,\theta}(\chi)$ (or another function based on the optimised solution) as a loss function $L(c, \hat{c})$ to train the NN. Hence, the NN $P(w)$ learns to predict $\hat{y}$ in such a way that it minimises the error regarding objective c of the optimisation Eq. (4a) instead of minimising concerning the standard loss Eq. (2).

2.1. Energy system scheduling

The objective function $f(\chi)$ in Eq. (4a) may be the economic operating cost of the power network. These functions $g(\chi)$ and $f(\chi)$ are defined, among other parameters θ, by the predictions of wind power $\hat{y}$, and they represent an energy scheduling problem over multiple time periods. There, this paper uses exemplary the linear DC load flow formulation to approximate the AC power flow [22]. Aside from the linear DC load flow formulation, several other advances to solve the nonlinear and nonconvex AC optimal power flow (OPF) were made to solve this in reasonable times [23,24]. Such approaches can decompose, relax, or reduce the problem in various forms. However, this work deploys a basic DC formulation that is suitable for transmission grids and based on the assumptions of constant voltage (at 1 p.u.) between buses, line reactances significantly bigger than line resistances, and small differences in phase angles ϕ. Thus, only active power flows (and phase angles) are calculated and no power losses are considered. In the optimal DC OPF problem, the objective function may consider the linear cost of generation

$$f_\theta(\chi) = \sum_t \sum_g \alpha_g p^G_{g,t}, \tag{5}$$

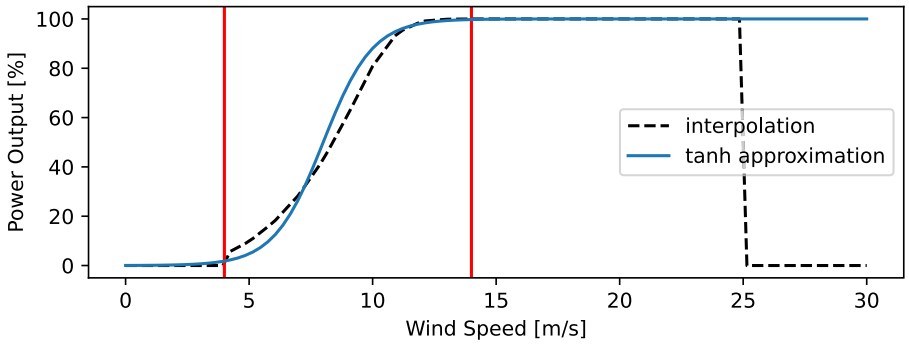


Fig. 3. Modelling of wind power injection.

where α_g are the linear cost components for the generators g with power injection $p^G_{g,t}$ in time interval t. In other variations of the OPF, this function can also be quadratic or other forms and has been selected as linear for simplicity reasons. The variables $\chi = \{p^G_{g,t}, \phi_{b,t}, v_{b,b',t}\}$ of the optimisation are the power injections $p^G_{g,t}$, phase angles $\phi_{b,t}$ of the buses b and line flows $v_{b,b',t}$. The following describes the set of inequality and equality constraints $g(\chi) \leq 0$. By using the load reference-arrow system ($p^L_l \geq 0$; $p^G_g \leq 0$), this paper can formulate the node balance for each bus b in each time interval t based on the generated power injection $\sum_{g\in G_b} p^G_{g,t}$ and the nodal load $\sum_{l\in L_b} p^L_{l,t}$ and the flows over lines connected to the bus b as follows

$$\sum_{l\in L_b} p^L_{l,t} + \sum_{g\in G_b} p^G_{g,t} + \sum_{b'\in B^L_{b,b'}} (v_{b',b,t} - v_{b,b',t}) = 0 \quad \forall b,\ t. \tag{6}$$

There, the non-symmetric set $B^L_{b,b'}$ expresses the line connectivity of the network and defines the direction of power flow, e.g. $B^L_{b,b'} = 1$, if the corresponding power of the line connecting b and b' is transmitted from bus b to b'. The power flows between two nodes b and b' are

$$v_{b,b',t} - \mathrm{M}\frac{\phi_{b,t} - \phi_{b',t}}{\mathcal{Z}_{b,b'}} = 0 \qquad \forall b, b' \in B^L_{b,b'},\ t \tag{7a}$$

$$-\ v^{LMax}_{b,b'} \leq v_{b,b',t} \leq v^{LMax}_{b,b'} \qquad \forall b, b' \in B^L_{b,b'},\ t, \tag{7b}$$

where the parameters are the line reactances $\mathcal{Z}_{b,b'}$ and the factor M represents the system base [MVA p.u.]. The variables are the phase angles $\phi_{b,t}$ and the line flows $v_{b,b',t}$. Additionally, each line flow $v_{b,b',t}$ is limited by the maximum thermal rating $v^{LMax}_{b,b'}$. The main flexibility of the considered power system is based on the flexible generation of renewables and thermal power plants. The renewable generation is denoted by the subset G^{REN} of all generation, is time-varying and can be curtailed

$$0 \leq p^G_{g,t} \leq p^{GMax}_{g,t}. \qquad \forall g \in G^{REN},\ t \tag{8}$$

The optimisation run near to real-time t could be used to control a wind farm $p^G_{g,t}$, where the maximal available wind power is $p^{GMax}_{g,t}$. For example, to avoid congestion in the grid, the power from the wind farm would be set $p^G_{g,t} < p^{GMax}_{g,t}$. However, conventional power plants are only limited by upper and lower bounds of generation

$$p^{GMin}_g \leq p^G_{g,t} \leq p^{GMax}_g. \qquad \forall g,\ t \tag{9}$$

For thermal power plants, start-up and shut-down costs are considered with a more complex linear formulation as presented in [25,26], thus expanding the generation constraints Eq. (9) and the objective Eq. (5).

2.2. End-to-end training of neurons

In the introduced task, the predicted wind speed $\hat{y}$ is to derive the generated power $\hat{p}$, which subsequently is used to determine the predicted system cost $\hat{c}$ of the next day. However, it might not be relevant to obtain the most accurate wind speed $\hat{y}$ or wind power prediction $\hat{p}$, as long as the system cost prediction $\hat{c}$ is accurate. Hence, our loss on the task is actually $L(c, \hat{c})$. The wind power is derived using the wind turbine power/speed curve shown in Fig. 3. To calculate the 'true' wind power, we perform a numerical interpolation including low- and high-speed cut-offs. When obtaining the wind power prediction during NN training, we approximate this interpolation with a tanh function fitted to the interpolation curve. The parameters θ correspond to fixed parameters in the power system that are not learned, such as the grid topology, system loads, power plant settings. As illustrated in Fig. 4, the cost is therefore an output variable of the solver that minimises Eq. (4a), leading to equations

$$c = s(\theta; y) \tag{10}$$

$$\hat{c} = s(\theta; \hat{y}). \tag{11}$$

where

$$s(\theta; \hat{y}) = f_{\hat{y},\theta}(\chi) \tag{12}$$

at the minimum found by the solver for the problem in Eq. (4a) with constraints from Eq. (4b). If the prediction error could be eliminated, then, these losses would equal and amount to zero. However, in practice, there will always be some residual forecast error and it is not obvious, whether the set of NN parameters

$$w_y = \arg\min_w L(y, \hat{y}) = \arg\min_w L(y, p(x, w)) \tag{13}$$

that minimises the forecasting error in y is equal to the set of parameters

$$w_c = \arg\min_w L(c, \hat{c}) = \arg\min_w L(c, s(\theta; p(x, w))) \tag{14}$$

that minimise the forecasting error in c. Here, the system cost c is the result of a mathematical optimisation solver $s(\theta; y)$ and the power system is parametrised in θ, leading to the hypothesis

$$w_y \neq w_c \quad , \forall\, L > 0. \tag{15}$$

Hence, as t is not possible to totally reducing the prediction error to zero, this paper calculates the loss directly with respect to the task loss $L(c, \hat{c})$ that minimises the error in the system cost prediction. The update formula from Eq. (3) for our NN parameters then changes to

$$\begin{aligned} w_{t+1} &= w_t - \gamma * \nabla_w L(c, \hat{c}) \\ &= w_t - \gamma * \nabla_w L(c, s(\theta; \hat{y})) \\ &= w_t - \gamma * \nabla_w L(c, s(\theta; p(x, w))) \end{aligned} \tag{16}$$

This type of learning constitutes end-to-end learning as the model is trained with regard to the actual task and not some intermediate value.

For a perfect prediction, the task loss is equal to zero both in the end-to-end and the conventional learning setting. Additionally, the power system objective value of the true wind feed-in is a lower bound for the predicted wind power. Thus, any increase in forecasting accuracy always improves the power system objective and simultaneously

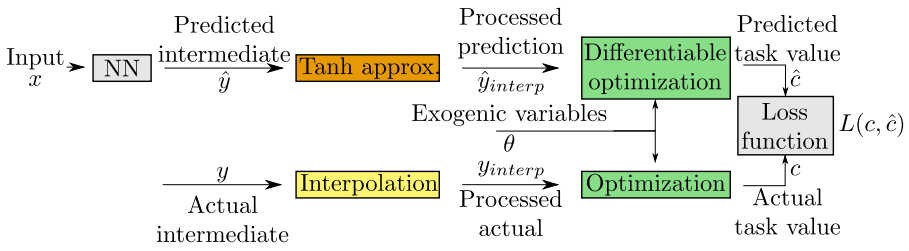


Fig. 4. End-to-end learning.

the end-to-end. The advantage of end-to-end learning lies in adjusting the forecasting accuracy with respect to the sensitivity of the power system objective. Any change in the power system objective function (that is required to be convex) alters this sensitivity and therefore refocuses the forecast learning.

2.3. End-to-end training of wind power neurons

Unwrapping the gradient-descent update formula in Eq. (16) leads to an expression for the weight difference

$$\begin{aligned}\Delta w &= w_{t+1} - w_t \\ &= -\gamma * \nabla_w L(c, s(\theta; p(x, w))) \\ &= -\gamma * \nabla_w p(x, w) * \nabla_{\hat{y}} s(\theta; \hat{y}) * \nabla_{\hat{c}} L(c, \hat{c}).\end{aligned} \tag{17}$$

Updating our weights w therefore requires

- $\nabla_w p(x, w)$: gradient of the predicted wind power to the input data,
- $\nabla_{\hat{y}} s(\hat{y}, \theta)$: gradient of the system cost with respect to the predicted wind power,
- $\nabla_{\hat{c}} L(c, \hat{c})$: gradient of the loss with respect to the predicted cost

Obtaining the first and last gradient is straightforward as calculating the NN output and loss function gradients are well-known techniques. However, obtaining $\nabla_{\hat{y}} s(\hat{y}, \theta)$ is not straightforward. The wind power prediction $\hat{y}$ is part of the optimisation problem parameters, and its problem solution $s(\hat{y}, \theta)$ is generally calculated numerically. That means an analytical and, therefore differentiable mapping from $\hat{y}$ to $s(\hat{y}, \theta)$ does not necessarily exist. Instead, this paper uses the implicit function theorem to implicitly differentiate the KKT conditions that solutions to convex optimisation problems must fulfil.

The KKT conditions are necessary conditions for optimal solutions of nonlinear optimisation problems. For a cone program of the form

$$\begin{aligned}&\underset{z}{\text{minimise}} \frac{1}{2} z^T Q z + q^T z \\ &\quad \text{s.t.} Az = b, Gz \leq h\end{aligned} \tag{18}$$

The equations for stationarity, primal feasibility and complementary slackness are

$$Qz^* + q + A^T v^* + G^T \lambda^* = 0 \tag{19}$$

$$Az^* - b = 0 \tag{20}$$

$$D(\lambda^*)(Gz^* - h) = 0 \tag{21}$$

and must be fulfilled at valid solutions of the optimisation problem. The stationarity condition (Eq. (19)) describes that the solution must minimise the objective function. The primal feasibility condition (Eq. (20)) holds true for all values that lie within the problem constraints. The complementary slackness condition (Eq. (21)) tells us that the inequality conditions of the dual problem are binding for non-zero solution variables in the primal or vice-versa. These conditions can (for the most part) be implicitly differentiated around solutions found by a solver in regards to any data point, e.g. a constraint value.

There, the implicit function theorem states that if

$$f : \mathbb{R}^d \times \mathbb{R}^n \rightarrow \mathbb{R}^n \text{ is continuously differentiable}$$

$$\text{in a neighbourhood of } (\overline{p}, \overline{x}) \tag{22}$$

$$f(\overline{p}, \overline{x}) = 0 \tag{23}$$

$$\nabla f(\overline{p}, \overline{x}) \text{ nonsingular,} \tag{24}$$

then, the solution mapping

$$S(p) = \{x \in \mathbb{R}^n | f(p, x) = 0\} \tag{25}$$

can be continuously differentiated in a neighbourhood Q of $\overline{p}$ with

$$\forall_{p \in Q} : \nabla s(p) = -\nabla_x f(p, s(p))^{-1} \nabla_p f(p, s(p)) \tag{26}$$

In this paper, the relevant constraint value to differentiate around is the wind power injection, which is part of the node balance constraints. Hence, they are part of the optimisation constraint vector b and obtaining $\frac{\delta z^*}{\delta b}$ allows us to successfully back-propagate errors in system cost to the prediction of wind power injection [13,20,27].

To programmatically implement a differentiable optimisation layer, previous work introduced DPP as a subclass of convex problems for which an automatic differentiation of the output with respect to parameters can be obtained. The difference between DPP and all convex problems lies within the mathematical atoms for which the curvature can be determined. The DPP ruleset does not consider products of two variable-dependent expressions as convex and hence will not accept objective functions or upper bounds depending on them, even if the expression is actually convex. For this and some other examples, libraries such as CVXPY typically provide functions that are manually marked as convex such as $entr(x > 0) = -sum(x \cdot log(x))$ (even though using $-sum(x \cdot log(x))$ directly would lead to a DPP compliance error due to unknown curvature) that can be used instead. Aside from the curvature determination, all convex problems are by definition DPP compliant [21,27].

3. Neural networks for wind power forecasts

This paper proposes two different NN-models, the *recurrent model*, *RBF model* for predicting the wind power injection from historic meteorological measurements and a tailored algorithm for optimising hyperparameters in end-to-end learning.

3.1. Neural network model structures

The *recurrent model* uses recurrence to calculate the predictions while the *RBF model* involves specialised activation functions and advanced convolutional filtering [4,8–10]. These two models have common hyperparameters (learning rate, batch size and probability for the dropout layers) and the following differences.

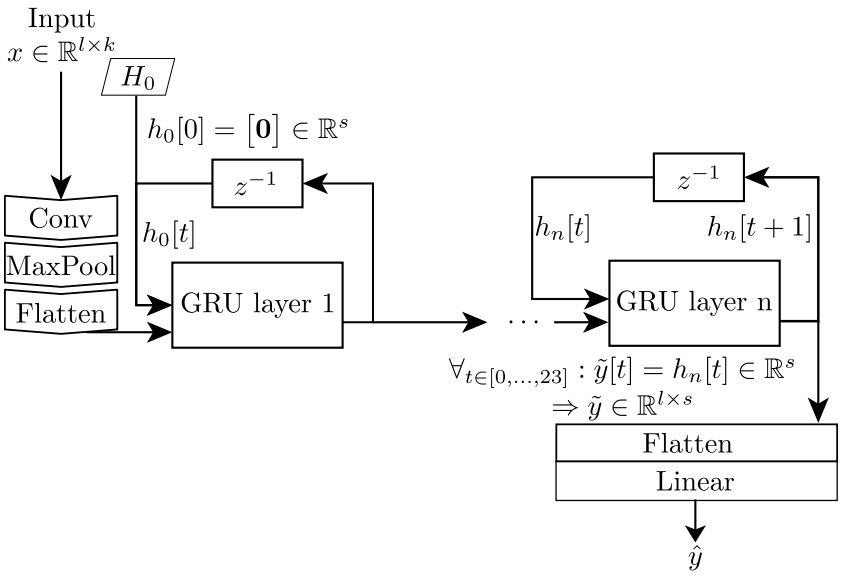


Fig. 5. *Recurrent model* to predict wind power.

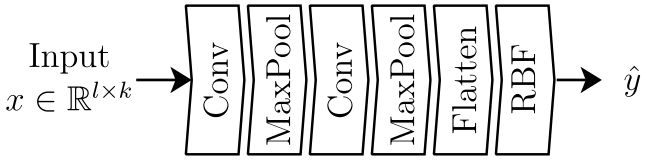


Fig. 6. *RBF model* to predict wind power.

Recurrent model. The first NN structure is a hybrid model consisting of GRUs and fully connected (FC)-layers based on the approach in [8] (in the following *recurrent model*). A GRU provides the ability to detect time-domain patterns and utilise them for the prediction. GRUs are originally intended for text processing but since have been used for many applications and are part of the standard ML repertoire [28]. This NN structure focuses on recurrent data processing for which it uses n GRU layers, each with hidden size s. Only one initial convolutional stack is present in the recurrent network. Fig. 5 depicts the layout of the recurrent model. The input $x \in \mathbb{R}^{l \times k}$ is first fed through a convolutional layer with MaxPool and flattened. A zero-initialised ($H_0 = 0$) GRU with n layers processes the flattened data before a final flatten and linear layer computes the output.

RBF model. The second NN model, the *RBF model* does not use recurrent structures and instead relies on trainable activation functions in the last layer to generate the output for different hours based on [9]. To improve the convergence of the NN exponential linear unitss (ELUs) calculate intermediate activations [29] for both networks. Following prior research, the RBF model has a two-layer deep convolutional feature extractor while the recurrent model uses one. The reasoning behind this difference is that the GRUs provide enough complexity to extract and learn meaningful features, while the simpler structure of the RBF model requires conjunction with a more complex feature detector. A layout of the RBF model is shown in Fig. 6. Two convolution/pooling stacks process the k-dimensional input x with length l and flatten it. The resulting one-dimensional vector is FC to 24 RBF-neurons that uses the function

$$\Phi(X) = \frac{1}{\sqrt{2 * \pi * \sigma^2}} * e^{-\frac{(X-m)^2}{2\sigma^2}}, \tag{27}$$

where σ is the standard deviation and m is the centre. As the input and output of wind prediction have a high underlying uncertainty while having a non-linear relationship, a double Gaussian function is used. The double Gaussian approach enables highly non-linear output dependencies while providing a 'dead zone' in which minor deviations or noisy measurements do not impact the output. The output is calculated as the combination of upper and lower Gaussian function $\overline{\Phi}$ and $\underline{\Phi}$. Their corresponding equations are

$$\overline{\Phi}(x) = \begin{cases} \Phi(m_1, \sigma_1; x), & x < m_1 \\ 1, & m_1 \le x \le m_2 \\ \Phi(m_2, \sigma_2; x), & x > m_2 \end{cases} \tag{28}$$

$$\underline{\Phi}(x) = \begin{cases} \Phi(m_1, \sigma_1; x), & x \le \frac{m_1+m_2}{2} \\ \Phi(m_2, \sigma_2; x), & x > \frac{m_1+m_2}{2} \end{cases} \tag{29}$$

where m_1, m_2, σ_1 and σ_2 are unique learnable parameters per neuron, resulting in $4 \times 24 = 96$ parameters for 24 outputs. The output is the average of the upper and lower Gaussian

$$y(x) = \frac{\overline{\Phi}(x) + \underline{\Phi}(x)}{2} \tag{30}$$

This output of the RBF layer is FC to the output layer without another non-linearity, hence, 1 equals the identity function $f(X) = X$.

3.2. Efficient end-to-end training by optimised hyperparameters

This section proposes the training algorithm that involves hyperparameter optimisation (HPO) for end-to-end learning of wind-power predictions, as illustrated in Fig. 7. The algorithm involves three steps, the data preparation, the training of the NN itself, and the HPO.

The first step generates the training data from the load, wind power and weather data. Then, a numerical optimisation completes a training data sample by computing the actual system cost c to generate (x, y, c) combinations.

The second step takes an assigned hyperparameter combination to train the model. The differential optimisation described in Section 2 uses this prediction to minimise the predicted system cost $\hat{c}$. The output of this step is a trained model $P(w)$ with the model parameters w. The performance of the model $P(w)$ depends highly on the assigned hyperparameters, and assigning values for hyperparameters is not straightforward. In this work, many hyperparameters need to be assigned, including the choice of the optimiser and its parameters (e.g. learning rate, weight decay) and parameters regarding the training process, such as batch size. In addition to these, for our networks, the hyperparameters include the filter-, kernel- and pool size of the

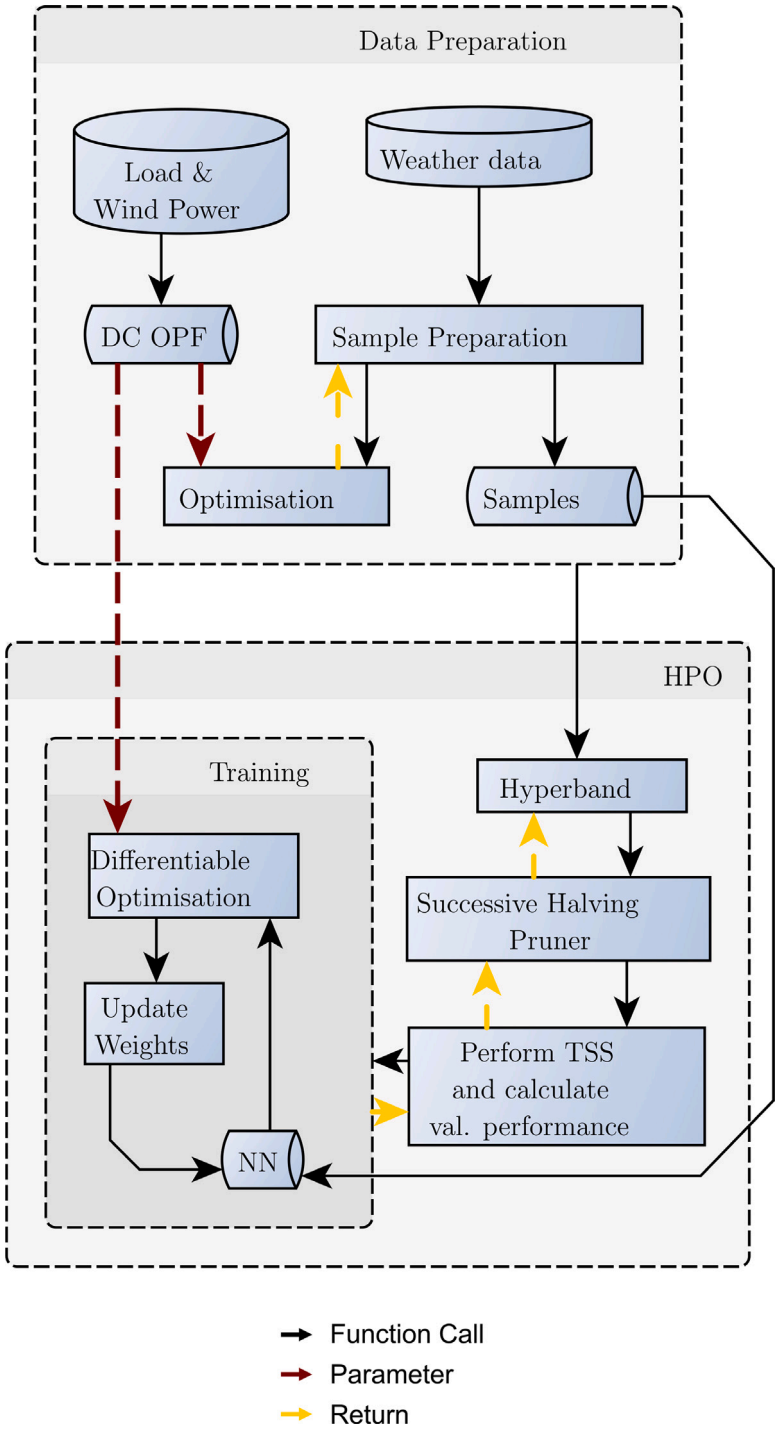


Fig. 7. Overview of the training process.

convolution layers and the number and size of the GRU and linear layers.

The third step, the HPO, optimises these many hyperparameters automatically using various approaches. Fig. 8 shows two typical approaches to assign hyperparameter values, which are the *random-search* that samples their values randomly uniformly distributed or the *grid-search*. Random search often performs better than grid search; typically, the different hyperparameters are unequally important. However, these approaches are unsuitable for end-to-end learning since it is unfeasible to train many models where each forward pass includes a computationally expensive numerical optimisation. Therefore, reducing this number of combinations is especially relevant for end-to-end learning, where the computational complexity of training a model is very high; hence, the fewer models that need to be trained, the better.

In response to the challenge in step three, this paper proposes using a pruner for HPO of end-to-end learning. There, *successive halving* uses n different combinations and trains them for at least r epochs [30]. After that, the worst half of the combinations stop training and are omitted (halving). The training and halving continue until the number of epochs reaches the maximum R. An example of this for $n = 8$, $r = 2$ and $R = 8$ is given in Fig. 9. Successive halving, therefore, requires three parameters n, r and R. *Hyperband* is a pruning strategy comprised of multiple differently configured successive halvings. The strategy balances the exploration of many combinations with decreased

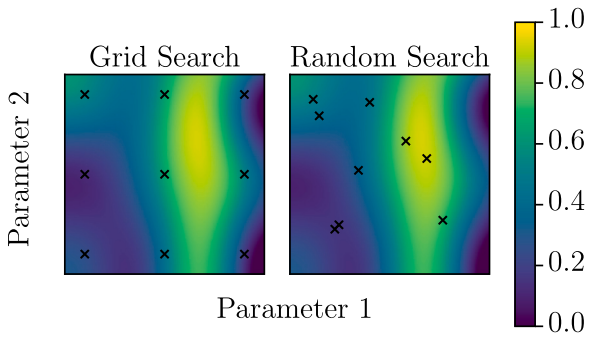


Fig. 8. Two different HPO strategies, grid and random search, and the model performance after being trained on the corresponding hyperparameters. Yellow is high performance and blue is low performance.

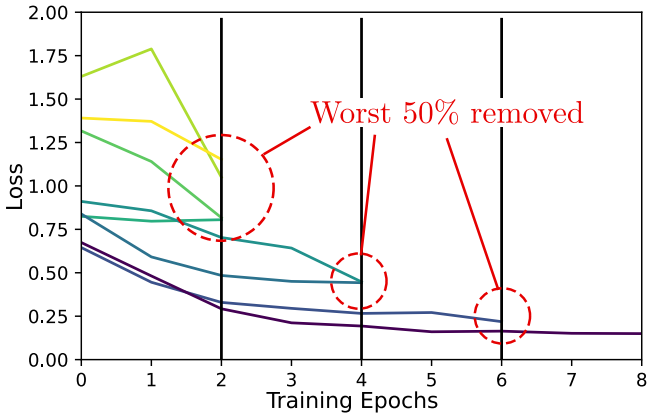


Fig. 9. Successive Halving Pruner for HPO.

uncertainty of specific combinations by utilising multiple successive halvings. This strategy varies the parameters from a large n combined with small r and R (exploring many combinations) to large r and R for small n (reducing the uncertainty of few combinations). Eventually, one successive halving with large $r = R$ and small n is initialised, essentially one random search run. This strategy can speed up the Hyperband's convergence over random search in the range of 6× to 70× [30].

4. Case study

The case studies focus on the possible performances of the algorithm regarding reductions in economic cost, improvement of sustainability, reductions in grid congestion, and computational training efficiency.

4.1. Test settings

The case studies test the proposed algorithm on 20 years of historical ERA5 weather data for two locations, an onshore location in Germany and an offshore location in the Netherlands [31]. The weather time series are converted into wind power injection potentials using typical wind turbine characteristics. The maximum output of the wind park relative to the maximum electric energy system load

$$p^{Wind} = \frac{\text{installed wind power capacity}}{\text{Maximum load}} \tag{31}$$

varies between $p^{Wind} = 33\%$, 67% and 100% to simulate possible energy future scenarios. This paper uses two variations of the IEEE 6-bus system shown in Fig. 10. First, a study on the unmodified IEEE 6-bus system is performed. Secondly, the IEEE 6-bus system is modified to consider power network constraints. This paper calls the *normal system* a system implemented as shown in Fig. 10(a). This paper calls a *congested system* a modified system where the wind power injection connects to the same node but through a limiting line as shown in Fig. 10(b). This paper assumes this line is congested and set the transport limit at half the maximum wind power injection capacity.

Table 1
Software packages.

Purpose	Software	Version
Main solver	SCS	2.1.4
Solver for comparison	GUROBI	9.5.1
Solver for comparison	ECOS	2.0.10
GPU acceleration	ROCm	4.3.1-1
ML framework	PyTorch	1.10.0
ML Meta-Framework	PyTorch Lightning	1.4.0
HPO	Optuna	2.8.0

When learning the NN models, the optimiser AdamW was selected. The training and testing split is performed using Time Series Split (TSS). TSS splits a historical dataset sorted by time into five folds, where each fold is a super-set of all previous folds. Each fold is split into training and testing shares. There, the validation subset of each fold is subsequent data to the training subset. Five folds of 80% training and 20% validation data each were used. The last fold was used for testing, meaning the validation-tuned model was trained on 80% of the total historical data, and the 20% most recent data is the hold-out test set. All studies were repeated nine times, including HPO. The *RBF model* structure includes two convolutional layers with 19 and 28 channels and kernel/max-pool sizes 6×1 and 7×1 respectively. A dropout with $p^{Dropout} = 0.2$ is consecutive to the last convolution. The fully-connected layer has 1428 input neurons, which feed into 24 RBFs. The *recurrent model* structure includes a single convolutional layer with 19 input- and 57 output-channels, a dropout with $p^{Dropout} = 0.46$ follow by a three-layer GRU with a hidden size of 20. Two fully-connected layers compute output activations, with 17,480 and 456 weights, which feed into a randomised leaky rectified linear unit (RReLU). For the two models, ELU calculates intermediate activations, and BatchNorm normalises the values in-between layers and activations. Weights are initialised randomly and uniformly distributed.

The same NN architecture trained on wind predictions only was used as a baseline approach. The two approaches were separately optimised for hyperparameters, including minor adjustments of the NN structure. Two computers were used for the training and will be referred to by *Computer A* and *Computer B* in the corresponding case studies. *Computer A* is equipped with an Intel i7 9700K 8 Cores @ 4.90 GHz and 24 GB of RAM. Training is accelerated using an AMD Radeon RX Vega 64 GPU with 8 GB of HBM2 memory. *Computer B* is an Apple MacBook Pro equipped with an Apple M1 Pro (8 Cores CPU, 14 Cores GPU) with 16 GB of unified memory. Table 1 displays the relevant software.

4.2. Economic performance

The objective of this case study is to investigate the performance of the prediction quality on the power system economic cost. The system cost prediction quality is also considered an indicator for the prediction quality of other cost-dependent quantities such as energy storage or flexible demand and generation in general, e.g. hydropower plants and batteries. Since the storage schedule depends on price activity, the accuracy of the predicted optimal schedule correlates with an accurate price prediction. The installed wind power capacity varies from $p^{Wind} = 33\%$, 66% and 100% of the maximum load. The baseline was the NN model trained on wind power data only. Fig. 11 shows the changes of the proposed method on the prediction output, wind power and system cost errors in the form of a histogram. The average relative changes in normalised root mean squared error (NRMSE) of system cost predictions varies for onshore and offshore data as shown in Table 2. In the onshore data from Germany, the proposed approach shows error rates

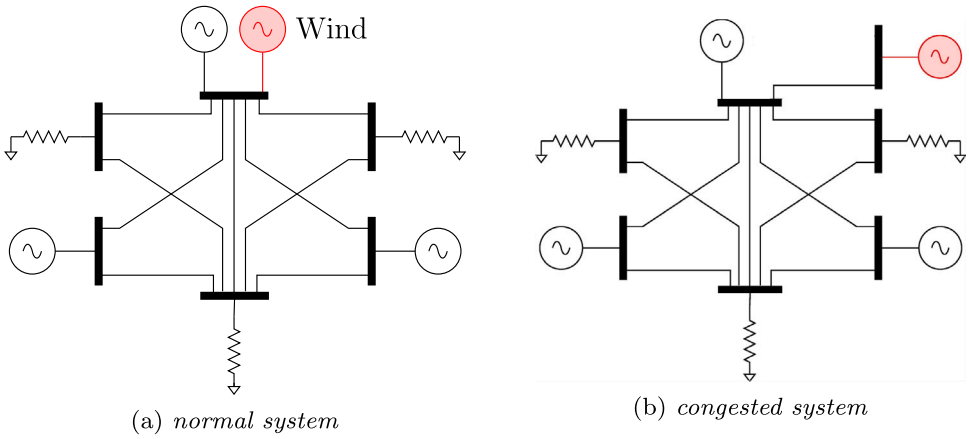


Fig. 10. Modified IEEE 6-bus test system. The wind generator in red in (a) is not connected to a congested line where in (b) this generator is connected to a transmission line that may become congested at high wind injections. This modification simulates a constraint in a congested system, where in response wind power may require curtailment. With this modification, this paper investigates whether end-to-end learning can learn the sensitivity required by forecasts to these complex congestions.

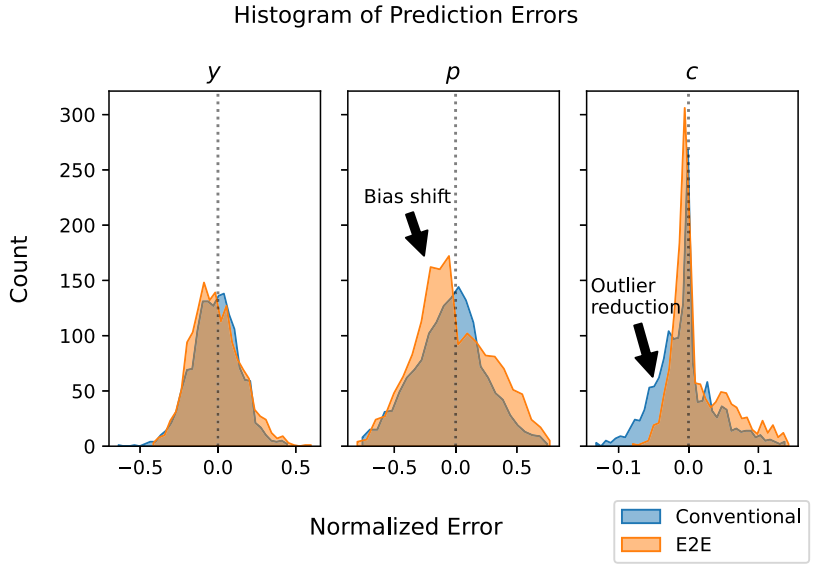


Fig. 11. Histogram of prediction errors for wind speed (y), power generation (p) and system cost (c).

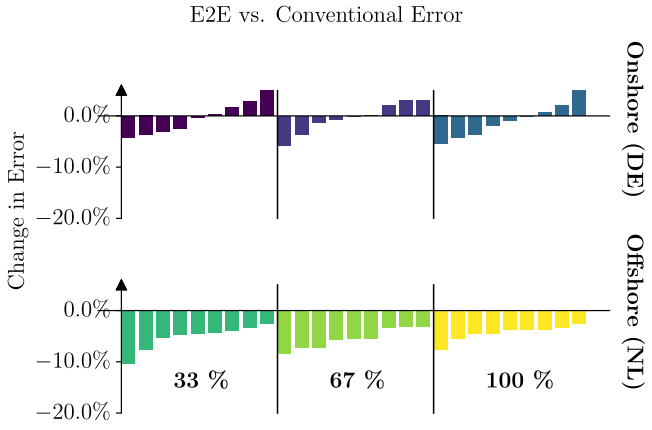


Fig. 12. Error change for the *normal system* of 9 repeated individual training processes ordered by the change of error, e.g., each bar represents one training of a NN. Three different cases of installed wind power capacity p^{Wind} = 33%, 66% and 100% in an offshore or onshore setting.

Table 2
Relative test NRMSE difference between end-to-end and conventional learning for the *normal system*.

Dataset \ p^{Wind}	System Cost Error Improvements (End-to-end - Conventional)		
	33%	66%	100%
Onshore (DE)	−0.26%	−1.30%	−0.63%
Offshore (NL)	−8.31%	−9.80%	−7.97%

reduced by 1.3% than the baseline approach. In the offshore data from the Netherlands, the proposed approach shows improvements up to 9.8%. The high variance within the data corresponds to the inconsistent performance of the baseline approach, as shown in Fig. 12. Therefore, the proposed approach generally provides more consistent (and, on average better) performance. However, despite otherwise identical system conditions, the improvements vary significantly between the two data sets (1.3% versus 9.8%) and are maximised for the medium installed capacity.

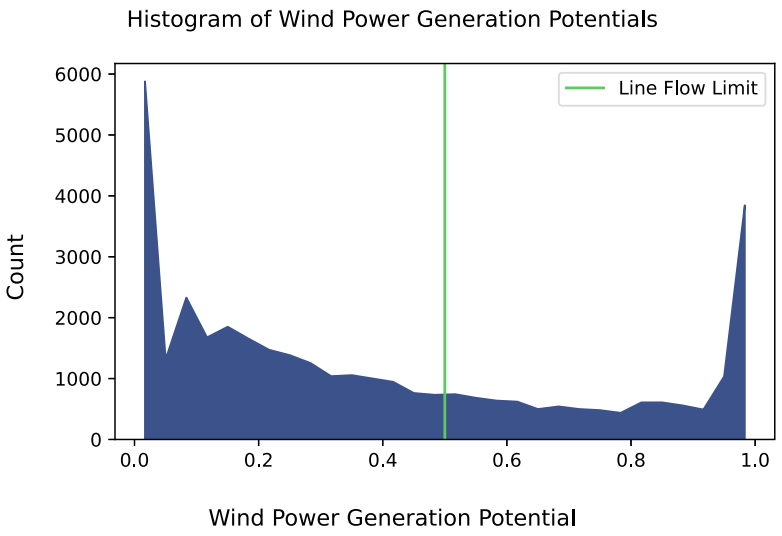


Fig. 13. Histogram of hourly wind power generation potentials and line flow limit in the congested case.

Table 3
Wind curtailment rates for the *normal system*.

Dataset	Variant \ p^{Wind}	33%	66%	100%
Onshore (DE)	Conventional	0.31%	1.89%	2.43%
	End-to-End	0.43%	2.54%	2.91%
Offshore (NL)	Conventional	0.20%	0.97%	0.50%
	End-to-End	0.45%	2.09%	1.92%

4.3. Sensitivity to grid congestion

This case study evaluates the effect of increased power grid constraints by introducing constrained lines. Instead of injecting wind power directly at its original bus, the power injection is separated and fed through a bus congested to half of the installed capacity, as shown in Fig. 10(b). This leads to a limitation in the actual power injection. Fig. 13 shows the number of hours with a certain wind power generation potential and the line flow limitation. This decreases the overall energy that the wind turbine can inject by 27%. The standard deviation of the injection decreased by 43%. The metric to assess the sensitivity to grid congestion is the change in error introduced through the congestion. The experiments were repeated nine times.

The proposed approach outperforms the conventional training in all cases, as the results in Table 4 show, albeit with only small gains on the onshore weather data from Germany. Training on offshore weather data from the Netherlands significantly improves NRMSE by around 8 to 10%. There, Fig. 14 shows the individual training processes' results which exhibit a standard deviation of around 4% for DE and 3% for NL. These results suggest that congestion adds information that the proposed end-to-end learning approach can use. These results also show that information about the congestion (or system) can amplify the effect of end-to-end learning in positive and negative directions. The changes compared to conventional learning increase in magnitude but are not necessarily better. The cases where end-to-end learning was better already have improved even more, while in the other cases where performance decreased, the performance decreased further (see Table 5).

4.4. Sustainability impact

This case study aims to evaluate the impact on the sustainability of the power system. There, the assessment metric is the reduction in expected wind curtailment caused by errors in the prediction. A lower wind curtailment rate corresponds to using more carbon-free wind energy, which is more sustainable than higher wind curtailment rates. In this study, the *normal system* and *congested system* were investigated.

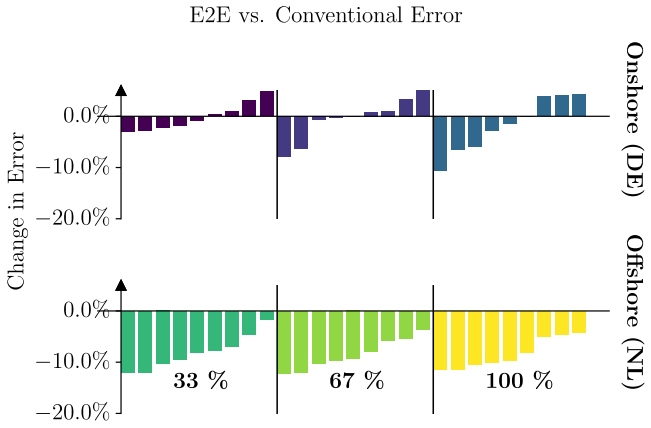


Fig. 14. Error change for the *congested system* of 9 repeated individual training processes ordered by the change of error, e.g., each bar represents one training of an NN. Three different cases of installed wind power capacity p^{Wind} = 33%, 66% and 100% in an offshore or onshore setting.

The results in Table 3 (for the normal system) and Table 5 (for the congested system) show that the curtailment rates decreased only for the offshore data set in the congested system for the very high installed wind capacity of p^{Wind} = 100%, however, in all other cases, the proposed approach results in lower sustainability in the onshore data set. These results are unsurprising as the learned objective was optimising system cost prediction and not minimising wind curtailment. Furthermore, as conventional power plants exhibit low flexibility and incur start-up costs during ramp-up, it is not always optimal to reduce their output to prevent wind curtailment. Therefore, minimising the system costs did not necessarily correspond to reducing wind curtailment.

4.5. Computational efficiency

This case study investigates the computational training times of the proposed end-to-end learning approach and their improvements when using commercial solvers for optimisations.

The first part of this study quantified the differences between proposed and conventional training by analysing the time spent in the

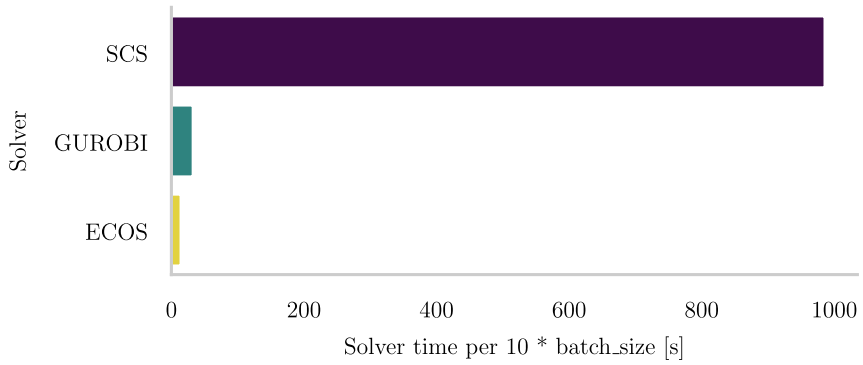


Fig. 15. Comparison of solver run-times.

Table 4
Relative test NRMSE difference between end-to-end and conventional learning for the *congested system*.

Dataset \ p^{Wind}	System Cost Improvement (End-to-end - Conventional)		
	33%	66%	100%
Onshore (DE)	−0.28%	−0.83%	−1.03%
Offshore (NL)	−8.09%	−8.50%	−8.33%

Table 5
Wind curtailment rates for the *congested system*.

Dataset	Variant \ p^{Wind}	33%	66%	100%
Onshore (DE)	Conventional	0.31%	1.92%	2.54%
	End-to-End	0.44%	2.34%	2.87%
Offshore (NL)	Conventional	0.35%	2.18%	2.47%
	End-to-End	0.44%	2.21%	1.95%

Table 6
Combined CPU/GPU time for forward pass.

Step	Conventional [ms]	End-to-End [ms]
Norm.	0.67	
Convolution	18.89	
GRU	563.96	
Linear	3.45	
Denorm.	0.61	
Loss Calculation	0.54	1,941.00
Other	0.86	10.70
Total	**588.98**	**2,539.28**

forward and backward passes. Using a batch size of 64, the forward pass time in conventional learning was 0.59 s, and in end-to-end learning was 2.51 s on *Computer A*. Table 6 shows the full breakdown of the computational times in the individual algorithmic steps. As expected, the computational time from conventional to end-to-end learning increases drastically (×4.25 fold increase) as the loss computation in the forward pass involves solving the mathematical optimisation in end-to-end learning. On the other hand, the loss computation does not involve optimisations in conventional learning, so the times are lower than during end-to-end learning.

The second part of this study investigated whether commercial solvers can reduce the significant share of the computational times that is solving the optimisations in the forward-pass. The computational times of different solvers in the forward-pass were compared on *Computer B*. The compared solvers are SCS (default of CVXPYLayers), GUROBI and ECOS, GUROBI and ECOS show in Fig. 15 significantly better performances than SCS, by a factor of >50. Therefore, when end-to-end training uses these solvers, significant reductions can be expected, as our projection shows in Fig. 16. However, currently, these solvers are not (yet) available for the package this paper used.

4.6. Discussion

The benefits of end-to-end learning wind-power forecast models are the flexibility for many possible purposes, the low error variance, and the benefits in the targeted objective, here system costs. End-to-end learning can be designed using any combination of variables, constraints and objectives, such as system cost, sustainability or optimal scheduling. For the selected learning target system costs, the approach improves over 10% in our case study. The variations in the improvement were due to inconsistent results of conventional training. The end-to-end training had, on average, 50.08% lower error variance across different training with reductions for different cases ranging from 37.96% to 69.73%. The benefits of end-to-end learning wind power predictions were the highest for our study's offshore wind time series. Interestingly, the benefits of end-to-end learning become amplified when using a congested system which may suggest that the system's complexity (purpose) may influence the learning of the model. When comparing the statistical means μ_{onshore}, μ_{offshore} and standard deviations σ_{onshore}, σ_{offshore} showed that the fact that $2 * \sigma_{\text{onshore}} \approx \sigma_{\text{offshore}}$ holds true suggests worse predictability for the offshore data. Putting this into context by considering the normalised standard deviations $z_{\text{onshore}} = (\frac{\sigma}{\mu})_{\text{onshore}}$, $z_{\text{offshore}} = (\frac{\sigma}{\mu})_{\text{offshore}}$ shows that $z_{\text{onshore}} \approx 0.5 * z_{\text{offshore}}$ holds true.

The limitations of the approach are the significant computational training times, their performance not always guaranteed, and the approach being single-purpose. Particularly, the computational training time is expected to perform poorly for large electrical power systems with many nodes and lines. Novel optimisation approaches from operations research and making advanced solvers accessible for CVXPY would address this limitation. End-to-end learning involves solving optimisations in the training loss resulting in substantial computational times for large systems; however, using commercial solvers can drastically reduce the times. End-to-end learning was not always better than conventional learning, depending on the geographical location and data settings. In our study, end-to-end learning worked better when predicting offshore wind rather than wind onshore. Another limitation is that the trained NN models are single-purpose, an inherent drawback of end-to-end learning. In comparison, using conventional learning optimising for accuracy of the wind prediction generally develops well-accepted multi-purpose prediction models.

5. Conclusion

The end-to-end training of wind power prediction models appears promising if mathematical optimisation can model the purpose of the prediction. Interestingly, in our study, end-to-end learning showed particularly strong benefits in complex and constraining environments, e.g. complex congestion constraints or overloads in the power system at high offshore wind conditions. In such environments, the trained

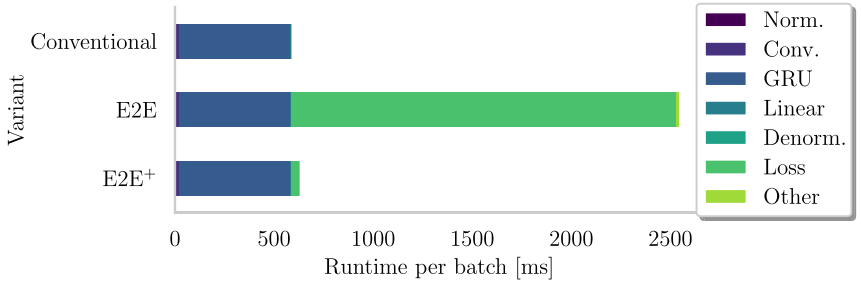


Fig. 16. Computational requirement of conventional and end-to-end learning. End-to-End+ denotes a hypothetical training method that incorporates faster solvers.

wind power model can result in 10% lower system cost errors and up to 70% reduced error variance. Constraining the OPF through tighter line limitations amplifies the advantages of the proposed method. However, minimising the system cost errors did not necessarily correspond to a reduction in wind curtailment, as these are currently not considered with penalty fines. Furthermore, the computation times of end-to-end learning are high, and the specialised nature of the trained model (desirable) limits its application for other prediction purposes. Therefore, our recommendations are to implement commercial solvers in packages that can be integrated into NN training environments to speed-up computations. Additionally, our future research will investigate multi-purpose end-to-end learning of wind power prediction models with different (multiple-) objective functions for different purposes.

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

Data will be made available on request.

Acknowledgments

This work is supported by an IDEA League research scholarship, and TU Delft AI Labs Programme, NL, The Netherlands.

References

[1] M. Joos, I. Staffell, Short-term integration costs of variable renewable energy: Wind curtailment and balancing in Britain and Germany, Renew. Sustain. Energy Rev. 86 (2018) 45–65, http://dx.doi.org/10.1016/j.rser.2018.01.009.

[2] B. Yang, L. Zhong, J. Wang, H. Shu, X. Zhang, T. Yu, S. Liming, State-of-the-art one-stop handbook on wind forecasting technologies: An overview of classifications, methodologies, and analysis, J. Clean. Prod. (2020) 124628.

[3] E. Constantinescu, V. Zavala, M. Rocklin, S. Lee, M. Anitescu, Unit Commitment with Wind Power Generation: Integrating Wind Forecast Uncertainty and Stochastic Programming, Tech. Rep., Argonne National Lab. (ANL), Argonne, IL (United States), 2009.

[4] X. Zhao, N. Jiang, J. Liu, D. Yu, J. Chang, Short-term average wind speed and turbulent standard deviation forecasts based on one-dimensional convolutional neural network and the integrate method for probabilistic framework, Energy Convers. Manage. 203 (2020) 112239.

[5] M. Lotfi, M. Javadi, G.J. Osório, C. Monteiro, J.P. Catalão, A novel ensemble algorithm for solar power forecasting based on kernel density estimation, Energies 13 (1) (2020) 216.

[6] M. Mobtahej, K. Esapour, S.Z. Tajalli, M. Mohammadi, Effective demand response and GANs for optimal constraint unit commitment in solar-tidal based microgrids, IET Renew. Power Gener. 16 (16) (2022) 3485–3495.

[7] M. Lotfi, G.J. Osório, M.S. Javadi, M.S. El Moursi, C. Monteiro, J.P. Catalão, A fully decentralized machine learning algorithm for optimal power flow with cooperative information exchange, Int. J. Electr. Power Energy Syst. 139 (2022) 107990.

[8] M.A. Hossain, R.K. Chakrabortty, S. Elsawah, M.J. Ryan, Very short-term forecasting of wind power generation using hybrid deep learning model, J. Clean. Prod. 296 (2021) 126564.

[9] Y.-Y. Hong, C.L.P.P. Rioflorido, A hybrid deep learning-based neural network for 24-h ahead wind power forecasting, Appl. Energy 250 (2019) 530–539.

[10] S. Harbola, V. Coors, One dimensional convolutional neural network architectures for wind prediction, Energy Convers. Manage. 195 (2019) 70–75.

[11] S. Shah, K. Wang, B. Wilder, A. Perrault, M. Tambe, Decision-focused learning without decision-making: Learning locally optimized decision losses, in: Advances in Neural Information Processing Systems, 2022.

[12] A.N. Elmachtoub, P. Grigas, Smart "predict, then optimize", Manage. Sci. 68 (1) (2022) 9–26.

[13] B. Amos, J.Z. Kolter, Optnet: Differentiable optimization as a layer in neural networks, in: International Conference on Machine Learning, PMLR, 2017, pp. 136–145.

[14] S. Bai, J.Z. Kolter, V. Koltun, Deep equilibrium models, Adv. Neural Inf. Process. Syst. 32 (2019).

[15] S. Gould, R. Hartley, D.J. Campbell, Deep declarative networks, IEEE Trans. Pattern Anal. Mach. Intell. (2021).

[16] J. Kotary, F. Fioretto, P. van Hentenryck, B. Wilder, End-to-end constrained optimization learning: A survey, in: 30th International Joint Conference on Artificial Intelligence, IJCAI 2021, International Joint Conferences on Artificial Intelligence, 2021, pp. 4475–4482.

[17] J. Kotary, M.H. Dinh, F. Fioretto, Folded optimization for end-to-end model-based learning, 2023, arXiv preprint arXiv:2301.12047.

[18] P.L. Donti, B. Amos, J.Z. Kolter, Task-based end-to-end model learning in stochastic optimization, Adv. Neural Inf. Process. Syst. (2017) 5484–5494, arXiv:1703.04529.

[19] R. Vohra, A. Rajaei, J.L. Cremer, End-to-end learning with multiple modalities for system-optimised renewables nowcasting, in: 2023 IEEE Belgrade PowerTech, IEEE, 2023, pp. 1–6.

[20] A. Agrawal, S. Barratt, S. Boyd, E. Busseti, W.M. Moursi, Differentiating through a cone program, J. Appl. Numer. Optim. 1 (2) (2019) 107–115.

[21] A. Agrawal, B. Amos, S. Barratt, S. Boyd, S. Diamond, J.Z. Kolter, Differentiable convex optimization layers, Adv. Neural Inf. Process. Syst. 32 (2019).

[22] A.J. Wood, B. Wollenberg, Power Generation, Operation and Control, John Willey & Sons, Inc, New York, 1984.

[23] M.B. Cain, R.P. O'neill, A. Castillo, et al., History of optimal power flow and formulations, Fed. Energy Regul. Comm. 1 (2012) 1–36.

[24] F. Capitanescu, Critical review of recent advances and further developments needed in AC optimal power flow, Electr. Power Syst. Res. 136 (2016) 57–68.

[25] M. Borning, A. Moser, Integrated grid and power market simulation: Investigating the required modeling level of detail, EEM, in: 16th International Conference on the European Energy Market, IEEE, Ljubljana, Slovenia, 2019, pp. 1–5, http://dx.doi.org/10.1109/EEM.2019.8916331.

[26] C. Weber, Uncertainty in the Electric Power Industry: Methods and Models for Decision Support, vol. 77, Springer Science & Business Media, 2006.

[27] B. Amos, Differentiable Optimization-Based Modeling for Machine Learning, (Ph.D. thesis), Carnegie Mellon University, 2019.

[28] K. Cho, B. van Merriënboer, D. Bahdanau, Y. Bengio, On the properties of neural machine translation: Encoder–decoder approaches, Syntax Semant. Struct. Stat. Transl. (2014) 103.

[29] D. Clevert, T. Unterthiner, S. Hochreiter, Fast and accurate deep network learning by exponential linear units (ELUs), in: Y. Bengio, Y. LeCun (Eds.), 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2–4, 2016, Conference Track Proceedings, 2016, URL http://arxiv.org/abs/1511.07289.

[30] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, A. Talwalkar, Hyperband: A novel bandit-based approach to hyperparameter optimization, J. Mach. Learn. Res. 18 (1) (2017) 6765–6816.

[31] J. Muñoz Sabater, et al., ERA5-land hourly data from 1981 to present, in: Copernicus Climate Change Service (C3S) Climate Data Store (CDS), vol. 10, 2019.