university of groningen / faculty of science and engineering

WIND SPEED PREDICTION FEATURE DESIGN FOR ECHO STATE NETWORKS

Bachelor's Project Thesis

Niels de Jong · n.a.de.jong@student.rug.nl
Supervisor: Prof. Dr. H. Jaeger

Abstract: Due to a multiplicity of factors, the world's energy demand will increase for the upcoming decades. In meeting this demand, renewable energy sources such as wind are seen as attractive alternatives, but their high degree of variability makes integration into power grids a challenge. Intelligent prediction methods, such as echo state networks (ESNs), may prove to be part of the solution. However, studies on feature design for input into such networks is scarce. Thus, the question "What good features can be designed in helping to solve the very short-term wind speed prediction task by supervised machine learning?" is posed, concentrating on ESNs. To address this question, six meteorological variables from the WIND toolkit are subjected to a series of performance tests via appropriately tuned ESN models. By comparing resulting average NRMSE values to a reference persistence model, three conclusions are drawn: (I) only the feature wind speed appears to be informative in the task; (II) spatially diverse features are informative, and (III) decomposing input features into smooth and noise sub-signals raises informative quality.

Keywords: Echo State Networks (ESNs) · Wind Speed Prediction · Supervised Machine Learning

1 Introduction

According to the United Nation's world population medium-variant projection, mankind is expected to grow until at least the end of this century (United Nations, Department of Economic and Social Affairs, Population Division, 2019). Additionally, multiple nations are within an industrialising process, and see their living standards get raised. Consequentially, the global energy demand is anticipated to rise further.

In order to meet this demand, up until now, mostly traditional energy sources have been expended. These are primarily (hydro)carbons and their derivatives (petrol, coal, and natural gas; Panwar, Kaushik, and Kothari, 2011). However, multiple downsides to these sources have been identified in the last several decades, such as its expected depletion at the current rate of exploitation, as well as its environmental impact (Park, 2014) for instance, demonstrates this for ash from coal-powered plants). As such, policy makers and others have turned to so-called alternative (or renewable) energy sources: "[...] primary, domestic [i.e. not exclusively for industry-level usage], inexhaustible sources of energy" (Panwar et al., 2011, p. 1513). Renewables consist chiefly of solar, wind, hydro, geothermal, and tidal energy. Their contribution to the global energy supply is expected to rise by 20 percent points from 2020 to 2040 (Panwar et al., 2011).

However, adoption of renewable energy sources is obstructed by an inherent property that problematises integration into power grids: their intermittency (or, irregularity) of output (Diógenes, Claro, Rodrigues, & Loureiro, 2020; Sinsel, Riemke, & Hoffmann, 2020; Soman, Zareipour, Malik, & Mandal, 2010; Veers et al., 2019). While traditional sources such as coal can be steadily supplied and burned, users are dependent on meteorological circumstances whether wind will blow, or the sun will shine. The system that produces these conditions cannot easily be modelled, and, as a result, power grid operators cannot reliably depend on the output of renewables; this inevitably leads to over- and undersupply of power, which is expensive (Ahmed, Al-Ismail, Shafiullah, Al-Sulaiman, & El-Amin, 2020).

To address this irregularity problem, a straightforward idea is to predict the energy sources output

for some time horizon. Sinsel et al. (2020) reach this conclusion after their root-cause analysis on integration of renewable energy into power grids. Further, Veers et al. (2019) state in their third grand challenge of harnessing wind energy that "computational modelling methods" are required, as "[...] wind [...] must provide more predictable and controllable power" (p. 6). This latter community – that of wind energy – seems to have embraced the prediction approach, where some even argue that it is 'invaluable' for successfully harnessing wind for energy (Foley, Leahy, Marvuglia, & McKeogh, 2012).

Since numerous studies have been conducted on wind speed prediction, three taxonomies have been developed throughout reviews to categorise and orient approaches; they are based on time frame (Ahmed et al., 2020; Foley et al., 2012; Jung & Broadwater, 2011; Soman et al., 2010), intended use (Jung & Broadwater, 2011), and approach (Foley et al., 2012; Jung & Broadwater, 2011; H. Liu, Chen, Lv, Wu, & Liu, 2019). The first system categorises predictors based on the prediction range: very short-term (1 s–0.5 h), short-term (until 6 h), medium-term (until one week), and long term (up until years). System two classifies based on whether individual turbines, groups of turbines or several wind parks are used, whether predictions are given in probability ranges (or not), and whether turbines are on- or offshore. Lastly, the third system groups predictors by their method: either via uncertainty analysis (which is probabilistic), or single-point forecasts (which can be physical, statistical, intelligent, or hybrid in nature). (Here, 'intelligent' refers to the array of AI methods to address classification and regression problems.) When examining the existent wind speed prediction literature, it can be noted that most work is performed in the (very-)short time frame, focuses on one or multiple turbines, and use intelligent and hybrid predictors (sometimes also called 'ensemble' methods).

Especially for these frequently-used intelligent predictors, the quality of input data is of great importance, as it largely determines the success of its performance. Besides tuning the system to the input variables, it is also becoming central to select among features so as to raise the generalising ability of the network, in part due to the rise in availability and abundance of data (Guyon & Elisseeff, 2003). A brief overview of this discipline is therefore in order.

Feature selection, defined as "a process that selects a subset of original features," (H. Liu and Yu, 2005, p. 491) may benefit intelligent predictors in three ways: it reduces resources (such as computation time and energy), maintains or even increases the performance of models, and may give users insight into the relevant factors of the system which their model is seeking to mimic. Generally, feature selection approaches are divided into a filter, wrapper, and hybrid category (H. Liu & Yu, 2005; Salcedo-Sanz, Cornejo-Bueno, Prieto, Paredes, & García-Herrera, 2018). When a predictor's performance is used to evaluate subsets of data, the wrapping approach is utilised. If, instead, an external measure of 'goodness' is used (such as degree of correlation), the approach belongs to the former category. Combining the two lead to hybrid methods.

Within the literature on wind speed prediction, multiple authors have addressed the issue of feature selection. Examples include Martín-Vázquez, Aler, and Galván (2017), who studied selection of features for prediction of wind speed at a simulated wind farm in Sotavento, Spain. Specifically, they employed two machine learning models (a support vector machine, and a gradient tree boosting algorithm) that functioned as both wrapper evaluators, and as overall predictors. Similarly, Salcedo-Sanz et al. (2018) developed a unique algorithm, called Coral Reef Optimisation with Substrate Layer (CRO-SL) which, together with an Extreme Learning Machine (ELM), cooperated in a wrapper feature selector for wind speed prediction on an hourly and daily basis. The authors in this work also give a review of other studies on feature selection for renewable energy source prediction; they estimate from the collected results that utilising feature selection may lead to improvements between 5–40%.

Nonetheless, most studies in wind speed prediction appear to revolve mainly on the predictor and pre-processing techniques used. When more data becomes publicly available, and particularly when information collected at various turbines are considered (that is, a spatial dimension is added to the prediction), feature selection may improve results.

Statement of research question

Considering this relative scarcity of analyses on feature selection and design, this paper addresses the question: "What good features can be designed in helping to solve the very short-term wind speed prediction task by supervised machine learning?"

In order to operationalise this research question, some terms need to be specified. Particularly, 'good' is interpreted as the improvement (or deterioration) on one or more error metrics, in comparison to not including the feature in the net. Furthermore, multiple features will be proposed, but no claim will be made that this list exhausts all possible (or 'good') inputs. Also, a specific data set will be chosen to represent the task of wind speed prediction. Lastly, an echo state network neural network architecture will serve as supervised machine learning method, as it has certain properties that may prove useful in the task of predicting (highly) nonlinear time series, to which wind speed progression over time belongs. Moreover, within the literature, this network type appears to have not yet been considered in a feature selection scheme; its use could thereby contribute to the wind speed feature selection literature.

Furthermore, it must be noted that this study builds on the work of Khanal (2018). In their work, wind speed was predicted with an ESN at a thirty-minute interval. In this work, the scope of their work will be expanded by (I) explicitly considering *which* 'good' features can be designed, (II) at *numerous* time intervals (not only 30 minutes).

2 Task Description

Predicting upcoming wind speed at some location may be regarded as an instance of the supervised (machine) learning task; a high-level description is given below.

The supervised learning task

The supervised learning task consists of a so-called training and testing set, S_{train} and S_{test}. These sets share a similar structure: both contain exclusively pairs $\langle I, O \rangle$, where I (also known as the input) and O (the output) are tuples of lengths L and K, respectively. $L, K \geq 1$. Moreover, L and K are equal for all pairs in both S_{train} and S_{test}. If the elements of O are discrete (e.g. a subset of $\mathbb{Z}$), the task is sometimes referred to as a classification task. If, on the other hand, O stores continuous values, the term regression task is often employed.*

Now, the goal of this task is as follows. Given a presented input I, predict O 'as accurately as possible'. Mathematically, we seek the model f^* mapping I to O, such that:

$$f^* = \arg\min_{f \in F} \left(\varepsilon(O_f, O) \right), \text{where}$$
$$O_f = \{ f(i) \mid i \in I \} \quad \text{(Output of } f\text{)}$$

where F is the space of functions in which is searched, and $\varepsilon(\cdot, \cdot)$ represents a loss function that assigns to the discrepancy of O_f to O an (ordinarily non-negative) real value, indicating the degree of error.

In order to do so, the learning procedure is split into two parts: the training and testing phase. E_{train} and E_{test} belong to the former and latter phase, in that order. In the former phase, Os are given as feedback to improve – thus the supervisory part of the task – while in the latter either no feedback is used, or own predictions are used in future evaluations.

The notion of 'accuracy' in the task's goal left as-is is ambiguous: after all, how is ε implemented?' Numerous specific error metrics have been developed. Two often-used examples are the normalised root mean squared error (NRMSE) and the mean absolute percentage error (MAPE).

The wind speed prediction task

In the specific case of predicting wind speed, the inputs I consist of features that are intended to be of use in the forecasting process. Examples are the present-moment wind speed or temperature at the turbine's altitude, but derivative features such as decomposed sub-signals or signals from spatially distant turbines may be considered as well. O consists of the future wind speed. Different prediction horizons may be selected, but here the very short-term horizon is chosen; it varies from a couple of minutes to around half an hour.

*Of course, it may be that O holds both limited, discrete values and continuous values; this description is not intended to be exhaustive.

3 Model

The echo state network (ESN) is a specific kind of recurrent neural network, in turn an approach to addressing the supervised learning problem. Conceptually, the ESN architecture was conceived by two teams in tandem, with different research objectives. The first party consisted of Jaeger (2001), whose naming we use in this paper, while the second team composed of Maass, Natschlaeger, and Markram (2002) called the network type liquid state machines (LSMs). Nowadays, both approaches are categorised under the term reservoir computing (RC), along with similar techniques.

Describing the approach at a low level of resolution, there are two primary phases in training an ESN. At the first step, the network is driven by the sequence of training inputs, which alters the state of the reservoir. If this is done appropriately, the following step involves connecting a set of 'optimal' readout weights from the reservoir to the output. The network is then ready to map novel inputs to outputs.

Theory

The echo state network consists of a number of elements, with which we can compose the equations that govern its dynamics. To begin with the first set of elements, an echo state network consists of three primary layers: the input, reservoir, and output. These layers are represented by three vectors (which vary over discrete time), $\mathbf{u}(n)$, $\mathbf{x}(n)$, and $\mathbf{y}(n)$, and have corresponding dimensions L, N, and K.

Said layers are connected by a set of four (static) weight matrices: the input is linked to the reservoir via $\mathbf{W}^{\text{input}} \in \mathbb{R}^{N\times K}$, the reservoir has internal connections (i.e. recurrent pathways) via $\mathbf{W} \in \mathbb{R}^{N\times N}$, reservoir-to-output connections are governed by $\mathbf{W}^{\text{out}} \in \mathbb{R}^{L\times(K+N+L)}$ (although recurrent connections among the outputs are less common), and finally, output-to-reservoir feedback weights are specified with $\mathbf{W}^{\text{back}} \in \mathbb{R}^{N\times L}$.

Moreover, two-tuples of functions $\mathbf{f}$ and $\mathbf{f}^{\text{out}}$, a bias vector $\mathbf{b}$, and a noise vector $\nu(n)$ will be employed. The first two consist of functions to apply to single entries of a reservoir state. Further, $\mathbf{b}$ and $\nu(n)$ are vectors whose entries are sampled from a distribution (one frequently used is the uniform distribution: unif$(-a, a)$, $a \in \mathbb{R}^+$). The main difference between the two is that the latter changes per time step; the former is 'static', and serves to diversify the nature of the reservoir neurons by applying individual shifts to their activations.

Given these components, we can specify how the ESN updates its state in light of new input. Furthermore, we can describe how, after having been presented with the training inputs, the ESN can compute an appropriate $\mathbf{W}^{\text{out}}$.

The former is defined in two steps. Specifically, the state of an ESN at time t is updated with

$$\widetilde{\mathbf{x}}(n) = \mathbf{f}\Big(\mathbf{W}^{\text{in}}\mathbf{u}(n) + \mathbf{W}\mathbf{x}(n-1) + \mathbf{W}^{\text{back}}\mathbf{y}(n-1) + \mathbf{b}\Big) + \nu(n), \tag{3.1}$$

$$\mathbf{x}(n) = (1-\alpha)\,\mathbf{x}(n-1) + \alpha\widetilde{\mathbf{x}}(n). \tag{3.2}$$

Here, Equations 3.1 and 3.2 are the first and second steps, respectively. The latter step is characteristic for leaky integrator neurons; we use these neuron types in this work. $\alpha \in \langle 0, 1]$ is the leaking rate – a parameter of the network.

The update equation is repeated numerous times during training, once for each input vector $\mathbf{u}(n)$. Once the training reaches its end, the second central equation in the ESN scheme is used; it specifies the weights from the reservoir to the output layer. Multiple options are available, but two common ones are:

$$\mathbf{W}^{\text{out}} = \big(\mathbf{f}^{\text{out}}\big)^{-1}(\mathbf{D})\,\mathbf{E}^{+}; \tag{3.3}$$

$$\mathbf{W}^{\text{out}} = \big(\mathbf{f}^{\text{out}}\big)^{-1}(\mathbf{D})\,\mathbf{E}\,(\mathbf{E}\mathbf{E}' + \beta\mathbf{I})^{-1}, \tag{3.4}$$

where $\mathbf{D} \in \mathbb{R}^{L\times T}$ (with T the number of discrete time steps) is the teacher signal, and $\mathbf{E} = [\mathbf{U}; \mathbf{X}; \mathbf{Y}] \in \mathbb{R}^{(K+N+L)\times T}$ the matrix of extended system states, collected over columns. In Equations 3.3 and 3.4, the first approach is the direct pseudo-inverse solution (which may be computationally demanding), and the second ridge regression (or regression with Tikhonov regularisation); $\beta \in \langle 0, 1]$ is another parameter to the model: the ridge regression coefficient.

Given this $\mathbf{W}^{\text{out}}$, the output of the system at the specified time is given by

$$\mathbf{y}(n) = \mathbf{f}^{\text{out}}\big(\mathbf{W}^{\text{out}}\mathbf{e}(n)\big), \tag{3.5}$$

where $\mathbf{e}(n)$ is the extended system state at time step n.

The last topic of the theory behind ESNs discussed here involves the echo state property (or ESP). The ESP is applicable to an ESN when the reservoir state can be constructed completely via the (left-infinite) input history. Expressed mathematically:

$$\mathbf{x}(n) = \mathbf{f}^{\text{echo}}(\ldots, \mathbf{u}(n-1), \mathbf{u}(n)), \tag{3.6}$$

where the $f \in \mathbf{f}^{\text{echo}}$ are the so-called echo functions.

The ESP is of essential importance in both theory and practice of ESNs, as it is a necessary condition for them to function properly. It also provides the foundation for ESNs functioning as networks with the capability to store historical inputs (that is, to 'memorise'); this is central for certain tasks such as time series prediction, in which consecutive inputs are heavily determined by preceding inputs.

Theoretical findings, laid out in Jaeger (2001), give two conditions that may serve as 'configuration boundaries' of ESNs. The first condition is sufficient for the ESP to hold. It states that if the largest singular value $\sigma_{\max}$ of the internal weight matrix $\mathbf{W}$ is less than 1, the system has echo states. In contrast, the second condition is sufficient to rule out echo states, and asserts that if the spectral radius of $\mathbf{W}$ (its largest absolute eigenvalue) is strictly greater than 1, and the input consists of zeroes, the ESN does not have echo states.

Architecture

As was stated before, although echo state networks are conceptually rich, the basic algorithm can be

Algorithm 3.1 Initialise the ESN.

Require: $\langle K, N, L \rangle$, the input, reservoir, and output layer sizes; c, the connectivity parameter, and s_{in}, s_{res}, and s_{back}, the connection scalars.
Ensure: An initialised ESN, ready for training.
$\mathbf{W}^{\text{in}} \leftarrow$ A dense $N \times K$ matrix, whose entries are sampled from $s_{\text{in}} \cdot \text{unif}(-1, 1)$
$\mathbf{W} \leftarrow$ A sparse $N \times N$ matrix with c percent connectivity, whose values – if nonzero – are sampled from $\text{unif}(-1, 1)$. Hereafter, $\mathbf{W}$ gets scaled by $s_{\text{res}}/\rho(\mathbf{W})$
$\mathbf{W}^{\text{back}} \leftarrow$ A dense $N \times L$ matrix, whose values are sampled from $s_{\text{back}} \cdot \text{unif}(-1, 1)$
$\mathbf{x} \leftarrow \mathbf{0}$ (of size $\mathbb{R}^N$)

Algorithm 3.2 Train the ESN.

Require: Leaking rate α and ridge regression coefficient β; an initialised ESN, and a training input-output tuple $\langle \mathbf{U}, \mathbf{D} \rangle$.
Ensure: A weight matrix $\mathbf{W}^{\text{out}}$ that minimises the error between $\mathbf{D}$ and $\mathbf{f}^{\text{out}}(\mathbf{W}^{\text{out}}\mathbf{E})$ in a least squares sense. (NOTE. Here, we use Equation 3.4, but this is not the only option available.)
for $0 \leq n < T$ **do**
Set $\widetilde{\mathbf{x}}(n)$ according to Equation 3.1
Set $\mathbf{x}(n)$ according to Equation 3.2
$\mathbf{e}(n) \leftarrow [\mathbf{u}(n); \mathbf{x}(n); \mathbf{y}(n-1)]$
end for
$\mathbf{E} \leftarrow \mathbf{E}\,[:, n_{\text{transient}} : T]$
$\mathbf{Y}^{\text{train}} \leftarrow \mathbf{Y}^{\text{train}}\,[:, n_{\text{transient}} : T]$
$\mathbf{W}^{\text{out}} \leftarrow (\mathbf{f}^{\text{out}})^{-1}\left(\mathbf{Y}^{\text{train}}\right)\mathbf{E}'\left(\mathbf{E}\mathbf{E}' + \beta\mathbf{I}\right)^{-1}$

Algorithm 3.3 Predict with the ESN.

Require: A trained ESN and a set of testing inputs $\mathbf{U} \in \mathbb{R}^{K \times T}$.
Ensure: A sequence of outputs $\mathbf{Y}^{\text{pred}}$ obtained from the high-dimensional ESN reservoir.
$\mathbf{x}, \mathbf{Z} \leftarrow$ empty
$\mathbf{y}^{\text{prev}} \leftarrow \mathbf{0}$
for $0 \leq n < T$ **do**
Set $\widetilde{\mathbf{x}}(n)$ according to Equation 3.1
Set $\mathbf{x}(n)$ according to Equation 3.2
$\mathbf{e} \leftarrow [\mathbf{u}(n); \mathbf{x}(n); \mathbf{y}^{\text{prev}}]$
$\mathbf{y}^{\text{prev}} \leftarrow \mathbf{W}^{\text{out}}\mathbf{e}(n)$
end for
return $\mathbf{f}^{\text{out}}(\mathbf{W}^{\text{out}}\mathbf{E})$

described quite concisely. In essence, to train these networks, three stages follow up one another. These are the (I) initialisation, (II) training, and (III) prediction stages of the ESN. These will now be discussed briefly; the full pseudocode is laid out in Algorithms 3.1–3.3. Note here, that $\rho(\cdot)$ is the spectral radius of a matrix, that is, the largest absolute value of said matrix's spectrum (or, its set of eigenvalues).

During the section on ESN theory, it already became apparent that echo state networks consist of multiple elements, such as layers, weights, and multiple central state-changing formulas. Most of these elements are initialised at the conception of the network – that is, even before it is trained. Algorithm 3.1 shows specifically which components fall within this category. Notice that $\mathbf{W}^{\text{back}}$ may be disabled by setting $s_{\text{back}} = 0$. Doing so is required for any problem that does not show elaborate pattern-generating behaviour. The other parameters, on the other hand, are meant to be set to nonzero values; the network will break otherwise.

Note that the distributions used during the initialisation can be replaced by, for instance, normal$(0, 1)$. Indeed, in general, many components in this outline are exchangeable for similar parts. Since this specific setup gives a good impression of the idea of ESNs in practice, we use these settings.

After initialisation, the network is driven with the training input. If feedback is enabled, the desired, previous outputs $\mathbf{y}^{\text{train}}(n-1)$ are supplied via the $\mathbf{W}^{\text{back}}$ into the reservoir, which is called teacher forcing. (In contrast, during prediction, the ESN's own outputs $\mathbf{y}(n-1)$ are used.) While continually executing the two update steps, an extended system state matrix $\mathbf{E} \in \mathbb{R}^{(K+N)\times T}$ is filled with the current vertically stacked $[\mathbf{u}(n); \mathbf{x}(n)]$s. Once finished, a transient period (in which the network had to escape from the initial $\mathbf{x} = \mathbf{0}$ condition) $t_{\text{transient}}$ is left-removed from the extended system state and training output matrices $\mathbf{E}$ and $\mathbf{Y}^{\text{train}}$. Then, the 'supervision step' of the ESN commences: linear regression with respect to the training output.

Two approaches to this linear regression problem – finding that $\mathbf{W}^{\text{out}}$ that minimises the error between $\mathbf{Y}^{\text{train}}$ and $\mathbf{W}^{\text{out}}\mathbf{E}$ (in what specific sense depends on implementation) – have already been listed. If users opt for Equation 3.4 (which we do in our listings), a regression parameter β needs to be set, which manages regularisation during the training. This penalises excessively large $\mathbf{W}^{\text{out}}$, suspect of potential overfitting.

Once $\mathbf{W}^{\text{out}}$ has been calculated, the network can predict novel input sequences $\mathbf{U}$. The procedure works largely the same as it was during training: drive the reservoir by the input data, and store the results in $\mathbf{E}$. The only large differences are that we do not remove the transient period – the network is already attuned to the type of data it needs to handle – and the fact that we store outputs for the final result by computing $\mathbf{y}(n) \leftarrow \mathbf{f}^{\text{out}}(\mathbf{W}^{\text{out}}\mathbf{e}(n))$ for the different ns.

Considering the complete algorithm, we may see why ESNs are adept at handling nonlinear data, and why they are relatively fast in training.

The first phenomenon can be explained largely by the projection of the input sequences $\mathbf{u}(n)$ into the high-dimensional 'reservoir space' $\mathbb{R}^N$ via the $\mathbf{W}^{\text{in}}$, especially with relatively large networks, such as those with thousands of nodes. This allows the information to be more easily discriminated. Moreover, due to the relative sparseness and appropriate spectral radius of $\mathbf{W}$, the network has rich, nonlinear dynamics, which also greatly contributes to the flexibility of the ESN in prediction and classification tasks.

Phenomenon two emerges mainly due to the last step of the output training. Normally, RNNs need to use intensive backpropagation techniques for training. In contrast, the ESN needs to perform a relatively inexpensive linear regression of the teacher output.

Prediction Task

Lastly, a brief exposition of the prediction task in the context of the ESN is follows, relating the discussed material to the task at hand.

Given a set of inputs $I_{\text{test}} \in \mathbb{R}^{L\times T}$ for testing, the goal is to obtain predictions $O_{\text{test}} \in \mathbb{R}^{K\times T}$ that are $l \in \mathbb{Z}^+$ steps forward in time with respect to those of I_{test}. This is achieved by (I) initialising the ESN according to Algorithm 3.1, (II) training the network with training data according to Algorithm 3.2 (which involves inputs different from sI_{test}), and, lastly, (III) driving the ESN's reservoir by I_{test} and reading out its response at the output layer, yielding O_{test}.

4 Methods

Data

For this project, a dataset needs to meet a number of requirements so as to be useful in answering the research question. First, it has to be of sufficient size, as machine learning methods such as the used ESN simply require considerable amounts of data, or at least tend to perform better when more data is available to train, validate, and test with. Second, it has to be of sufficient 'temporal granularity' in the sense that it could be employed for very short-term wind speed prediction. Third, it has to have more features than just wind speed data, as the research question relies on the breadth of raw features to design new (and potentially useful) ones. Lastly, it has to capture data for multiple locations, so that the spatial dimension of features could be used in feature design, too; Jung and Broadwater (2011) and others suggest this may be useful.

A dataset that supersedes the one used by Khanal (2018) (i.e. the Eastern Wind dataset) has been selected. The Wind Integration National Dataset (WIND), built by Draxl, Clifton, Hodge, and McCaa (2015) at the National Renewable Energy Laboratory (NREL), is a large set of simulated data for the integration of wind into the United States power grid. It consists of data for over 126,000 sites (equally spaced out in 2×2 km patches, with 7 years of data (ranging from 2007–2013), where each site possesses 17 variables. Since not all features were available to the public, only wind speed, wind direction, temperature, barometric pressure, air density, and wind power are used (all available features, except time, which was used for linking the data indices with dates and times); also see Table 4.1 for an overview. Requirements I, III and IV are thereby met; condition II is achieved due to the grid spacing in the data.

Since working with all data would be unnecessarily cumbersome, an arbitrary 3×3 set of sites in Diamond, Missouri have been chosen; see Figure 4.1 for details. From these sites, the 6 features are tried and combined both temporally and spatially to determine what may be of help in solving the wind speed prediction problem using the ESN.

Importantly, the features or feature combinations are all normalised before exposing them to the ESN, meaning the mean value of the signals is brought

Figure 4.1: The locations of the turbines, along with their relative locations listed above the points. Coordinates range from N 37°0′34.3″ to N 37°2′48.3″, and from W 94°18′10.5″ to W 94°21′0.1″. IDs of the sites are – from top-left to bottom-right – 30844, 30845, 30846, 30682, 30683, 30684, 30509, 3050, and 30511.

#	Feature	Detail	Dimension	Available	Used
1	Time	In UTC	s	×	
2	Wind speed	At 100 m	m/s	×	×
3	Wind direction	At 100 m	deg	×	×
4	Temperature	At 2 m	K	×	×
5	Barometric pressure	At surface	Pa	×	×
6	Air density	At 100 m	kg/m^3	×	×
7	Wind power	Per 5 min	MW	×	×

Table 4.1: The Seven available features (of seventeen) provided by the WIND dataset (partially adapted from Draxl et al. (2015)). × indicates applicability.

to zero, and the variance is set to one. This is central for echo state networks (among other network types), as it allows all input signals to contribute equally to the reservoir state (although different input weight scalings could alter this situation), and prevents saturation by certain high-magnitude signals.

For two examples of original (not-yet normalised) input signals, and of a decomposed signal, see Figure 4.2.

Decomposition

As was noted in the introduction, wind speed (and generally any meteorological variable) tends to vary quite a lot over time. This volatility may lower the quality of the data in certain contexts, such as supervised machine learning.

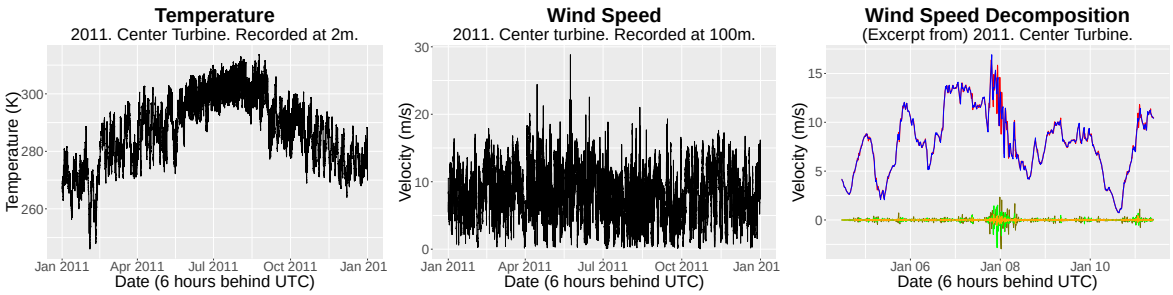
This problem has been identified by multiple researchers (for a review, see Qian, Pei, Zareipour, and Chen, 2019); the literature appears to largely agree that decomposition of the input data may prove to be a partial solution.

Decomposition of time series data, in essence, involves applying a procedure to said data to obtain multiple 'derived signals'. For instance, trends, seasonal oscillations and noise may be separated from one another using certain methods. Since such 'subsignals' only represent one aspect of the original time series, users can identify what components of the data may be informative for their task, and continue with only those items. These parts may be left as-is, or can be synthesised again with others to obtain filtered signals. A common filtering strategy is to only retain large trends, effectively smoothing the original time series.

Two decomposition methods often used within the wind speed prediction literature are the wavelet transform (as in D. Liu, Wang, and Wang, 2005; H. Liu and Chen, 2019; Wang, Lei, Liu, Peng, and Liu, 2019) and singular spectrum analysis (such as Chen et al., 2019). Due to the relatively large size of the input features – leading to considerable computation times – only the wavelet transform is employed.

The wavelet transform has been incrementally developed by multiple scholars over a couple of decades (starting in the eighties), but one researcher often referred to as the instigator of modern wavelet theory is Daubechies (1992). It is closely related to the Fourier transform in two regards: both transport a signal to a different domain, and both extract frequency information from the original signal. Different from the latter, however, is that the wavelet transform also obtains localised, temporal information; it actually strikes a balance between the temporal and frequency domains. Thereby, it avoids part of the limitation of the Fourier transform: requiring signals to be represented as being composed of a set of infinitely re-occurring sines and cosines. Especially for more erratically behaving signals, this would lead to a complex representation. Besides inclusion of temporal information, the wavelet transform allows for many basis functions – called wavelets – instead of sines and cosines. Selecting an appropriate basis for representation in the wavelet domain is crucial in effectively decomposing via its technique.

Although the scope of this article does not allow us to go into fine detail of the wavelet transform, the main idea is as follows: a high- and low-pass filter variant of the wavelet are convolved with



(a) Temperature for 2011. **(b) Wind Speed for 2011.** **(c) Decomposition.**

Figure 4.2: Two signals from the input data and a four-level decomposition of the 2011 center turbine wind speed, plotted. Note that for the decomposition, the red signal is the original, while the blue, brown, green, and orange ones are derived signals (from smooth to 'detailed', or, noisy).

the target signal, leading in-place in the signal's left half to an approximation (roughly, 'smoothed') sub-signal, and, in its right half, to a detail component (often containing noise). The procedure then is continually repeated in-place with approximation sections of the signal, until the filter's length has surpassed that of the approximation section; the signal then has been fully decomposed. At this point, undesired sections can be set to zeroes, after which an inverse operation recomposes the signal. The end result is a filtered signal, hopefully more useful for the intended task.

During the experiment, the wavelet and decomposition strategy used (if applicable) will be mentioned in the configuration listings. Furthermore, the Python package by Lee, Gommers, Waselewski, Wohlfahrt, and O'leary (2019) will be used.

Evaluation

Per run, ESN parameters and a selection of to-be-used input features will be utilised. The output feature remains constant: (future) wind speed, specified by lag l. Here, a brief outline of the evaluation of single runs is explained.

First, the inputs $I \in \mathbb{R}^{L\times T}$ and expected outputs $O \in \mathbb{R}^{1\times T}$ (the output dimension K here set to 1) will be lagged by l, creating an input and output partition where respectively the first and last $T-l$ entries are selected: $I_l \in \mathbb{R}^{L\times(T-l)}$ and $O_l \in \mathbb{R}^{1\times(T-l)}$. In the dataset of the current experiment, $l = 1$ would imply a 5-minute lag between input and expected output, for instance.

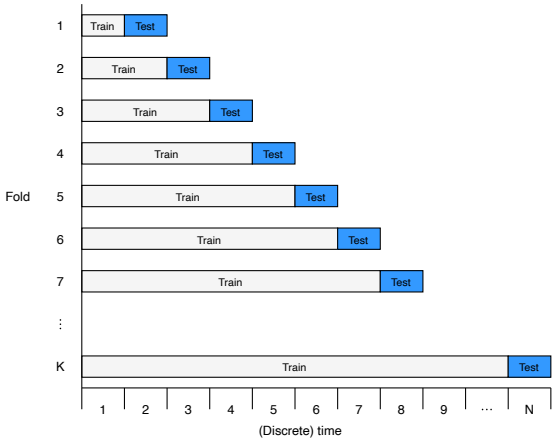


Figure 4.3: A visual summary of K_{split}-fold temporal cross-validation. Note that the lag l between input and output is contained in the training and test sets themselves; it is therefore not represented on the discrete timeline in this figure.

Next, I_l and O_l are split into $K_{\text{split}} \in \mathbb{Z}$ sections, producing $I_{l,1}, I_{l,2}, \ldots, I_{l,K_{\text{split}}} \in \mathbb{R}^{L\times M}$ and $O_{l,1}, O_{l,2}, \ldots, O_{l,K_{\text{split}}} \in \mathbb{R}^{L\times M}$. Here,

$$M = \lfloor (T-l)/K_{\text{split}} \rfloor,$$

where the first $(T-l) \mod K_{\text{split}}$ have one added entry.

With the $I_{l,i}$ and $O_{l,i}$, $K_{\text{split}} - 1$ instances of the supervised learning problem can be made, following

the temporal $K_{(\text{split})}$-fold cross validation scheme. Specifically, the $\langle S_{\text{train}}, S_{\text{test}} \rangle$ tuples are configured to be

$$\begin{aligned}
&\langle \langle I_{l,1}, O_{l,1} \rangle, \langle I_{l,2}, O_{l,2} \rangle \rangle, \\
&\langle \langle I_{l,1} \cup I_{l,2}, O_{l,1} \cup O_{l,2} \rangle, \langle I_{l,3}, O_{l,3} \rangle \rangle, \\
&\quad \vdots
\end{aligned}$$

Also see Figure 4.3 for a visual overview.

For each of the $K_{\text{split}} - 1$ supervised learning problems, the ESN is trained on S_{train} and tested on S_{test}, producing output $O_{\text{actual}} \in \mathbb{R}^{1 \times (T-l)}$ that can be compared with the relevant $O_{l,i}$.

This comparison is achieved via the normalised root mean square error (NRMSE) metric. It is defined as

$$\text{NRMSE}(\mathbf{x}_1(n), \mathbf{x}_2(n)) = \sqrt{\frac{\sum_{n=1}^{T} (x_1(n) - x_2(n))^2}{T \cdot \text{var}^2(\mathbf{x}_2)}}$$

where $\mathbf{x}_1(b)$ and $\mathbf{x}_2(n)$ are data sets (and the second one the expected, 'ground truth' output); T the number of discrete time steps present in the data series, and $\text{var}^2(\cdot)$ the squared variance. This metric is frequently used in the wind speed prediction literature (such as the papers in the introduction). Moreover, the NRMSE improves comparison across datasets (not even necessarily wind speed prediction ones) due to its normalisation component.

For the overall system, $K_{\text{split}} = 12$ folds is set for the cross validation. 'Goodness' is measured in terms of model error, which corresponds to the wrapper approach (cf. the filter and hybrid approaches) discussed in the introduction. Three parameters are optimised, following Lukoševičius (2012, p. 669): the spectral radius $\rho(\mathbf{W})$, the leaking rate α, and the input weight scales s_{in}. Optimisation will not occur on a per-fold level, but per execution of the complete validation scheme. The technique used here is a grid search, where all other parameters are set to the default values given in Table A.1. For the spectral radius, the options are $\{0.60, 0.80, 0.90, 0.95, 1.20\}$, and for the leaking rate they are $\{0.7, 0.8, 0.9, 1.0\}$. For the s_{in}, either decomposition is used, or not. If it is not used, the options are $\{0.8, 0.9, 1.0, 1.1, 1.2\}$; if wavelet decomposition is selected, a Daubechies wavelet with 6 moments (sometimes referred to as a 'D6' wavelet) and depth level 4 decomposition is utilised, producing subsignals $\langle a, d_1, d_2, d_3, d_4 \rangle$ (where a, in wavelet parlance, is called the approximation coefficient, and the ds the detail coefficients). From these, four reconstructions are made, using $\langle a \rangle$, $\langle d_1 \rangle$, $\langle d_2 \rangle$, and $\langle d_3 \rangle$. Then, together with the original signal (at the first index), the reconstructed signals are separately scaled with the following options:

$$\begin{aligned}
&\langle 0.2, 0.2, 0.2, 0.2, 0.2 \rangle; \\
&\langle 0.0, 0.5, 0.25, 0.125, 0.125 \rangle; \\
&\langle 0.0, 0.7, 0.2, 0.05, 0.05 \rangle; \\
&\langle 0.1, 0.8, 0.1, 0.0, 0.0 \rangle; \\
&\langle 0.0, 0.8, 0.2, 0.0, 0.0 \rangle.
\end{aligned}$$

In phases other than phase I (see 'Procedure' section below), input weight scales were determined manually; see Appendix A for further details.

Lastly, the error metrics are compared against those of a naive model, which is also subjected to the temporal $K_{(\text{split})}$-fold cross validation. This model is called the persistence model; for every point in time, it predicts wind speed according to

$$p(n) = o(n-1),$$

that is: the observation at the previous time point is used as prediction for the current time step. Thus, it implicitly assumes that the world will remain static over time. For such noisy data as wind speed, such a model may appear inappropriate. However, for the very short-term time frame, this model is used quite often as reference (see e.g. Jung and Broadwater, 2011).

Procedure

Having covered the data, decomposition method, and approach to performance evaluation, the overall procedure may now be described.

The experiment will unfold in three separate phases. Each phase has a separate question, which, together, aim to aim to answer the research question.

In the first phase, a single year and relative turbine location are chosen (2011 and the 'Center' turbine in Figure 4.1, respectively); only from this location input features are considered. Moreover, a lag of five minutes is used. In this static environment, each of the six input features are exposed to the ESN, first in isolation, and later in groups. This is done in a 'greedy' regime, meaning that, over

iterations, single variables are added to the current set of used features in order to see how performance improves or deteriorates. That feature which improves the performance most is selected as the 'next-in-rank' feature, thereby producing an ordering of the input features. In doing so, the question "what near-turbine (or, local) features are informative to the ESN?" is posed.

In phase two, the constancy constraint of locations is dropped, allowing consideration of input features from remote turbines. As the number of possible input configurations becomes quite large in this stage, configurations are curated based on meteorological plausibility. The question hereby addressed – and complementary to phase one – is: "what remote features are informative to the ESN?"

Phase three closes the experiment. Here, the question "do the findings of phases one and two hold in different situations?" is raised. An attempt to answer it is done by varying the year and degree of lag.

During these three phases, the average NRMSEs of the various configurations for the input weight scales (non-decomposed, and decomposed) are recorded, so as to determine how decomposition may possibly reduce model error (further). At the end of each phase, a percentage range of improvement (or deterioration) by decomposition on average NRMSEs is supplied, summarising over the input features (phase I), extra spatial information added (phase II), and so on. Here, the best non-decomposition input weight scaling model's average NRMSEs are compared with those of the best decomposition model.

By selection, decomposition, and combination (also spatially) of the six input features via these three incremental phases, the research question's 'good input features' are intended to be found in a relatively structured manner.

The project was developed in Python, using the line of packages by Virtanen et al. (2020).

5 Results

As was explained in the methods section, the discussion of the results is split into three parts. Each of these describes the outcomes of a separate phase. As there are numerous parameters that have been optimised per separate problem, only spectral radii and leaking rate ranges – two of the three primary parameters to tune, according to Lukoševičius (2012) – are given here. For all models under discussion here, the reservoir size is $N = 200$ nodes. A separate appendix lists full parameter configurations. Moreover, all models mentioned in this section have been tuned separately before comparison.

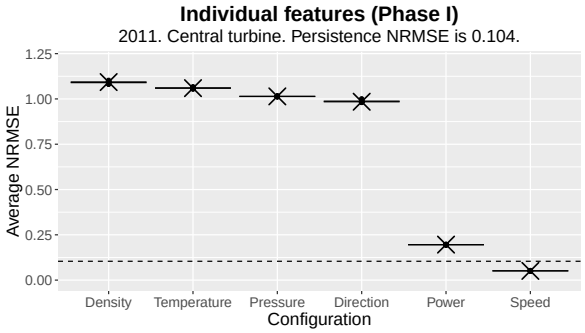


Figure 5.1: Dotplot of performances of individual-feature models in contrast to the persistence model. The features 'Density', ..., 'Speed' refer to features 6, 4, 5, 3, 7, and 2 in Table 4.1. Note: the cross symbolises the median; the bars the standard error.

Phase I

In phase I, the task was to determine, when leaving year, location, and degree of lag static, which input feature (or combination of features) was most informative. This was studied by collecting 10 average NRMSE values from 10 separate runs of the 12-fold temporal cross-validation scheme for each feature model under consideration. Here, a run is understood to be one complete execution of the cross-validation mechanism, where, between runs, only the rabdin ESN model initialisation (e.g.) varies. The resulting samples of average NRMSEs per model are box-plotted in Figure 5.1. Additionally, the average NRMSE of the persistence model under the same procedure is plotted as a dashed line as reference. In this plot, it appears that only the wind speed feature leads to an improvement over the reference model in terms of error. For the individual feature models, spectral radii varied from 0.6–0.7, and leaking rates from 0.9–1.0.

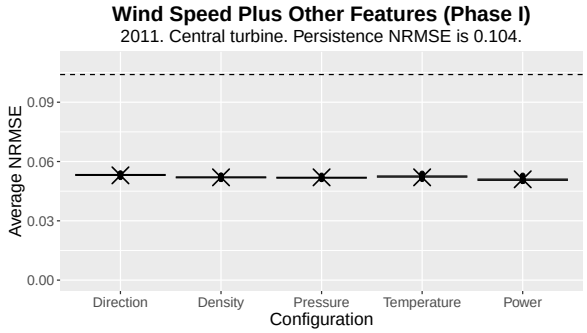


Figure 5.2: Dotplot of performances of non-wind-speed features, plus wind speed (from step 1 of phase I), in comparison to the persistence model.

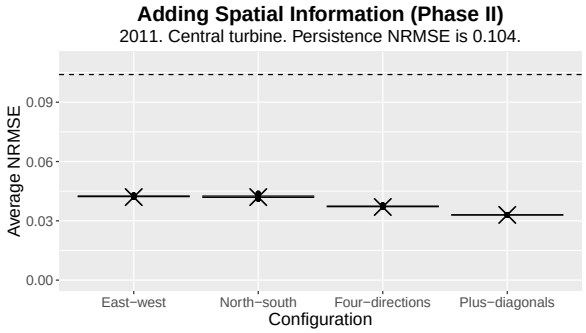


Figure 5.3: Dotplot of performances of models inspired by phase I's final model, augmented with extra spatial information from different directions. These directions refer to the turbines in Figure 4.1.

Continuing the greedy search procedure, a set of follow-up models were developed that incorporated, additionally to wind speed, one of the non-wind speed features with the intention to improve on the only-wind-speed model. The performances of these models in the 12-fold temporal cross-validation regime are displayed in Figure 5.2. Immediately noticed here is that the models do not truly seem to improve over the only-wind-speed model. As such, phase I is halted, concluding that only the wind speed feature appears to be sufficiently informative to the ESN. Quite similar to step 1 of this phase, spectral radii were all set to 0.6, while learning rates varied from 0.9–1.0.

For step I, for wind speed, decomposition reduced the average NRMSE by 30.1%. For air density and temperature, 21.7% and 11.1% improvements were observed, while for the remaining features no large effects were noted.

Phase II

The experiment was subsequently continued by starting phase II. In this second phase, the task was to determine whether features from spatially distant turbines could be successfully incorporated into the existing scheme from phase I to obtain a better prediction.

Building on the impression that only wind speed seemed to be of satisfactory informational quality, wind speed features of the turbine combinations north-south, east-west, north-east-south-west (the four cardinal directions), and all directions (including diagonals such as north-east) were added to the phase I model and were re-tuned. The resulting performances (following the same approach as in phase I) are plotted in Figure 5.3. An apparent downward trend on average NRMSE can be discerned when more spatial information is provided. For all the models during this phase, the spectral radii were set to 0.6, while the leaking rates were selected to be 1.0.

For phase II, decomposition structurally reduced NRMSE for all additions of spatial information; average NRMSEs reduced in the range 47.6%–53.7%.

Phase III

Having covered phases I and II, only phase III remains, in which it is examined whether results replicate in different years and with different degrees of lag.

To determine whether the findings extrapolate to other years, WIND toolkit data for the same features and identical locations, but for the year 2012 (instead of 2011) were prepared and presented to the set of models from phase II. Their performances in this different context can be examined in Figure 5.4. Besides the observation that errors are systematically lower for both the ESN instances and the persistence model, again it can be discerned that errors decrease once more spatial information is given to the ESN models.

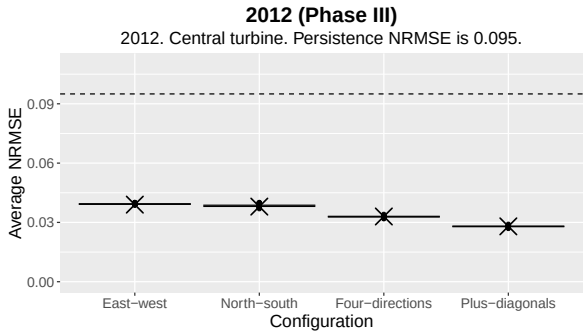


Figure 5.4: Dotplot of performances of the phase II models in the year 2012 – the first component of phase III. Note the subtle shift downwards of all errors in comparison to Figure 5.3.

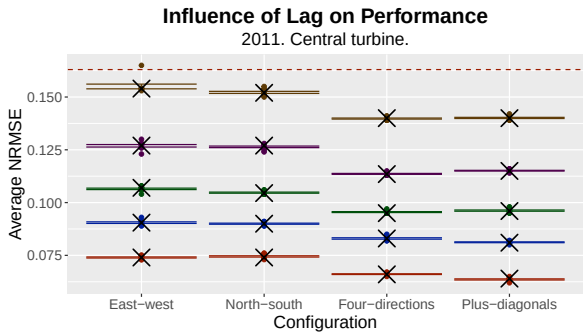


Figure 5.5: The influence of lag on model performances. Red, blue, green, purple, and brown stand for $l \in \{2, \ldots, 6\}$ (10 minutes to 30 minutes). The upper dashed line is the persistence model for $l = 2$; it is 0.163. The persistence NRMSEs for the other ls are 0.206, 0.240, 0.269, and 0.293.

Apart from varying the year of study, the lag used between input and expected output was raised from the 5-minute baseline (i.e. $l = 1$) to 5-minute increments between 10 minutes ($l = 2$) up until 30 minutes ($l = 6$; frequently defined to be the end of the 'very-short term'). As was the case under experiments for different years, models from phase II were used. The outcomes are shown in Figure 5.5. As lag increases, average NRMSEs increase for both the models as well as the persistence model: this is primarily due to increasing uncertainty effects of the future. However, for all lag scenarios, the seemingly best-performing model (the including diagonals) maintains an improvement over the persistence model's average NRMSE above 50%: even with a 30-minute lag, the reduction in average NRMSE is a $(1 - 0.140/0.293) \times -100\% \approx -52.2\%$ reduction in error (using the median of the ESN model).

For phase III, error reductions by decomposition for variations in lag laid within 37.2%–55.6%. For variations in year, these were 45%–57%.

Statistical analyses

In addition to the mentioned exploratory plots, two statistical analyses have been performed on the models' errors. Specifically, (I) an exact Wilcoxon signed-rank test for (a difference in) the median – with support for tied data – was conducted on each model's sample of average NRMSEs, comparing with the average NRMSE of the associated persistence model, and (II) a Kruskal-Wallis test, followed by a Dunn's test, was performed on phase I's wind speed data, versus the spatial addition data series of phase II.

For the former test, the null and alternative hypotheses were as follows:

$$H_0 : m \geq m_{\text{persistence}}, \text{ and}$$
$$H_A : m < m_{\text{persistence}}.$$

where m stands for the true median, and $m_{\text{persistence}}$ for the average NRMSE of the persistence model.

The tests' results are as follows. For the first four features shown within Figure 1, the actual median in average NRMSEs is not significantly less from the persistence model average NRMSE, $V = 55$, $p \approx 1.0$ (these statistics applying to these four samples individually). Thus, H_0 is not rejected for these models.

	Original	EW	NS	FD
EW	2.2168			
	0.0133			
NS	2.4338	0.2170		
	0.0075	0.4141		
FD	4.6506	2.4338	2.2168	
	0.0000	**0.0075**	**0.0133**	
PD	6.2008	3.9840	3.7670	1.5502
	0.0000	**0.0000**	**0.0001**	0.0605

Table 5.1: Outcomes of Dunn's test for the second statistical analysis (top: χ^2 statistic, below: p-value). EW, NS, FD and PD stand for east-west, north-south, four-directions, and plus-diagonals, respectively. 0.05 p-level significance in bold. 'Original' is phase I's wind speed.

In contrast, for all other samples, the actual median is significantly less from the persistence model error, $V = 0$, $p \approx 9.8 \cdot 10^{-4}$ (where again, the same statistic applies to each model individually), thereby rejecting H_0 and deeming H_A likely.

For the second analysis, comparing the wind speed data of phase I with east-west, north-south, four-directions, and plus-diagonals of phase II, the Kruskal-Wallis rank sum test tests whether all samples come from an identical distribution (H_0), or at least one group does not (H_A).

The results indicate that at least one group significantly differs from the rest in terms of average NRMSE outcomes, $\chi^2(4) = 45.707$, $p = 2.834 \cdot 10^{-9}$, rejecting H_0 and regarding H_A as likely. Hereafter, Dunn's test was applied to determine precisely which sample differences were significant. Its outcomes can be examined in Table 5.1. When comparing phase I's wind speed with the other samples (in which spatially diverse features were used), p-values lower than 0.05 can structurally be seen, suggesting addition of spatial information reduces average NRMSEs obtained.

Diagnostics

Lastly, a brief note on model diagnostics. Since echo state networks are machines that stand or fall depending on the 'quality' of their internal reservoir dynamics (among a set of other factors), four markers for performance have been tracked during testing: the actual model output, the internal node activations, the reservoir weight matrix sparseness, and the average output weight value. (The second and third diagnostic metrics are also discussed in Lukoševičius (2012), p. 670 and pp. 664–65, respectively.) All these metrics seemed to behave as expected during the testing procedure: the expected output generally follows the data, except at the beginning, where it needs to 'attenuate' to the new data, still being used to the training inputs; the internal node activations seem to form variations on linear combinations of the inputs; the internal weight matrix appears to be sparse, as well as have an equal distribution of positive- and negative-valued scalars, and the average output weight is maximally around 4.

6 Conclusion

In the context of wind speed prediction, detailed research has already been conducted, specifically on how now-popular intelligent predictors can aid in its goal. However, what input features help in solving the supervised learning task has largely remained implicit – instead of being studied explicitly. As such, the research question "What input features are informative in the very-short term wind speed prediction task?" was posed.

With the results collected, this question may be answered in the context of an echo state network (ESN) training and predicting on the WIND toolkit data from 2011 to 2012. This answer has three parts. They are as follows:

> **Part I.** Of all features, only wind speed appears to be sufficiently informative. Here, 'informative sufficiency' means: the ability to help the ESN reduce its error below reference (i.e. persistence model) levels.
>
> **Part II.** Introducing features that originate from different turbines than the one targeted for in the prediction seems to prove informative for prediction.
>
> **Part III.** As was shown at the end of the discussion of the phases in the results, decomposition of inputs into smooth and detail sub-signals appears to help in rais-

ing the quality of said signals (at least when utilising the wavelet transform, as was used here).

Of these three general findings that together answer the research question, parts II and III are largely supported by the literature, while the first is not. Specifically, using spatially diverse features to derive prediction models in the context of wind power prediction is explored in Martín-Vázquez et al. (2017), and decomposition methods are successfully used in many works, such as the wavelet transform (e.g. D. Liu et al., 2015; Wang, Lei, Zhang, Zhou, and Peng, 2019) and singular spectrum analysis (SSA; for instance Chen et al., 2019).

However, the notion that only wind speed is informative is not in agreement with results from, for example, Salcedo-Sanz et al. (2018) or the aforementioned work by Martín-Vázquez et al. (in which many diverse features are used in harmony). Three causes may be the root of the different results in this study: (I) the data, (II) the procedure, or (III) the model.

The WIND toolkit data is deemed as being an unlikely cause for the different results for two reasons. First, the data seems to originate from a respectable source (the NREL), and the data has been extensively verified, according to its authors (Draxl et al., 2015). Second, the data has been normalised before subjecting it to the ESN models; differences in magnitude and shift should have been corrected for.

A second hypothetical cause could be the procedure. Specifically, the fact that the non-wind speed features were dropped due to a greedy search among the possible features could have resulted in a premature removal of said features. However, the difference in average NRMSE reduction seemed to be quite considerable (consider again Figure 5.1), making this cause less probable.

Lastly, the ESN architecture may have been the reason for the relative ineffectiveness of the remaining input features. If this were the case, it would mean that the model somehow could not extract the 'useful' sections or patterns of these features. For instance, this may occur when such a feature largely behaves uncorrelated from the target signal. If used as input, it would function mostly as noise in the reservoir, hindering a successful build of an output weight matrix.

Apart from this open issue, the performance of the obtained models seems comparable (and even slightly better, although difference of location and year may be the cause) than that of Khanal (2018). In this work, an NRMSE of 0.1432 was obtained for a 30-minute prediction horizon, while for the 30-minute lag models here, average NRMSEs between 0.139–0.142 were collected (also see Figure 5.5). Other authors either use the RMSE instead of the NRMSE (such as in Wang, Lei, Liu, et al., 2019) or work on (typically) hourly or daily prediction horizons (as in the review by H. Liu et al., 2019), both which problematise comparison.

A couple of future lines of research may be of interest. First, the scope of the project could be raised to wind parks instead of turbines (one of the specialisations mentioned in the introduction; also see Jung and Broadwater, 2011). Moreover, more elaborate methods of constructing input features could be used, such as taking the first derivative (between sampled points); in light of the non-informative features, these methods may help, although decomposition techniques like the wavelet transform produce similar results with their detail coefficients. Lastly, since this project considered only Diamond, Missouri, it may be central to determine whether the results extrapolate to multiple sites (and thereby see whether the results are not place-bound).

To conclude, the task of wind speed prediction has once again been attempted, this time via combinations and decompositions of input features. With the insights from this study, the large-scale use of renewable energy sources is sought to be brought yet closer to reality.

References

Ahmed, S., Al-Ismail, F., Shafiullah, M., Al-Sulaiman, F., & El-Amin, I. (2020). Grid Integration Challenges of Wind Energy: A Review. IEEE *Access*, *8*, 10857–10878. (ISSN: 21693536) doi: 10.1109/ACCESS.2020.2964896

Chen, Y., He, Z., Shang, Z., Li, C., Li, L., & Xu, M. (2019). A novel combined model based on echo state network for multi-step ahead wind speed forecasting: A case study of NREL. *En-*

ergy conversion and management, *179*, 13–29. doi: 10.1016/j.enconman.2018.10.068

Daubechies, I. (1992). *Ten Lectures on Wavelets* (2nd ed.). Philadelphia, United States: Society for Industrial and Applied Mathematics (SIAM). doi: 10.1137/1.9781611970104

Diógenes, J. R. F., Claro, J., Rodrigues, J., & Loureiro, M. V. (2020, February 1). Barriers to Onshore Wind Energy Implementation: A Systematic Review. *Energy Research & Social Science*, *60*, 1–33. (ISBN: 2214-6296) doi: 10.1016/j.erss.2019.101337

Draxl, C., Clifton, A., Hodge, B.-M., & McCaa, J. (2015, August 1). The Wind Integration National Dataset (WIND) Toolkit. *Applied Energy*, *151*, 355–366. doi: 10.1016/j.apenergy.2015.03.121

Foley, A. M., Leahy, P. G., Marvuglia, A., & McKeogh, E. J. (2012, January). Current Methods and Advances in Forecasting of Wind Power Generation. *Renewable Energy*, *37*(1), 1–8. (ISSN: 0960-1481) doi: 10.1016/j.renene.2011.05.033

Guyon, I., & Elisseeff, A. (2003, March). An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research*, *3*, 1157–1182. Retrieved from `https://dl.acm.org/doi/abs/10.5555/944919.944968`

Jaeger, H. (2001). *The "Echo State" Approach to Analysing and Training Recurrent Neural Networks – With an Erratum Note* (Technical Report No. 148). Bonn, Germany: German National Research Center for Computer Science (GMD). Retrieved from `http://minds.jacobs-university.de/uploads/papers/EchoStatesTechRep.pdf` (This version of the report – actually from 2010 – is a revised version, provided with errata notes.)

Jung, J., & Broadwater, R. P. (2011, April). Current Status and Future Advances for Wind Speed and Power Forecasting. *Renewable and Sustainable Energy Reviews*, *31*, 762–777. (ISSN: 1364-0321) doi: 10.1016/j.rser.2013.12.054

Khanal, A. (2018, May 16). *Wind Speed Prediction Using Echo State Network.* Unpublished Bachelor's Thesis, Jacobs University, Bremen, Germany. (In consultation with the supervisor, a copy of this thesis can be requested.)

Lee, G. R., Gommers, R., Waselewski, F., Wohlfahrt, K., & O'leary, A. (2019). PyWavelets: A Python Package for Wavelet Analysis. *Journal of Open Source Software (JOSS)*, *4*(36), 1–2. (Is accessible as the 1237th paper in the journal.) doi: 10.21105/joss.01237

Liu, D., Wang, J., & Wang, H. (2015). Short-term Wind Speed Forecasting Based on Spectral Clustering and Optimised Echo State Networks. *Renewable Energy*, *78*, 599–608. doi: 10.1016/j.renene.2015.01.022

Liu, H., & Chen, C. (2019). Data Processing Strategies in Wind Energy Forecasting Models and Applications: A Comprehensive Review. *Applied Energy*, *249*, 392–408. doi: 10.1016/j.apenergy.2019.04.188

Liu, H., Chen, C., Lv, X., Wu, X., & Liu, M. (2019, September 1). Deterministic Wind Energy Forecasting: A Review of Intelligent Predictors and Auxiliary Methods. *Energy Conversion and Management*, *195*, 328–345. (ISSN: 0196-8904) doi: 10.1016/j.enconman.2019.05.020

Liu, H., & Yu, L. (2005). Toward Integrating Feature Selection Algorithms for Classification and Clustering. *March*, *17*(4), 491–502. doi: 10.1109/TKDE.2005.66

Lukoševičius, M. (2012). A Practical Guide to Applying Echo State Networks. In G. Montavon, G. B. Orr, & K.-R. Müller (Eds.), *Neural Networks: Trick of the Trade* (Vol. 7700, pp. 659–686). Heidelberg, Berlin: Springer Verlag. (ISBN: 978-3-642-35288-1, part of the *Lecture Notes in Computer Science* (LNCS) book series.) doi: 10.1007/978-3-642-35289-8_36

Martín-Vázquez, R., Aler, R., & Galván, I. M. (2017). A Study on Feature Selection methods for Wind Energy Prediction. In *International work-conference on artificial neural networks* (pp. 698–707).

Panwar, N. L., Kaushik, S. C., & Kothari, S. (2011, April). Role of Renewable Energy Sources in Environmental Protection: A Review. *Renewable and Sustainable Energy Reviews*, *15*(3), 1513–1524. (ISSN: 1364-0321) doi: 10.1016/j.rser.2010.11.037

Park, J. Y. (2014, November). Assessing Determinants of Industrial Waste Reuse: The Case of

Coal Ash in the United States. *Resources, Conservation and Recycling*, *92*, 116–127. (ISSN: 0921-3449) doi: 10.1016/j.resconrec.2014.09.004

Qian, Z., Pei, Y., Zareipour, H., & Chen, N. (2019, February 1). A Review and Discussion of Decomposition-based Hybrid Models for Wind Energy Forecasting Applications. *Applied Energy*, *235*, 939–953. (ISSN: 0306-2619) doi: 10.1016/j.apenergy.2018.10.080

Salcedo-Sanz, S., Cornejo-Bueno, L., Prieto, L., Paredes, D., & García-Herrera, R. (2018). Feature Selection in Machine learning Prediction Systems for Renewable Energy Applications. *Renewable and Sustainable Energy Reviews*, *90*, 728–741.

Sinsel, S. R., Riemke, R. L., & Hoffmann, V. H. (2020, January). Challenges and Solution Technologies for the Integration of Variable Renewable Energy Sources—A Review. *Renewable Energy*, *145*, 2271–2285. (SSIN: 0960-1481) doi: 10.1016/j.renene.2019.06.147

Soman, S. S., Zareipour, H., Malik, O., & Mandal, P. (2010, September 26–28). A Review of Wind Power and Wind Speed Forecasting Methods with Different Time Horizons. In *North American Power Symposium 2010* (pp. 1–8). doi: 10.1109/NAPS.2010.5619586

United Nations, Department of Economic and Social Affairs, Population Division. (2019, June). *World Population Prospects 2019: Ten Key Findings* (Yearly report). New York, United States. Retrieved 2020–02–29, from `https://population.un.org/wpp/Publications/Files/WPP2019_10KeyFindings.pdf`

Veers, P., Dykes, K., Lantz, E., Barth, S., Bottasso, C. L., Carlson, O., ... Wiser, R. (2019). Grand Challenges in the Science of Wind Energy. *Science*, *366*(6464). (ISSN: 0036-8075. The e-location ID of this item is `eaau2027`.) doi: 10.1126/science.aau2027

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., ... Contributors, S. . . (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, *17*, 261–272. doi: https://doi.org/10.1038/s41592-019-0686-2

Wang, H., Lei, Z., Liu, Y., Peng, J., & Liu, J. (2019). Echo State network Based Ensemble Approach for Wind power Forecasting. *Energy Conversion and Management*, *201*, 1–15. doi: 10.1016/j.enconman.2019.112188

Wang, H., Lei, Z., Zhang, X., Zhou, B., & Peng, J. (2019). A Review of Deep Learning for Renewable Energy Forecasting. *Energy Conversion and Management*, *198*, 1–16. doi: 10.1016/j.enconman.2019.111799

A Parameter Settings

In this appendix, the parameter settings for the various models mentioned in the thesis are described.

Unless otherwise specified, standard parameters listed in Table A.1 apply to the listed configurations.

During phase I, step 1, a grid search was performed to determine which parameter combination worked best for each of the individual features, before running separate tests with the final configuration. These final configurations are shown in Table A.2.

For the second step of phase I, wind speed was supplied in the form shown in Table A.2 (that is, in its decomposed form, with the same input weight scales). The other features were individually added to this wind speed feature following the configurations in Table A.3, following a similar grid procedure as in step 1. Importantly, they were not decomposed here. Also see the table caption.

In phase II, as the number of parameter combinations grew larger due to the number of sites to consider, no grid search was performed. Instead, combinations were tested manually, starting from the configurations of phase I. Moreover, spatially remote features were each decomposed by a 4-moments Daubechies wavelet, with 6 levels of decomposition (so as to more extensively smooth the data). Of the decomposition, sub-signals $\langle s_1, s_2 \rangle$ and $\langle s_3, s_4 \rangle$ were used for reconstruction. Then, these were prepended before the central turbine's decomposition of four sub-signals (identical to those of phase I), and all were input weight scaled according to Table A.4. Other parameters stayed as were specified in the 'standard parameters' table (Table A.1).

An example may clarify this procedure. Consider the addition of the eastern and western turbines (as in the first row of Table A.4). Both the wind speed signals for the eastern and western turbine were decomposed according to the aforementioned wavelet transform. Of all sub-signals, the first two ($\langle s_1, s_2 \rangle$) and the third and fourth ($\langle s_3, s_4 \rangle$) were selected to reconstruct into two reconstructed signals – call these r_1, r_2. This is done for both the eastern and western turbine, thus yielding $r_{1,\text{east}}$, $r_{2_{\text{east}}}$, $r_{1,\text{west}}$ and $r_{2,\text{west}}$. These signals are input scaled and sent into the network (columns 1–4 of Table A.4), besides the four central turbine reconstructed time series from phase I (columns 5–8). In a similar manner, four directions leads to $2 \times 4 + 4 = 12$ columns, or inputs (4 distant turbines, each yielding two reconstructed, smoothed signals r_1, r_2).

To close, phase III used parameters identical to phase II; the year and degree of lag was instead changed: tuning for these new situations did not appear to be contributing to a further error decrease.

Parameter	Description	Value
Reservoir size (N)	The number of nodes present in the reservoir.	200
Connectivity (c)	The fraction of non-zero entries in the $\mathbf{W}$ weight matrix.	1/20
Shifts	A vector of vertical shifts to apply to inputs.	$\mathbf{0}$
Noise (ν)	The scalar for dynamic unif$(-1, 1)$ noise to add to reservoir updates.	0
Tichonov coefficient (β)	A ridge regression regularisation coefficient.	1/20
Bias level ($\mathbf{b}$)	A static unif$(-1, 1)$ noise term to add to reservoir updates.	1/5
Transient period ($n_{\text{transient}}$)	The number of time steps to drop before linear regression.	600
$\mathbf{f}$	A squashing function to use in reservoir updates.	$\tanh(\cdot)$
$\mathbf{f}^{\text{out}}$	A squashing function to use for transfer to output.	$\mathbf{1}(\cdot)$
Regression mode	The approach to regression taken.	Tichonov
s_{back}	The scalar for $\mathbf{W}^{\text{back}}$.	0

Table A.1: Standard parameter values. Here, unif(a, b) stands for uniform noise sampled from the real-valued range $[a, b]$, $\mathbf{1}(\cdot)$ for the identity function, and 'Tichonov' for ridge regression with Tichonov regularisation via parameter β.

Parameter	Density	Direction	Power	Pressure	Speed	Temperature
Spectral radius	0.7	0.6	0.6	0.6	0.6	0.6
Leaking rate	0.9	0.9	0.9	1.0	0.9	0.9
Input weight scale I	0.2	0.0	0.1	0.8	0.0	0.2
Input weight scale II	0.2	0.8	0.8	–	0.7	0.2
Input weight scale III	0.2	0.2	0.1	–	0.2	0.2
Input weight scale IV	0.2	0.0	0.0	–	0.05	0.2
Input weight scale V	0.2	0.0	0.0	–	0.05	0.2

Table A.2: Parameters for phase I, step 1. The column names refer to the same parameters as listed in Figure 5.1. Moreover, when more than one input scale was used, the last four scales refer to scalings for decomposed signals via a Daubechies wavelet with two moments and four levels of decomposition. (So, the complete, non-decomposed signal is included at index one.)

Parameter	Density	Direction	Power	Pressure	Temperature
Spectral radius	0.6	0.6	0.6	0.6	0.6
Leaking rate	1.0	1.0	1.0	1.0	1.0
Input weight scale I	0.05	0.05	0.05	0.05	0.10
Input weight scale II	0.70	0.70	0.70	0.70	0.70
Input weight scale III	0.20	0.20	0.20	0.20	0.20
Input weight scale IV	0.05	0.05	0.05	0.05	0.05
Input weight scale V	0.05	0.05	0.05	0.05	0.05

Table A.3: Parameters for phase I, step 2. Input weight scale I refers to the non-decomposed feature, while the latter scales refer to the decomposed wind speed signal, using its input scalings from Table A.2.

	Input weight scales · All values ×100																			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
EW	18	1	18	1	70	10	1	1												
NS	Identical to EW.																			
FD	20	1	20	1	20	1	20	1	50	2	1	1								
PD	11	1	11	1	11	1	11	1	11	1	11	1	13	1	11	1	11	1	1	1

Table A.4: Parameters for phase II. EW up until PD stand for 'east-west' to 'plus-diagonals'; for all abbreviations see Table 5.1's caption. All values are multiplied by 100 for compactness (so in reality, we would have $0.18, 0.01, 0.18, \ldots$).