Conditional Forecasting of Water Level Time Series with RNNs

Bart J. van der Lugt(✉) and Ad J. Feelders

Department of Information and Computing Sciences, Utrecht University, Princetonplein 5, 3584 CC Utrecht, The Netherlands
b.j.vanderlugt@gmail.com

Abstract. We describe a practical situation in which the application of forecasting models could lead to energy efficiency and decreased risk in water level management. The practical challenge of forecasting water levels in the next 24 h and the available data are provided by a dutch regional water authority. We formalized the problem as conditional forecasting of hydrological time series: the resulting models can be used for real-life scenario evaluation and decision support. We propose the novel *Encoder/Decoder with Exogenous Variables* RNN (ED-RNN) architecture for conditional forecasting with RNNs, and contrast its performance with various other time series forecasting models. We show that the performance of the ED-RNN architecture is comparable to the best performing alternative model (a feedforward ANN for direct forecasting), and more accurately captures short-term fluctuations in the water heights.

Keywords: Time series · Conditional forecasting · Encoder/Decoder · Exogenous variables · Recurrent Neural Network

1 Introduction

In the Netherlands, water is all around us: knowing how to manage this water is key to sustaining our way of life. New technologies and an exponential increase in the amount of data available generate new possibilities in the field of hydrology. Accurate forecasts of weather, water levels and flow rates allow water boards (Waterschappen) to limit risk of flooding, drought damage and energy waste. Water boards are regional government bodies responsible for water quality, water levels and safety.

From this practice, the relevance of accurate *conditional forecasts* of time series data becomes especially clear. Conditional forecasts are a useful means of evaluating the impact of a hypothetical scenario. The goal is to predict the variables of interest conditioned on an *assumed future path* of one or more other variables in the system. These forecasts can be used to guide the decision making process by comparing various scenarios.

Made possible by Ynformed and Waterschap Zuiderzeeland.

V. Lemaire et al. (Eds.): AALTD 2019, LNAI 11986, pp. 55–71, 2020.
https://doi.org/10.1007/978-3-030-39098-3_5

Artificial Neural Network (ANN) models have become very popular in forecasting water level time series. They provide a good alternative to the traditional time series models (such as vector autoregressive (VAR) models) because ANNs do not assume linear dependencies. Recurrent Neural Networks (RNNs) were specifically designed to process sequential data, allowing for the model to retain short-term and long-term memory of the input series, thereby improving the performance of traditional feedforward ANNs on many machine learning tasks. Despite their great potential, the application of RNNs to conditional time series forecasting has not been extensively studied in the literature.

The method we propose in this paper is to use an Encoder/Decoder RNN architecture to generate conditional forecasts of time series data. Our approach distinguishes itself by considering the future path of the variables we conditioned on in the decoding step. It can therefore be characterized as an *Encoder/Decoder RNN with Exogenous Variables*. The resulting architecture is very flexible and can be used to model many real-world time series. In our experiments, we compare the performance of the ED-RNN with various other popular forecasting models in a conditional forecasting scenario. This forecasting problem, as well as the dataset of hydrological variables that is used for training and testing, is provided by the Waterschap Zuiderzeeland.

The rest of this paper is structured as follows. First, we present relevant literature regarding hydrological time series forecasting. We then provide a more in-depth description of our forecasting challenge, its relevance, and the various solutions to it. Lastly, we detail the results of our experiments and discuss their implications for the field.

2 Related Work

Traditionally, most of the research in water resource systems was done using hydrological models, which require significant amounts of domain knowledge. Studies showed that levels [19,24], flow dynamics [18] and quality [9] of surface water could be effectively modelled with this approach. However, a comparative study of such models for ground water by Konikow et al. [15] showed that it was impossible to scientifically verify and validate them, and argued that calibration procedures generate non-unique solutions with limited predictive accuracy.

Research into our geographical region of interest (the Noordoostpolder in Flevoland, The Netherlands) has been conducted in 2006 [5]. In this evaluation study, the authors studied the historical influx of water to the region, and contrasted it with its optimal value for different performance indicators using various hydrological models.

Modelling hydrological variables can also be done by applying time series models to a collection of historical data. Autoregressive (AR) models [16] fit a linear equation of previous measurements to predict the next value. The AR model is the special case of the Vector Auto Regressive (VAR) model [14], which is suited for multivariate instead of univariate time series. These models can be easily extended to include external regressors, moving averages, differencing, trends and seasonal components [16]. Limitation of VAR models include that they assume linear dependence between the variables, as well as linear dependence over time. Additionally, early studies [16, 26] concluded that VAR models are unable to learn both short- and long-term dependencies between variables in a single model.

More recently, Artificial Neural Network (ANN) models have become very popular in forecasting hydrological variables, especially for water levels. An important advantage is that an ANN model creates a non-linear mapping from input to output. The challenge in using feedforward ANNs for time series data is finding a suitable way of incorporating past observations into the model, such that the temporal correlation is utilized. A comparative study of ANN models for groundwater level forecasting was conducted by Yoon et al. [25]. Their solution to modelling the time series data was to *lag* the variables, i.e., using a time-delayed version of the data as additional input. Another approach to modelling time series data is by applying a Fourier Transformation to the input space, which was first proposed and applied by Wang et al. [23]. Tiwari et al. [22] used a similar approach, using wavelet transforms, to develop an accurate model for predicting floods in an Indian river basin up to 10 h ahead.

Recurrent Neural Networks (RNNs) were specifically designed to process sequential data: their architecture allows them to consider network output of previous time steps in later iterations [6], thus retaining memory. The difficult task of learning both short-term and long-term dependencies without losing efficiency is a well-studied problem [3, 13]. Various RNN architectures were compared and successfully applied by Groenen [7] for seasonality extraction, residual learning and accurate prediction of wastewater inflow at municipal wastewater treatment plants.

There are generally two approaches to forecasting multiple subsequent observations of a time series. A recursive forecasting (sometimes also called iterative forecasting) model repeatedly generates a prediction for a single period ahead, using a fixed window of observations and previous predictions. A direct forecasting model generates a prediction for variables multiple time steps ahead, and this predictions is independent of the (predictions of) observations in between. Various studies compared the performance of both approaches using ANNs, but the results are mixed and thus inconclusive [10, 12, 17].

3 Methodology

In this section, we first describe the conditional forecasting problem and the time series data that is available. Then, we describe several time series modelling approaches from literature, that we use to generate conditional forecasts for our data. Lastly, we propose our own method for conditional forecasting called the Encoder/Decoder RNN with Exogenous Variables.

3.1 Problem Description

The water management system in the Noordoostpolder is governed by employees of the Waterschap Zuiderzeeland. Their job is to manage the influx and efflux of water into the region. The influx of water to the polder consists of surface water from surrounding regions, groundwater seepage and precipitation. Water inside the polder flows along channels to one of 3 large pumping stations responsible for the efflux, which are operated manually.

Hydrologists at Waterschap Zuiderzeeland made an assessment of vital locations in the region based on connectivity, distance, flow directions and management practice. Surface water level series at 3 different locations were pointed out as target variables for the conditional forecast: predictions for these locations are informative and useful for the water managers. Besides this, it was decided that ground water level series for 2 locations, precipitation, and flow rates for the 3 large pumping stations would be the main descriptive variables for generating the prediction. Figure 1 shows a map of the Noordoostpolder, with the vital water system locations. Table 1 contains identifiers for each of the measurement series and a description of the location where these measurements were recorded.

Fig. 1. Vital locations in the water management system of the Noordoostpolder. Numbers indicate water level measurement locations, letters indicate pumping stations. See Table 1 for a description of some of these points.

We derive using domain knowledge that ground and surface water measurements should be considered endogenous variables, because they are dependent upon each other and on other variables. Flow rates and precipitation can be treated as exogenous variables. Precipitation is the influx of water to the Noordoostpolder and is therefore independent of the water levels. Flow rates measure the amount of water that is displaced by the pumping stations and are defined by the pumping protocol. The available data consists of measurements every 15 min for all time steps between 2015 and 2018.

The conditional forecasting problem is formalized as follows. Let us denote the multivariate time series of length T with n endogenous variables as $\mathbf{y}_t \in \mathbb{R}^n$,

Table 1. Names for important measurement locations in this work, along with their official identifier defined by Waterschap Zuiderzeeland, type of measurement recorded at the location, and an indication of its position in the map of the Noordoostpolder shown in Fig. 1

Name	Identifier	Type	Unit	Location
WH2	NOP.PM4810.LT1	Surface water	cm NAP	2
WH3	MP6002	Surface water	cm NAP	3
WH5	NOP.ST4775.LT3	Surface water	cm NAP	5
GW1	21BN.093.01	Ground water	cm NAP	Near C
GW2	15HN.018.01	Ground water	cm NAP	Between 1 and A
FR2000	NOP.2000_TOT	Flow rate	m^3/hour	A
FR2100	NOP.2100_TOT	Flow rate	m^3/hour	C
FR2200	NOP.2200_TOT	Flow rate	m^3/hour	B
MWSP	KNMI Marknesse	Precipitation	mm/hour	Near 4

and the time series with m exogenous variables $\mathbf{x}_t \in \mathbb{R}^m$ for $0 \leq t < T$. Given a time step O called the *forecasting origin* and a parameter $h > 0$ called the *forecasting horizon*, we want a prediction for observations y_{O+1}, ... y_{O+h}. The model generating these predictions should use y_t, ... y_O, x_t, ... x_O as well as x_{O+1}, ... x_{O+h} as input, for some $t < O$. This generates a forecast of $\mathbf{y}$ conditioned on the future path of $\mathbf{x}$.

In our experiments, we split the time series data in two parts. The training set consist of all observations in 2015, 2016 and 2017, the test set of all observations in 2018. The former is used to tune the model parameters, the latter is only used to realistically evaluate the forecasting performance. For forecast evaluation, we choose forecast origins O_i = 12:00:00 (midday) for each day i in out test set, and we choose $h = 96$, which corresponds to forecasts up to 24 h ahead. The target variables are series WH2, WH3 and WH5, since the surface water levels provide relevant insight for the Waterschap Zuiderzeeland.

The modelling choices described above, made in consultation with Waterschap Zuiderzeeland, ensure that the forecasting models are useful and meaningful for the hydrologists and managers. The selected measurement series can be made accessible online, ensuring up-to-date forecasts. The choice of exogenous variables ensures that future paths can be easily constructed, using an assumption about actions of pumping station managers. By generating conditional forecasts for up to 24 h in the future, hydrologists can compare and evaluate various scenario's, and use these to adjust and improve their operations for the next day.

3.2 Forecasting Models

The first method we use to forecast the values in $\mathbf{y}_t$ is using VAR and VECM models [16], which are commonly used in time series analysis and are characterized by the assumption of linear dependencies. These models produce recursive forecasts, and their most important parameter is the lag parameter p for defining the number of past observations of $\mathbf{y}_t$ and $\mathbf{x}_t$ that are taken into account.

As a generalization of these linear models, we consider the feedforward Artificial Neural Network (ANN) [11], also called Multilayer Perceptron (MLP). The generic model architecture is used to learn a non-linear mapping from some input vector $\mathbf{I}$ to output $\mathbf{O}$. It consists of k layers L_i, with $i = 1, \ldots, k$, having l_i neurons in each layer. Neurons in the input layer are activated using input data, simply using $L_1 = \mathbf{I}$. Neurons in consecutive layers are activated through weighted connections with the previous layer, as follows:

$$L_{i+1} = \alpha(\mathbf{W}_i L_i + b_i) \qquad \text{for } i = 1, \ldots, k-1 \tag{1}$$

where L_i and L_{i+1} are vectors of size $l_i \times 1$ and $l_{i+1} \times 1$ respectively, $\mathbf{W}_i$ is an $l_{i+1} \times l_i$ matrix containing coefficients (weights) and b_i is a vector of size $l_{i+1} \times 1$ containing coefficients. The function $\alpha : \mathbb{R} \to \mathbb{R}$ is some non-linear *activation function* that is applied element-wise. The above is called the *feedforward step*, producing the output L_k: using a learning algorithm such as gradient descent backpropagation [11], the error between L_k and the desired output $\mathbf{O}$ is iteratively decreased.

We apply the generic feedforward ANN to time series forecasting in two ways. Recursive forecasting can be done using $\mathbf{I} = \{\mathbf{y}_{t-p}, \ldots \mathbf{y}_{t-1}, \mathbf{x}_{t-p}, \ldots \mathbf{x}_t\}$ as input, and $\mathbf{O} = \{\hat{\mathbf{y}}_t\}$. We thus have $l_1 = p(n+m)+m$ and $l_k = n$. This network can now be used recursively, using previous predictions as input, to generate forecasts for time steps $t+1, \ldots t+h-1$. Alternatively, we can implement an ANN for direct forecasting by choosing $L_1 = \{\mathbf{y}_{t-p}, \ldots \mathbf{y}_{t-1}, \mathbf{x}_{t-p}, \ldots \mathbf{x}_{t+h-1}\}$ as input, and $L_k = \{\hat{\mathbf{y}}_t, \ldots \hat{\mathbf{y}}_{t+h-1}\}$. In this case, we have $l_1 = p(n+m) + hm$ and $l_k = hn$. This network is not used recursively, but instead generates predictions for all desired time steps using a single feedforward operation.

Finally, we consider Recurrent Neural Networks (RNNs) and look at two common ways in which we can use these to generate conditional forecasts. RNNs provide an extension to feedforward ANNs, in that they are specifically designed to process sequences of observations. Instead of concatenating many observations into a single input vector like the ANN, the RNN takes a multivariate time series $\mathbf{I}_t \in \mathbb{R}^m$ for $1 \leq t \leq T$ as input, mapping it to the output series $\mathbf{O}_t \in \mathbb{R}^n$ by processing each time step t individually. What also distinguishes the RNN architecture is that a layer of RNN neurons uses the layer's activations at the previous time step as additional input. A single RNN layer L_{i+1} thus computes:

$$L_{i+1,t} = \alpha(\mathbf{W}_1 \sigma(L_{i+1,t-1}) + \mathbf{W}_2 L_{i,t} + b) \qquad \text{for } t = 1, \ldots, T \tag{2}$$

where the function $\sigma : \mathbb{R} \to \mathbb{R}$ is an element-wise function and the other terms are similar to the feedforward ANN case. The term $\sigma(L_{i,t})$ is called the hidden state of the i'th network layer: its initial state $\sigma(L_{i,0})$ can be specified by the user, set to 0 or be learned.

Learning the weights for an RNN could lead to several problems: since errors must be propagated over many time steps, the gradients could either *vanish* or *explode* (i.e., extreme decrease or increase in the norm). To counteract this, specialized RNN neuron architectures have been developed, such as the *Long Short-Term Memory* (LSTM) [6] and the *Gated Recurrent Unit* (GRU) [4].

Conditional time series forecasting using an RNN is not straightforward. The most common solution is to define an RNN that generates 1-ahead predictions, as seen in [8]. For this model, we choose $\mathbf{I}_t = \{\mathbf{y}_{t-1}, \mathbf{x}_t\}$ and $\mathbf{O}_t = \{\mathbf{y}_t\}$. However, for multi-step ahead forecasting, this recursive approach is counter intuitive. The RNN uses as input both its previous prediction and its previous hidden state: this could distort the memory management of the RNN and lead to extreme accumulation of errors for large h.

The second approach is to use $\mathbf{I}_t = \{\mathbf{y}_{t-h}, \mathbf{x}_t\}$ and $\mathbf{O}_t = \{\mathbf{y}_t\}$. This way, the RNN is not applied recursively and its memory is efficiently used. However, the input series is now composed of variables with substantial time shifts, which could lead to problems. To prevent the extraction of misleading patterns, we are required to use a moving average series of $\mathbf{y}_t$ instead of the actual measurements, resulting in a loss of information. Additionally, this approach forces us to use input sequences of $\mathbf{y}$ and $\mathbf{x}$ of the same length and we can not use $\mathbf{x}_t$ for $t < T$.

3.3 Encoder/Decoder RNN with Exogenous Variables

The above suggests that conditional forecasting using RNNs does not work very well, so we propose our own architecture for this problem. Requirements are that it allows us to use input sequences of $\mathbf{y}$ and $\mathbf{x}$ of varying length without time shifts, and that the memory management of the RNN is used efficiently during forecasting.

Our proposed approach to conditional forecasting with RNNs can be described as an *encoder/decoder approach with exogenous variables*. It consist of two steps: first, given origin O, horizon h and lag p, we encode the observations in $\mathbf{x}$ and $\mathbf{y}$ at times $O - p \leq t < O$ using an RNN layer of size l_1. This can simply be done using $\mathbf{I}^1_t = \{\mathbf{y}_t, \mathbf{x}_t\}$, since all these observations are known. We only keep the last output of this RNN layer, instead of the entire series. The resulting output, which we call the encoding E, is a vector of values of size $l_1 \times 1$.

The second step is to turn the output of the first RNN layer, E, into predictions for $\mathbf{y}_{O+1}$, ..., $\mathbf{y}_{O+h}$. We do this by concatenating copies of the encoding to each time step in $\mathbf{x}_{O+1}$, ..., $\mathbf{x}_{O+h}$: the result is a new time series $\mathbf{z}_{O+1}$, ..., $\mathbf{z}_{O+h}$, with $\mathbf{z}_t = \{\mathbf{x}_t, E\}$. We use this combined time series as input to the second RNN layer, thus we write $\mathbf{I}^2_t = \{\mathbf{x}_t, E\}$. This second RNN layer produces the predictions for $\mathbf{y}_{O+1}$, ..., $\mathbf{y}_{O+h}$, thus we write $\mathbf{O}_t = \{\mathbf{y}_t\}$. While the architecture appears to consist of two separate models, the encoder can not be trained without a decoder. In summary, our architecture generates predictions $\mathbf{O}_O$, ... $\mathbf{O}_{O+h-1}$ using an encoding of $\mathbf{I}^1_{O-p}$, ... $\mathbf{I}^1_{O-1}$ and using $\mathbf{I}^2_O$, ... $\mathbf{I}^2_{O+h-1}$ as additional information during decoding. Figure 2 contains a simplified, schematic representation of this RNN architecture.

The procedure of encoding and decoding a time series using RNNs is also called *sequence to sequence* (seq2seq) modelling, as proposed by Sutskever et al. [21]. In that work, it was not applied to measurement time series, but to words: translating sentences from one language to another, with good results.

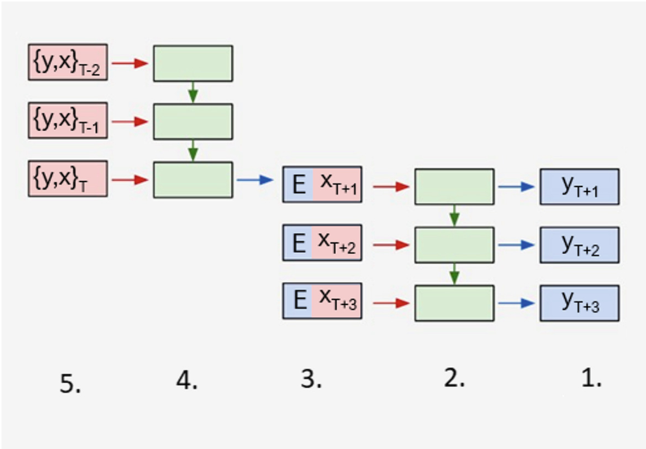


Fig. 2. Schematic representation of the Encoder/Decoder with Exogenous Variables RNN structure, with input data before the forecast horizon (5), first RNN layer (4), concatenation of exogenous variables after the forecast horizon and the copies of the encoding (3), second RNN layer (2) and the generated forecasts (1).

The most important difference with our application, and therefore the novelty of our approach, is that the initial seq2seq modelling was not conditional: there was no notion of exogenous values $\mathbf{x}_{O+1}$, ..., $\mathbf{x}_{O+h}$, but instead, only the copies of the encoding E were used as input to the second RNN layer.

4 Experimental Setup

As mentioned before, we use two parts of the available data: observations in 2015, 2016 and 2017 for tuning the model parameters, observations in 2018 to evaluate forecasts. The forecasting approach described, using O_i = 12:00:00 (midday) for each day i in 2018 and $h = 96$, results in a series of predictions $\hat{\mathbf{y}}$ for all the water levels $\mathbf{y}$ in the testing set. We choose to evaluate the forecasts by determining the Root Mean Squared Error (RMSE) and Mean Average Percentage Error (MAPE) [2] of $\hat{\mathbf{y}}$ and $\mathbf{y}$. We also determine the squared Pearson correlation coefficient of $\hat{\mathbf{y}}$ and $\mathbf{y}$, sometimes referred to as the 'pseudo-R^2', which we shall simply reference as R^2. We show the results for the surface water level series WH2, WH3 and WH5, since these are the target variables of this study.

Methods for estimating VAR and VECM models, as well as generating their predictions on new data, are provided in the R package 'tsDyn'[20]. The package does not support the inclusion of lags of exogenous variables: the exogenous data thus had to be temporally transformed, creating new columns with time-delayed measurements.

All methods for defining and training ANN and RNN models are implemented in the 'Keras' python package, which can be controlled from the R environment using the R package called 'Keras' [1]. Implementing the feedforward

ANN models required temporal transformation of the data. To speed up training, measurements were normalized to the interval [0,1] using min-max scaling. Keras automatically uses the last 20% of observations in the training set as validation set: to prevent overfitting we defined an early stopping criterion based on the performance on the validation set. No other regularization techniques were used to generate the results in this paper.

To be able to efficiently reference the neural network model architectures, we introduce the following notation. 1-NN$_{i,j}(p)$ is used to describe a neural network generating 1-step ahead predictions, with two hidden layers of sizes i and j, and lag order p for the endogenous variables. Neural networks with only 1 hidden layer have no parameter j. D-NN$_{i,j}(p)$ is used to describe a direct forecasting ANN, with the same parameters. We reference the RNN architectures in a similar fashion. First, models are referenced by the types of neurons in the layers, namely, "LSTM" or "GRU". 1-LSTM$_{i,j}$ indicates an LSTM model that is trained for 1-ahead predictions with two layers of sizes i and j. XY-LSTM$_{i,j}$ is used to reference an LSTM model that implements the second RNN approach described: predicting $\mathbf{y_t}$ using $\mathbf{I}_t = \{\text{Mean}(\mathbf{y}_{t-h-192}, \ldots \mathbf{y}_{t-h}), \mathbf{x}_t\}$. If either of the two models above has only one layer, the variable j is omitted. Lastly, ED-LSTM$_{i,j}$ is used to reference an LSTM model that implements the encoder/decoder architecture with exogenous variables, which always requires at least two RNN layers.

5 Results

We begin by determining a baseline performance. This naive forecast is defined as follows: for each forecasting horizon T_i = 12:00:00 (midday) for every day i, we predict that the water levels in the next 24 h are equal to the average water level of the past 24 h. In Table 2, we show this baseline performance, expressed RMSE, MAPE and R^2.

Also in Table 2 are the RMSE, MAPE and R^2 of various VAR(p) and VECM(p) models. We observe that increasing the lag order p, and thereby increasing the model complexity, leads to a better performance. We also observe that even with the simplest VAR model, we are able to greatly improve the naive baseline performance.

Next, we evaluate the feedforward ANN architectures for both recursive and direct forecasting. We chose to present the models with lag order $p = 96$: lower values for p yielded worse results, and results for higher values of p were comparable or worse. The RMSE, MAPE and R^2 for the predictions on the test set are shown in Table 3. These ANN models are non-linear and contain many more parameters than the VAR models: the NN model complexity is therefore much higher. We observe that generating the recursive forecasts using ANNs yields bad results: it hardly improves the baseline model. This can likely be attributed to the accumulation of errors when forecasting multiple time steps ahead: for NN models, these negative effects are more extreme compared to the VAR/VECM models, due to the increased model complexity.

Table 2. Forecasting accuracy expressed as RMSE, MAPE and R^2 for the baseline predictor and various VAR(p) and VECM(p) models evaluated on the test set. In bold: for each target variable, its highest quality prediction with respect to RMSE, MAPE or R^2.

Model	WH2			WH3			WH5		
	RMSE	MAPE	R^2	RMSE	MAPE	R^2	RMSE	MAPE	R^2
Baseline	4.94	.0067	.45	5.17	.0064	.78	5.45	.0069	.37
VAR(8)	3.36	.0040	.84	3.65	.0043	.9	3.5	.0038	.77
VECM(8)	3.23	.0039	.83	3.51	.0040	.91	3.35	.0035	.78
VECM(24)	3.15	.0037	.84	3.35	.0038	**.92**	3.33	.0035	.78
VECM(48)	2.85	.0032	.86	3.26	**.0035**	**.92**	3.33	**.0034**	.79
VECM(96)	2.66	.0029	.86	3.28	.0036	.91	3.3	.0035	.78
VAR(192)	**2.50**	**.0028**	**.87**	3.22	.0037	**.92**	**3.13**	.0035	.78
VECM(192)	2.60	.0029	**.87**	**3.18**	**.0035**	**.92**	3.15	**.0034**	**.80**

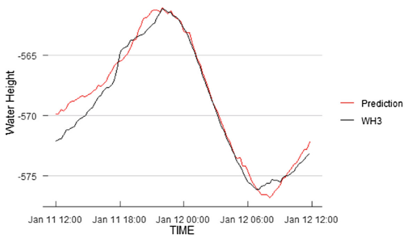
We conclude from the D-NN results that generating direct forecasts using ANNs improves the performance of the VAR/VECM models, in particular for WH5. Nevertheless, the VAR/VECM model performance proves to be a reasonable baseline for model comparison. We also generated visualizations of the D-NN_{96}(96) predictions for an informative subset of observations: these are shown in Fig. 3. We observe that the minimal and maximal water levels are properly predicted by the model, but the timing and steepness of the fluctuations is not very accurate. The performance metrics and visualizations for the D-NN model were presented to hydrologists at Waterschap Zuiderzeeland: they indicated that the forecasts are accurate enough to be used for decision support if they are generated for up to 24 h ahead.

The forecasting performance metrics of several models of the 1-RNN and XY-RNN architecture are shown in Table 4. These models were evaluated to be able to compare ANN and RNN architectures, to study the effect of using either LSTM neurons or GRU neurons, and to get an idea of the optimal RNN model size. We observe that recursive forecasting using RNNs (the 1-RNN models) yield even worse results than the recursive ANN models. A possible reason for this is that the error accumulation occurs both in the hidden states and in the input data for recursive forecasting RNNs. It also becomes clear that models with GRU neurons consistently yield better results than LSTM neurons. Lastly, we see that the results of the XY-RNN are slightly worse than the D-NN results. Moreover, the XY-RNN models are unable to yield better results than the VAR/VECM models. This is remarkable, because both the D-NN and XY-RNN models generate forecasts that do not suffer from an accumulation of error in the input data.

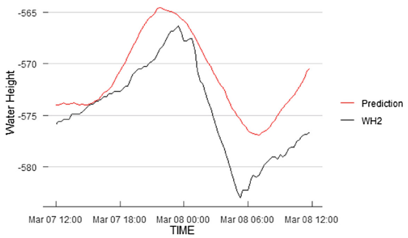
In Table 5 we show the forecasting performance of several RNN models with the proposed Encoder/Decoder RNN architectre. We conclude that the perfor-

Table 3. Forecasting performance expressed as RMSE, MAPE and R^2 for various ANN models with lag order $p = 96$. Performance was evaluated by generating forecasts on the test set. In bold: for each target variable, its highest quality prediction with respect to RMSE, MAPE or R^2.

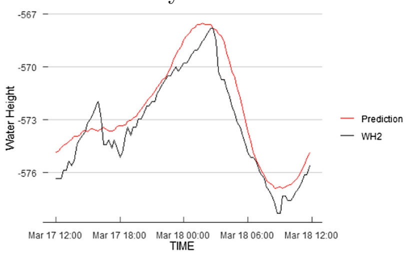
Model	WH2			WH3			WH5		
	RMSE	MAPE	R^2	RMSE	MAPE	R^2	RMSE	MAPE	R^2
1-NN_{24}(96)	7.22	.0080	.36	16.17	.0231	.49	8.77	.0096	.24
1-NN_{48}(96)	4.23	.0051	.71	6.1	.0074	.78	5.86	.0076	.58
1-$NN_{48,16}$(96)	3.28	.0049	.78	4.73	.0059	.87	4.59	.0060	.70
1-$NN_{48,48}$(96)	4.18	.0052	.64	5.41	.0071	.82	5.2	.0069	.57
D-NN_{24}(96)	2.37	.0030	**.89**	3.2	.0040	.92	2.76	.0038	.84
D-NN_{48}(96)	**2.35**	**.0029**	**.89**	3.15	**.0037**	**.92**	2.85	.0038	.84
D-NN_{96}(96)	2.68	.0036	**.89**	**3.11**	.0038	**.92**	**2.49**	**.0033**	**.86**
D-NN_{200}(96)	3.05	.0040	.84	3.45	.0045	.91	3.64	.0047	.81
D-$NN_{48,16}$(96)	2.63	.0039	.86	4.23	.0053	.91	2.76	.0039	.85
D-$NN_{48,48}$(96)	3.03	.0043	.85	3.94	.0044	**.92**	3.35	.0047	.82
D-$NN_{96,16}$(96)	2.46	.0033	.88	4.16	.0049	.9	2.57	**.0033**	.85
D-$NN_{96,48}$(96)	2.98	.0040	.82	3.46	.0045	.91	3.39	.0043	.80



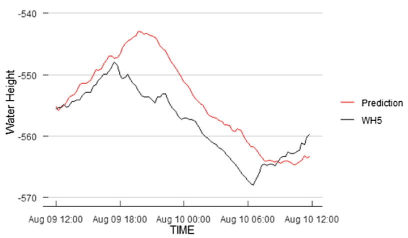
Dry weather



Moderate rainfall



Dry, after rainfall



Heavy rainfall

Fig. 3. Exemplary selection of WH2, WH3 and WH5 water levels, and the direct forecasts generated by the D-NN_{96}(96) model. Captions indicate the weather conditions on the day for which the forecast was generated.

Table 4. Forecasting performance expressed as RMSE, MAPE and R^2 of RNN models. Performance was evaluated by generating forecasts on the test set. In bold: for each target variable, its highest quality prediction with respect to either RMSE, MAPE or R^2.

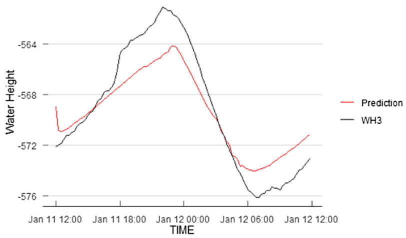
Model	WH2			WH3			WH5		
	RMSE	MAPE	R^2	RMSE	MAPE	R^2	RMSE	MAPE	R^2
1-LSTM$_{25}$	7.00	.0091	.48	2.84	.0289	.55	6.65	.0067	.48
1-GRU$_{25}$	6.47	.0081	.57	7.32	.0121	.72	5.01	.0061	.64
1-LSTM$_{50}$	12.66	.0168	.00	24.83	.0210	.35	17.21	.0209	.00
1-GRU$_{50}$	4.04	.0047	.70	9.90	.0115	.41	3.97	.0042	.69
1-GRU$_{100}$	5.83	.0070	.40	11.09	.0163	.64	7.22	.0085	.35
1-GRU$_{50,50}$	9.50	.0122	.58	7.56	.0089	.54	6.16	.0075	.46
1-GRU$_{100,50}$	5.15	.0066	.69	8.96	.0113	.67	6.20	.0079	.52
XY-GRU$_{16}$	3.40	.0047	**.81**	3.79	.0047	.89	3.56	.0044	.71
XY-GRU$_{25}$	3.35	.0046	**.81**	**3.47**	**.0044**	**.91**	**2.83**	**.0037**	**.81**
XY-GRU$_{50}$	4.07	.0056	.78	4.43	.0057	.86	3.71	.0047	.76
XY-GRU$_{25,25}$	3.83	.0054	.78	4.03	.0051	.87	3.47	.0044	.74
XY-GRU$_{50,16}$	3.68	.0053	**.81**	3.83	.0051	.90	3.21	.0042	.77
XY-GRU$_{50,50}$	4.45	.0062	.74	4.17	.0055	.87	4.04	.0052	.70
XY-GRU$_{100,50}$	**3.01**	**.0042**	**.81**	3.94	.0054	.90	3.27	.0045	.78
XY-GRU$_{100,100}$	3.73	.0053	.80	3.89	.0049	.88	3.40	.0042	.74

mance of the ED-RNN model is comparable to the best performing alternative model (the D-NN) and better than the other results we have seen so far. Table 5 also contains the performance metrics for an ensemble model, containing the models ED-GRU$_{12,12}$, ED-GRU$_{25,50}$, D-NN$_{48}$(96) and D-NN$_{96}$(96). The predictions of the ensemble model are defined as the average of the 4 predictions from the individual models at each time step. The ensemble model shows that there is still room for improvement. Additionally, Fig. 4 contains informative visualizations of the predictions generated by the ED-GRU$_{25,50}$ model. From this, we conclude that the timing and steepness of fluctuations are more accurately predicted by the ED-RNN model compared to the D-NN, except for situations with heavy rainfall.

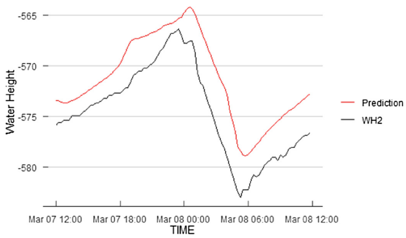
Lastly, we contrast the conditional forecasts generated by the D-NN$_{96}$(96) and ED-GRU$_{25,50}$ models in the case of an alternative path for the exogenous variables. Figure 5 contains, for a selected date in the test set, the original and alternative future paths for the variables on which we condition the forecasts. The alternative scenario has flow rates that are lower (less pumping) and with different timing (FR2000 is activated later, FR2200 is de-activated earlier). Hydrologists at Waterschap Zuiderzeeland indicated that, based on their domain experience, this alternative pumping scenario should result in higher water levels

Table 5. Forecasting performance expressed as RMSE, MAPE and R^2 of Encoder/Decoder RNNs with exogenous variables. Additionally, the performance for an ensemble model consisting of 2 ED-GRU models and 2 D-NN models. Performance was evaluated by generating forecasts on the test set. In bold: for each target variable, its highest quality prediction with respect to either RMSE, MAPE or R^2 by a single model.

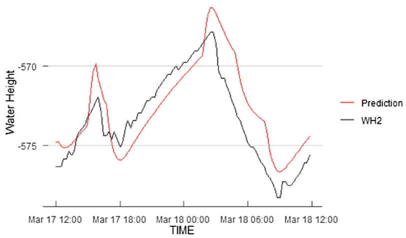
Model	WH2			WH3			WH5		
	RMSE	MAPE	R^2	RMSE	MAPE	R^2	RMSE	MAPE	R^2
ED-GRU$_{8,8}$	2.98	.0039	.84	5.39	.0055	.78	3.01	.0038	.80
ED-GRU$_{8,12}$	2.51	.0032	**.87**	3.31	.0041	.92	**2.74**	**.0036**	.83
ED-GRU$_{12,12}$	**2.37**	**.0029**	**.87**	3.24	.0040	.92	2.79	**.0036**	.83
ED-GRU$_{25,25}$	2.85	.0036	.85	3.23	.0041	.92	3	.0038	.79
ED-GRU$_{25,50}$	2.95	.0038	.86	**3.04**	**.0037**	**.93**	2.85	**.0036**	**.84**
ED-GRU$_{50,25}$	3.28	.0044	.85	3.31	.0040	.92	2.84	**.0036**	.83
ED-GRU$_{50,50}$	3.04	.0039	.84	3.49	.0043	.91	2.87	**.0036**	.82
Ensemble	2.29	.0029	.91	2.71	.0032	.94	2.25	.0029	.89



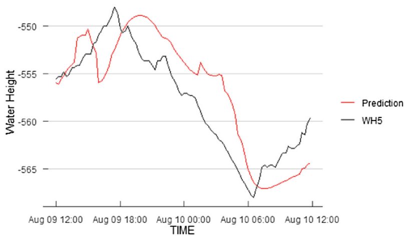
Dry weather



Moderate rainfall



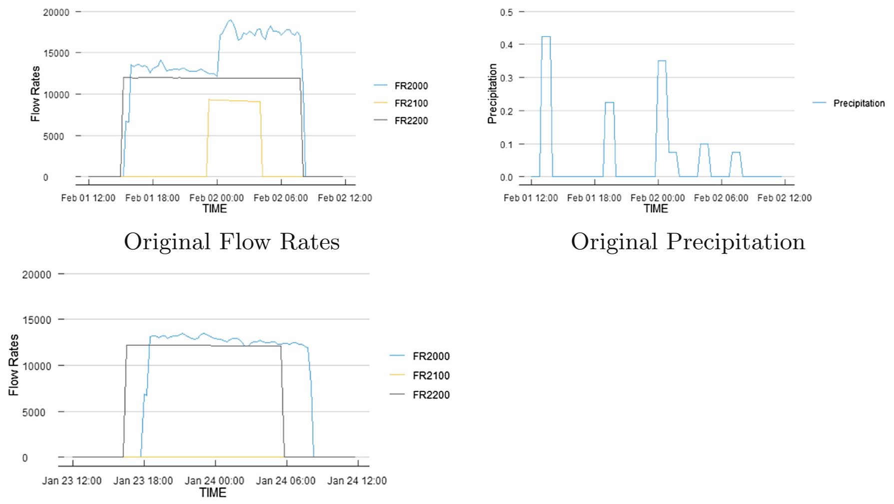
Dry, after rainfall



Heavy rainfall

Fig. 4. Exemplary selection of WH2, WH3 and WH5 water levels, and the forecasts generated by the ED-GRU$_{25,50}$ model. Captions indicate the weather conditions on the day for which the forecast was generated.

with a delayed peak. Figures 6 and 7 contain the conditional forecasts generated for both the original and alternative scenario, and the original water levels. The alternative D-NN forecasts did not contain the expected delayed peaks, and for WH3 it has mostly lower water levels, which is unrealistic. The alternative ED-RNN forecasts did contain the expected changes compared to the original scenario. According to the knowledge and experience of the hydrologists at Waterschap Zuiderzeeland, the alternative forecasts of the ED-RNN model are therefore realistic and could be used for real-life scenario evaluation.



Original Flow Rates

Original Precipitation

Alternative Flow Rates

Fig. 5. Visualization of the exogenous variables used to generate the original (old) prediction above, and below for the alternative (new) prediction for February 2, 2018.

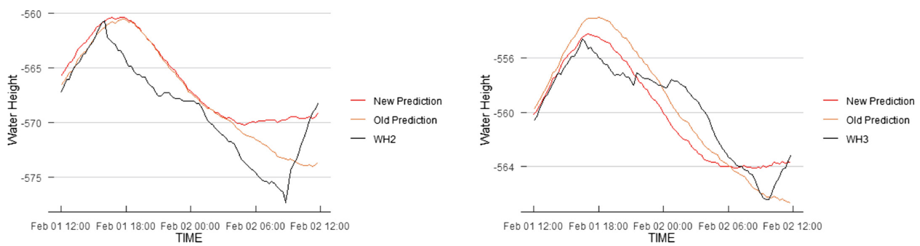


Fig. 6. Actual water height, original forecast (old) and alternative conditional forecast (new) on February 2, for target series WH2 and WH3. Predictions are generated by the D-NN$_{96}$(96) model.

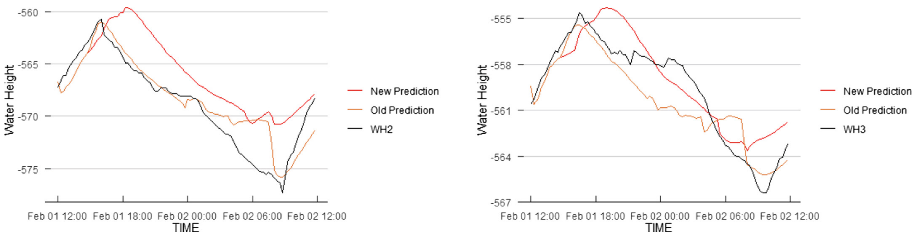


Fig. 7. Actual water height, original forecast (old) and alternative conditional forecast (new) on February 2, for target series WH2 and WH3. Predictions are generated by the ED-RNN$_{25,50}$ model.

6 Conclusion

This paper describes a practical situation in which the application of forecasting models could lead to energy efficiency and decreased risk in water level management. In consultation with the Waterschap Zuiderzeeland, we formalized the problem as a conditional forecasting problem using time series measurements of hydrological variables. Our modelling choices, e.g. the forecasting horizon and the inclusion of future paths of exogenous variables, ensure that the resulting forecasting problem imitates a real-life forecasting task.

We have proposed a novel architecture for conditional forecasting with RNNs. Our approach, described as an *Encoder/Decoder with Exogenous Variables* RNN (ED-RNN) model, is intuitive in its memory management, can be easily extended and is flexible in the inclusion of past observations and future paths of exogenous variables. It combines the ideas of sequence-to-sequence (Seq2Seq) RNN models with the conditional forecasting literature, resulting in a model architecture that can be used for modelling in all sorts of time series problems.

We showed that on this specific conditional forecasting problem, the results of the ED-RNN are comparable to the best performing alternative model (the D-NN), and better than the other alternatives considered. We observe that the predictions generated by the ED-RNN more accurately capture short-term fluctuations in the water heights than the predictions of the D-NN. Additionally, the ED-RNN generates realistic alternative conditional forecasts. Therefore, the model is applicable for real-life scenario evaluation and decision support by the Waterschap Zuiderzeeland.

Our search for model parameters, such as the model size, lag parameter, number of iterations and stopping criterion, was not exhaustive. Balancing the data and inclusion of additional measurement series were also not investigated. Hence, there is still room for improvement of our models. We conclude that our ED-RNN model architecture is a very interesting option for conditional forecasting with RNNs and that more empirical evaluation is needed to contrast the performance with alternative modelling techniques.

References

1. Alaire, J.J., Chollet, F., RStudio, Google: R Interface to Keras (2019). https://keras.rstudio.com/. Accessed 05 May 2019
2. Armstrong, J.S., Collopy, F.: Error measures for generalizing about forecasting methods: empirical comparisons. Int. J. Forecast. **8**(1), 69–80 (1992)
3. Che, Z., Purushotham, S., Cho, K., Sontag, D., Liu, Y.: Recurrent neural networks for multivariate time series with missing values. Sci. Rep. **8**(1), 6085 (2018)
4. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555 (2014)
5. FutureWater: Rapport Wateraanvoer Noordoostpolder (2006). https://www.futurewater.nl/downloads/2006_Immerzeel_FW50.pdf. Accessed 19 Oct 2018
6. Goodfellow, I., Bengio, Y., Courville, A., Bengio, Y.: Deep Learning, vol. 1. MIT Press, Cambridge (2016)
7. Groenen, I.: Representing seasonal patterns in gated recurrent neural networks for multivariate time series forecasting. Master thesis (2018). http://www.scriptiesonline.uba.uva.nl/657906. Accessed 20 Oct 2018
8. Guo, T., Lin, T., Lu, Y.: An interpretable LSTM neural network for autoregressive exogenous model. arXiv preprint arXiv:1804.05251 (2018)
9. Hamilton, D.P., Schladow, S.G.: Prediction of water quality in lakes and reservoirs. Part I-model description. Ecol. Model. **96**(1–3), 91–110 (1997)
10. Hamzaçebi, C., Akay, D., Kutay, F.: Comparison of direct and iterative artificial neural network forecast approaches in multi-periodic time series forecasting. Expert Syst. Appl. **36**(2), 3839–3844 (2009)
11. Haykin, S.S.: Neural Networks and Learning Machines, vol. 3. Pearson Education, Upper Saddle River (2009)
12. Hill, T., Marquez, L., O'Connor, M., Remus, W.: Artificial neural network models for forecasting and decision making. Int. J. Forecast. **10**(1), 5–15 (1994)
13. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
14. Johansen, S.: Estimation and hypothesis testing of cointegration vectors in Gaussian vector autoregressive models. Econometrica: J. Econ. Soc. **59**, 1551–1580 (1991)
15. Konikow, L.F., Bredehoeft, J.D.: Ground-water models cannot be validated. Adv. Water Resour. **15**(1), 75–83 (1992)
16. Lütkepohl, H.: New Introduction to Multiple Time Series Analysis. Springer, Heidelberg (2005)
17. Mishra, A., Desai, V.: Drought forecasting using feed-forward recursive neural network. Ecol. Model. **198**(1–2), 127–138 (2006)
18. Nepf, H.: Drag, turbulence, and diffusion in flow through emergent vegetation. Water Resour. Res. **35**(2), 479–489 (1999)
19. Sophocleous, M.: Interactions between groundwater and surface water: the state of the science. Hydrol. J. **10**(1), 52–67 (2002)
20. Stigler, M.: tsDyn: nonlinear time series models with regime switching (2019). https://www.rdocumentation.org/packages/tsDyn/versions/0.9-44. Accessed 12 June 2019
21. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Advances in Neural Information Processing Systems, pp. 3104–3112 (2014)

22. Tiwari, M.K., Chatterjee, C.: Development of an accurate and reliable hourly flood forecasting model using wavelet-bootstrap-ANN (WBANN) hybrid approach. J. Hydrol. **394**(3–4), 458–470 (2010)
23. Wang, W., Ding, J.: Wavelet network model and its application to the prediction of hydrology. Nat. Sci. **1**(1), 67–71 (2003)
24. Yen, P.H., Jan, C.D., Lee, Y.P., Lee, H.F.: Application of Kalman filter to short-term tide level prediction. J. Waterw. Port Coast. Ocean Eng. **122**(5), 226–231 (1996)
25. Yoon, H., Jun, S.C., Hyun, Y., Bae, G.O., Lee, K.K.: A comparative study of artificial neural networks and support vector machines for predicting groundwater levels in a coastal aquifer. J. Hydrol. **396**(1–2), 128–138 (2011)
26. Zhang, G.P.: Time series forecasting using a hybrid ARIMA and neural network model. Neurocomputing **50**, 159–175 (2003)