Representing Seasonal Patterns in Gated Recurrent Neural Networks for Multivariate Time Series Forecasting

submitted in partial fulfillment for the degree of master of science

Inske Groenen
10120459

master information studies
data science
faculty of science
university of amsterdam

2018-07-06

	Internal Supervisor	**External Supervisor**
Title, Name	Dr Thomas Mensink	Ruben Peters
Affiliation	UvA, FNWI	Ynformed
Email	T.E.J.Mensink@uva.nl	Ruben@Ynformed.nl

Universiteit van Amsterdam

Representing Seasonal Patterns in Gated Recurrent Neural Networks for Multivariate Time Series Forecasting

Inske Groenen
University of Amsterdam
inske@groenen.me

ABSTRACT

Predicting wastewater inflow at municipal wastewater treatment plants is a multivariate time series forecasting task involving both long- and short-term time and seasonal dynamics, which are predominantly related to weather conditions and human activity. Gated Recurrent Neural Networks, including the Gated Recurrent Unit (GRU), have recently been shown to be effective in capturing long- and short-term dependencies. However, how to best represent and incorporate different recurring seasonal patterns with various timescales in a GRU model is a little explored challenge. This study proposes the seasonal GRU + residual GRU (sGRU+rGRU) model that leverages the strengths of GRU, seasonality extraction, and residual learning. Experimental results show that the sGRU+rGRU outperforms other models that only use parts of the techniques employed by the sGRU+rGRU. The study further shows that a GRU model is much better suited for multivariate time series forecasting than a Multilayer Perceptron (MLP) model. It is also shown that combining seasonality extraction with residual learning significantly improves the performance of both MLP and GRU.

1 INTRODUCTION

The amount of inflow at municipal wastewater treatment plants (WWTPs) over time is characterized by clear seasonal patterns as the main sources of sewage water are household waste and rainfall [24][25][26]. The volume of water that is being processed inside the plant at any point in time naturally fluctuates as a consequence. Compared to summer months, inflow is higher during winter months due to an, on average, higher amount of rainfall. Further, following the rhythm of human activity, there generally is a greater amount of inflow during daytime, as opposed to nighttime. In the weekend, the daytime peak is later on during the day, as people become active later on during the day compared to workweek days.

Equalizing the volume that is being processed over time would increase a plant's efficiency [10]. This could be achieved by using buffer basins at the start of the treatment process, but requires continuous evaluation of whether, and how much, water needs to buffered. As such, the inflow to be expected over the following hours needs to be predicted accurately on a continuous basis. A model that is able to effectively represent and incorporate the seasonal time patterns of inflow is expected to be best suited for this task. Following this, this research focuses on how the seasonal patterns present in the inflow data, which are weather- as well as human-related, can best be represented and incorporated in a model for multivariate time series forecasting.

A time series is a sequence of measurements over time. Due to the temporal ordering of the data, consecutive measurements are related to one another. This makes time series analysis distinct from cross-sectional data analysis, which involves data that does not depend on time. Multivariate time series forecasting has been studied extensively with traditional time series models such as Vector Autoregression (VAR) [2]. Two major limitations of this model are that it assumes linear dependence over time and is not well-suited to learning both short- and long-term dependencies [2][12][29][31]. In recent years, Recurrent Neural Networks (RNNs), in particular Long-Short Term Memory (LSTM) [15] and Gated Recurrent Unit (GRU) [6], have proven effective in capturing short- and long-term dependencies of multivariate input [1][14][19][27]. Therefore, a GRU model is used in this study to forecast wastewater inflow for Dutch WWTPs Losser and Oldenzaal on an hourly basis.

The characteristic of time series containing seasonal trends is that they exhibit periodically recurring behaviour. How to best incorporate recurring seasonal patterns in a GRU model is a challenge that has not been studied extensively. It has been observed that forecasting using neural networks can be significantly improved by deseasonalizing the data prior to feeding it into the network [33]. Based on this finding, this study evaluates the effect on GRU model performance of extracting the seasonal component from the inflow data by a separate model.

The seasonal component of the inflow is allowed to bypass the network, a method that is inspired by residual learning. Residual learning encompasses learning an identity mapping instead of the direct mapping input $x \rightarrow$ output $H(x)$. As such, the herein proposed residual networks only need to learn the deviation from the seasonal component rather than the direct mapping of total predicted inflow for each of the prediction time steps.

While using a GRU model for this specific forecasting task represents a relatively unexplored approach in itself, the main contribution of this research lies in combining GRU with a seasonality extraction and residual learning approach. This approach is also used in combination with a Multilayer Perceptron (MLP) for comparison. The performance of a GRU model on multivariate time series forecasting is also compared to a MLP model. A further contribution lies in evaluating the effectiveness of a GRU in learning the seasonal patterns of the inflow data. Additionally, the impact of continuing updating a model as new data becomes available is assessed.

The paper is structured as following. First, relevant previous studies are discussed (Section 2). This is followed by a description of the evaluated models (Section 3). Then, the evaluation of the experiments is covered (Section 4). Finally, conclusions are drawn based on the findings (Section 5).

2 RELATED WORK

This section describes previous research done on forecasting wastewater inflow (2.1), Recurrent Neural Networks (2.2), seasonality extraction (2.3), and residual learning (2.4).

2.1 Forecasting Wastewater Inflow

Wei *et al.* (2013) [28] found that a MLP performed well for forecasting wastewater inflow up to 150 minutes ahead, compared to a random forest, boosted tree, and support vector machine (SVM). One of the limitations of MLPs, however, are that they approach the task as a static problem, taking a fixed length vector as input without the possibility of looking at previous inputs. As such, each input vector needs to include shift variables of previous time steps, aside from the variables of the current time step. Since gated RNNs do have the ability to look back at previous inputs through their internal memory, they are much better equipped to taking temporal dynamics into account. Thus, it is expected that a gated RNN will outperform a MLP, especially for predictions further ahead in time. For forecasting inflow just one hour ahead, it was shown that a LSTM outperformed MLP and Support Vector Regression (SVR) [30].

Specifically for WWTPs Losser and Oldenzaal, Jordens (2018) [16] compared a Boosted Regression Tree, MLP, K-Nearest Neighbour and SVR for forecasting the inflow one hour (t+1), two hours (t+2), and three hours (t+3) ahead of time. The study concluded that a MLP model performed best for WWTP Losser, whereas the results for WWTP Oldenzaal were inconclusive. A gated RNN was not included in the study, nor was there a specific focus on how the seasonality could best be extracted from the inflow data. As such, this study expands on the earlier research. It also focuses on making predictions further ahead of time (t+6, t+12, and t+24). To be able to compare the results of this study with the earlier research, a MLP will be used as one of the baseline models.

2.2 Recurrent Neural Networks

RNNs were specifically developed to process sequential data as they take into account not only the input at the current time step, but also their output of the previous time step [11]. A well-researched problem of traditional recurrent neural networks, though, is that back-propagation over many stages can cause gradients to either vanish or explode, making it difficult to learn long-term dependencies [11][15]. Gated RNNs, including the successful LSTM [15] and GRU [6], were developed to solve this problem by carefully controlling what information to keep and what information to update at each time step [11]. Apart from incorporating time dynamics, gated RNNs are able to learn different timescales [4][18]. However, training of such networks is not a trivial task [9][23]. The long-term dependencies still need to be transmitted over each time step.

Various research has been done to make learning of multiple timescales more explicit. Lui *et al.* (2015) proposed the Multi-Timescale LSTM, which divides LSTM units in different groups operating at different time periods [21]. The Fast-Slow RNN has a hierarchical structure, whereby the higher hierarchical layer updates slower than lower layers [22]. In this study, a stacked GRU architecture is used to allow the network to implicitly learn different hierarchical time scales. However, for incorporating the longer seasonal time scales (monthly, weekly and daily) this study proposes the seasonality extraction and residual learning approach.

2.3 Seasonality Extraction

Much research has been done on how to best extract seasonality from seasonal time series [3][33]. A classical approach is to decompose seasonal time series into trend, seasonal, cyclical, and residual components by a smoothing technique such as moving average. A popular model for univariate time series that involves removing the seasonality from the input data is the seasonal Autoregressive Integrated Moving Average (ARIMA) model [3]. It was shown that a combined model of ARIMA and MLP obtained better results than either model could on its own [32]. Previous research also showed that a MLP performed better when the input data was deseasonalized compared to a MLP that predicted based on the non-deseasonalized data [33].

In this study, an ARIMA model for extracting the seasonality from the inflow data was judged unsuitable. This was due to the fact that an unequal amount of data was available on the hourly seasonality in contrast to the monthly, weekly, and weekday seasonality. Therefore, one approach used in this study was to extract the seasonality based on lookup tables containing average seasonal values for months, weeks, weekdays, and hours. The other approach of this study was to employ a GRU for seasonality extraction.

2.4 Residual Learning

Residual learning was introduced as a way to train deep neural networks. By allowing some parts of the Residual Net (ResNet) to skip connections and pass their output on to some lower layer, the intermediate layers are forced to learn the residual between their input and output [13]. Thus, $H(x) = F(x) + x$, whereby $F(x)$ is the residual. This architecture with skip connections has allowed ResNet to outperform earlier models on various tasks, such as image recognition and object detection [5][13]. It was noted that, while $F(x)$ can take various forms, a skip connection needs to pass at least two layers for the improvement in performance to occur [13]. Based on this last observation, the architecture of the residual GRUs of this study comprises stacked GRU cells.

The recently proposed Long- and Short-term Time-series network (LSTNet) was shown to outperform various variations of the VAR model in time series forecasting [20]. Its architecture includes recurrent skip connections that take the same inputs as the recurrent part of the network. Through this setup, LSTNet is thought to be better equipped to pass on relatively long-term dependencies. It further involves another bypass connection that, via an Autoregressive (AR) model, processes the linear component of the data. This study also allows long-term dependencies, in this case the daily, weekly, and monthly seasonality, to bypass the network in order to improve model performance.

3 MODELS

This section describes the details of the evaluated baseline (3.1) and residual models (3.2).

3.1 Baseline Models

3.1.1 Weighted Seasonal Component (WSC). This model is used to extract the monthly, weekly and daily seasonality from the inflow data. This is done by taking a weighted sum of the seasonal features. The seasonal features are based on lookup tables that contain the mean hourly inflow at each hour of the day, the difference for each month from the mean hourly inflow over all months, the difference for each week from the mean hourly inflow over all weeks, and the difference for each weekday from the mean hourly inflow over all weekdays. The weighting factors for the seasonal features are found by zero-initializing them and using the Adam algorithm for optimization [17]. The equation for the WSC at prediction time step i is defined as following:

$$WSC_i = W_h \bar{h}_i + W_m m_i + W_w w_i + W_d d_i \tag{1}$$

whereby W_h is the weighting factor for $\bar{h}_i$, which is the mean hourly inflow at prediction step i, W_m is the weighting factor for m_i, which is the difference in hourly inflow for the concerned month from the mean hourly inflow over all months, W_w is the weighting factor for w_i, which is the difference in hourly inflow for the concerned week from the mean hourly inflow over all weeks, and W_d is the weighting factor for d_i, which is the difference in hourly inflow for the concerned weekday from the mean hourly inflow over all weekdays.

The prediction for all 6 prediction steps is then according to Equation 2:

$$\hat{Y} = WSC(x_{ms}) \tag{2}$$

whereby $\hat{Y}$ denotes a vector containing the predicted inflow values for all 6 prediction steps, $WSC(x_{ms})$ denotes a vector containing the WSC values for all 6 prediction steps, and x_{ms} denotes the seasonal features for all 6 prediction steps.

3.1.2 Multilayer Perceptron (MLP). The MLP model consists of four hidden layers of 512, 256, 128, and 75 units. The number of hidden layers and units for each layer, except the last one, are found via grid search. The number of units of the last hidden layer is set to the same number of units as the last GRU cell in the GRU model. All hidden layers have Exponential Linear Unit (ELU) as activation function, which is defined as following [8]:

$$f(x) = \begin{cases} x, & \text{if } x > 0 \\ \exp(x) - 1, & \text{if } x \leq 0 \end{cases} \tag{3}$$

The fully-connected output layer consists of 6 units with no activation function for regression output (predicted inflow at t+1, t+2, t+3, t+6, t+12, and t+24). Thus, the prediction is according to Equation 4:

$$\hat{Y} = H(x_t) \tag{4}$$

whereby x_t denotes the input vector at time step t, containing both seasonal and residual features.

3.1.3 Gated Recurrent Unit (GRU). For the gated RNN model, a GRU cell was used as it has been shown that GRU yields similar results compared to LSTM, while converging faster [7]. The model consists of two stacked GRU cells, each of 75 units. The number of stacked cells, units for each cell, and time steps to process before prediction are found via grid search. At each time step t, the first GRU cell takes as input the explanatory variables at time step t and the previous hidden state h_{t-1} to produce the hidden state h_t. Determining h_t is done according to Equations 5-8:

$$r_t = \sigma(W_{xr} x_t + W_{hr} h_{t-1} + b_r) \tag{5}$$

$$u_t = \sigma(W_{xu} x_t + W_{hu} h_{t-1} + b_u) \tag{6}$$

$$c_t = \tanh(W_{xc} x_t + r_t \odot (W_{hc} h_{t-1}) + b_c) \tag{7}$$

$$h_t = (1 - u_t) \odot h_{t-1} + u_t \odot c_t \tag{8}$$

whereby r_t denotes the reset gate, W are weight matrices, b are bias terms, σ denotes the logistic sigmoid activation function, u_t denotes the update gate, tanh denotes the tangent hyperbolic activation function, c_t denotes the candidate hidden state, and $\odot$ denotes element-wise multiplication.

The second GRU cell takes the hidden state of the first GRU cell and its own previous hidden state as input. Producing hidden state h_t is then done according to Equations 5-8.

The model is enabled to process 72 time steps before making a prediction. After the 72 time steps, the last hidden state of the second GRU cell is put through a fully-connected output layer of 6 units with no activation function to produce the predicted inflow for all 6 prediction steps. As such, the prediction is according to Equation 4.

3.1.4 Seasonal GRU (sGRU). This model uses a GRU to extract the seasonal component from the inflow data. The model consists of a GRU cell of 18 units. The number of layers and units of the GRU cell are found via grid search. Rather than taking average seasonal values, this model takes the numerical values corresponding to the relevant month, week, weekday, and hour as input. Thus, at time step t, the sGRU takes as input the corresponding hour, weekday, week, and month for all 6 prediction steps at time step t and the previous hidden state h_{t-1} to produce h_t following equations 5-8. Equal to the GRU model, this model is enabled to process 72 time steps before producing the prediction. After the 72 time steps, the last hidden state is put through a fully-connected output layer of 6 units with no activation function to output the seasonal component for all 6 prediction steps. Thus, the prediction for all 6 time steps is according to the following equation:

$$\hat{Y} = S(x_s) \tag{9}$$

whereby $S(x_s)$ denotes a vector consisting of the seasonal components of the 6 prediction steps, and x_s denotes an input vector consisting of the numerical seasonal features for all 6 prediction steps.

3.2 Residual Learning

3.2.1 Weighted Seasonal Component + residual MLP (WSC+rMLP). This model is a combination of the WSC and MLP model, whereby the architecture of the rMLP is equivalent to that of the MLP model. The difference being that, rather than using a MLP to directly map $x_t \rightarrow \hat{Y}$, a MLP is now used to map $x_r \rightarrow F(x_r)$, whereby $F(x_r)$ denotes the deviation from the WSC for all prediction steps and x_r the input vector consisting of residual features concerning time step t. Combining this with the WSC for each corresponding prediction

step, calculated according to Equation 1, results in the predicted inflow for all 6 prediction steps, following Equation 10:

$$\hat{Y} = WSC(x_{ms}) + F(x_r). \tag{10}$$

3.2.2 Weighted Seasonal Component + residual GRU (WSC+rGRU). This model is similar to WSC+rMLP, but, instead of using a MLP, a GRU is used to determine the deviation from the WSC for all 6 prediction steps. The architecture of the rGRU is the same as that of the GRU model. Combining the WSC and the output of the rGRU results in the predicted inflow for all 6 steps, according to Equation 10.

3.2.3 Seasonal GRU + residual GRU (sGRU+rGRU). This model combines the sGRU model and the rGRU from the WSC+rGRU model. The sGRU of this model has the same architectural setup as the sGRU model, while the setup of the rGRU is equal to the rGRU in the WSC+rGRU model. Combining the output of the sGRU and rGRU results in the predicted inflow for all 6 steps, following Equation 11:

$$\hat{Y} = S(x_s) + F(x_r). \tag{11}$$

4 EVALUATION

This section covers the details on the data used in the study (4.1), the implementation details of training the models (4.2), the model performance metrics used for evaluation (4.3), the seasonality present in the data (4.4), and the results obtained in the different experiments (4.5).

4.1 Data

The following three datasets were the basis for this research:

(1) Sensor data from WWTPs Losser and Oldenzaal provided by Dutch Water Authority Vechtstromen;
   - Daily wastewater inflow data covering the period 1 January 2010 up to 25 February 2018;
   - Half-hourly inflow data covering the period 03:00 02-10-2016 up to 08:00 17-04-2018;

(2) Weather data from weather station Twente made publically available by the Royal Netherlands Meteorological Institute (KNMI);
   - Hourly data covering the period 01:00 1 January 2010 up to 08:00 17-04-2018;

(3) Drink water usage in the area of Oldenzaal-Losser and Oldenzaal-De Lutte provided by water company Vitens;
   - Hourly data covering the period 01:00 1 January 2010 up to 08:00 17-04-2018.

From these datasets the following variables were extracted; wastewater inflow, drink water usage, temperature, sunshine duration, precipitation duration, precipitation sum, and whether it had snowed or not. Jordens (2018) [16] was the underlying basis for extracting these variables. The variables of which hour it was and whether it was a holiday, or not, were further included. Based on the daily wastewater inflow data over 2010-2018, three additional features were established for each target variable (t+1, t+2, t+3, t+6, t+12, t+24): the difference for the concerned month from the mean hourly inflow over all months (m), the difference for the concerned week from the mean hourly inflow over all weeks (w), and the difference for the concerned weekday from the mean hourly inflow over all weekdays (d). In addition, the mean hourly inflow for the concerned hour of the day was included for each target variable. This resulted in 35 explanatory variables (of which 11 concerned the current hour, and 24 were seasonal) for the data used in the WSC+GRU model.

In the case of the sGRU+rGRU model, the seasonal features were not based on lookup tables. Numerical values corresponding to the concerned hour, weekday, week, and month were used instead. Thus, there were also 35 explanatory variables (11 concerning the current hour, and 24 seasonal) for this model.

As for the GRU model, m and w were only included for t+1, and d was only included for t+1 and t+24. This was done to not have the number of input variables for the GRU model triple, while, in most cases, these variables are not any different from each other for the 6 prediction time steps of one time step vector. In addition, the GRU model is able to look back at 72 previous inputs via its internal memory. Thus, it indirectly does have access to m, w, and d of the other prediction steps. As such, there were 21 explanatory variables (11 concerning the current hour, and 10 seasonal variables) for the GRU model.

The data used in the MLP models included shift variables (t-1, t-2, t-6, t-12, t-18, t-21, t-22 and t-23) based on the variables that concerned the current hour, but excluding the current hour variable, and whether it was a holiday or not. Care was taken to, for each target variable, include shift variables regarding the situation on the day before on that exact hour. Thus, the data for the MLP and WSC+rMLP consisted of 99 explanatory variables (of which 11 concerned the current hour, 64 concerned previous hours, and 24 were seasonal).

4.2 Implementation Details

The train dataset covered the period 03:00 02-10-2016 up until 01:00 30-09-2017, which consisted of 8711 observations. Data over the period 01:00 30-09-2017 up to 23:00 31-01-2018, consisting of 2975 observations, was used as validation set during training. All models were evaluated based on a test dataset covering the period 00:00 01-02-2018 up to 08:00 17-04-2018, which consisted of 1808 observations.

For training MLP and WSC+rMLP, dropout on the second hidden layer of the MLP was set to 0.7 and dropout on the last hidden layer was set to 0.3. For training GRU, WSC+rGRU and sGRU+GRU, dropout was set to 0.3 on each GRU cell in the network. For training the WSC+rGRU model, the previously trained WSC was loaded in and frozen at first. When no further improvement was obtained in training with this setup, the WSC was allowed to be trained together with the rGRU. The same was true for the sGRU+rGRU model, whereby now the sGRU part was frozen at first.

Grid search $\{50, 75, 100\}$ was performed to find the optimal amount of units for the GRU cells in the GRU and WSC+rGRU models. For the sGRU+GRU, a grid search $\{1, 2\}$ and $\{18, 30, 50\}$ was done to find, respectively, the optimal amount of layers, and the number of units for the GRU cell of the sGRU. For the MLP model, a grid search $\{265, 512, 1024\}$ was performed to find the optimal amount of units for the first hidden layer. Each consecutive

hidden layer, except the last hidden layer, was set to be half that of the preceding hidden layer.

For all models, the Adam algorithm was used for optimization [17]. Initially, the learning rate was set to 0.01. It was decreased by a factor 10 when, after 40 epochs, no increase in performance was observed. Finally, models were deemed trained when no increase in performance was observed with further decreasing the learning rate, or in case the learning rate was dropping below the set threshold value of 0.00001.

4.3 Model Performance Metrics

Three metrics were used to evaluate the performance of the models; root mean squared error (RMSE), mean absolute error (MAE), and mean absolute percentage error (MAPE). These metrics are computed as following:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{n}} \tag{12}$$

$$MAE = \frac{\sum_{i=1}^{n}|\hat{y}_i - y_i|}{n} \tag{13}$$

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{\hat{y}_i - y_i}{y_i}\right| \tag{14}$$

whereby $\hat{y}_i$ is the predicted value, y_i is the actual value, and n is the total number of observations.

For all three metrics a lower value means better performance. Dutch Water Authority Vechtstromen has indicated that a model with an average MAPE value below 10% is usable in practice.

4.4 Seasonality of the Data

First, the seasonality present in the data of WWTPs Oldenzaal and Losser was visually explored. Figure 1 shows the monthly seasonality. When analyzed in respect to the average hourly rainfall duration per month (see Figure 2) and average hourly amount of rainfall per month (see Figure 3), it is clear that there is a strong correlation. The peaks in inflow in January and February are explained by peaks in the average hourly rainfall duration. The increase in inflow in August is explained by the combination of an increase in average hourly rainfall duration and an increase in average hourly amount of rainfall, compared to the month before. The same is true for November and December, when compared to October.

Figure 4 shows the weekly seasonality. This follows a similar pattern as the monthly seasonality, albeit more erratic. The increase in inflow in weeks 5, 17, 19, 34, and 45 are explained by an increase in both the average rainfall duration (see Figure 5) and amount of rainfall for these weeks (see Figure 6), compared to all respective previous weeks. The higher amount of inflow over week 22, compared to week 21, seems to be mainly due to an increase in the average amount of rainfall in this week.

Figure 7 shows the hourly seasonality for workweek and weekend days. When compared to the average drink water usage per hour (see Figure 8), the shift in the daytime peak of inflow on weekend days, compared to working days, is clearly correlated to the shift in the daytime peak in drink water usage. Also, the higher amount of inflow during evening hours on workweek days can be explained by the higher usage of drink water during those hours on those days.

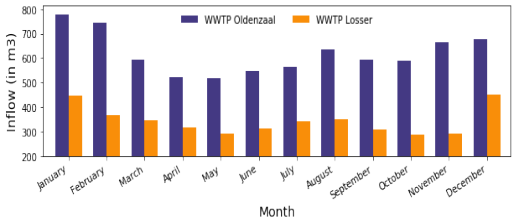

Figure 1: Average hourly inflow per month at WWTPs Oldenzaal and Losser (based on daily data over 1 January 2010 - 25 February 2018).

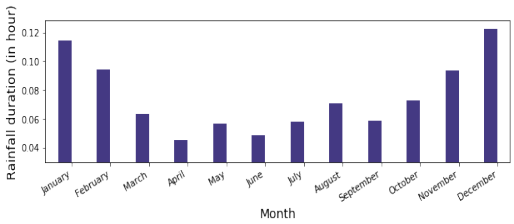

Figure 2: Average hourly rainfall duration per month in the region of Oldenzaal and Losser (based on hourly data over 1 January 2010 - 25 February 2018).

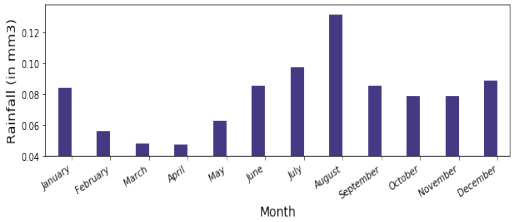

Figure 3: Average hourly amount of rainfall per month in the region of Oldenzaal and Losser (based on hourly data over 1 January 2010 - 25 February 2018).

These observations show the strong correlation between the amount of inflow at municipal WWTPs, rainfall and human usage of drink water. They empirically support previously stated assumptions on the seasonal time patterns of inflow data, and the strong seasonal component present in this time series data. These observations, thus, justify the rationale behind including monthly, weekly, weekday, and hourly seasonal features to extract the seasonal component from the total inflow.

4.5 Results

4.5.1 Experiment 1: Evaluating MLP model performance. In the first experiment the results obtained with the MLP model are compared to the results obtained in Jordens (2018) [16] with a MLP model . This is done to evaluate whether results obtained in this study can be compared to the results of the previous study. The results in Tables 1 and 2 show that for t+1 comparable results are obtained. MLP [16] seems to perform slightly better for t+1 in

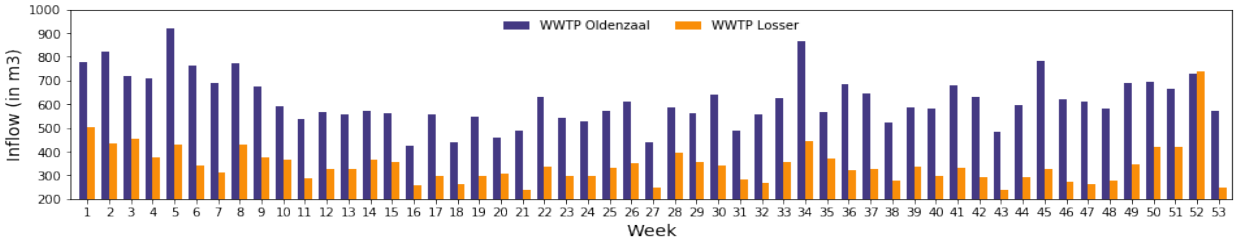

Figure 4: Average hourly inflow per week at WWTPs Oldenzaal and Losser (based on daily data over 1 January 2010 - 25 February 2018).

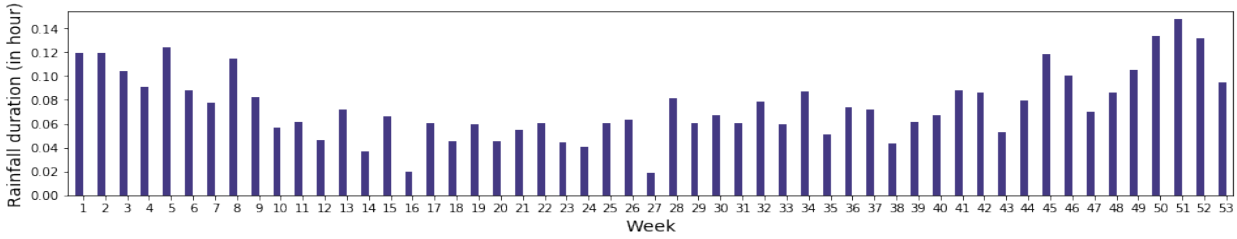

Figure 5: Average hourly rainfall duration per week in the region of Oldenzaal and Losser (based on hourly data over 1 January 2010 - 25 February 2018).

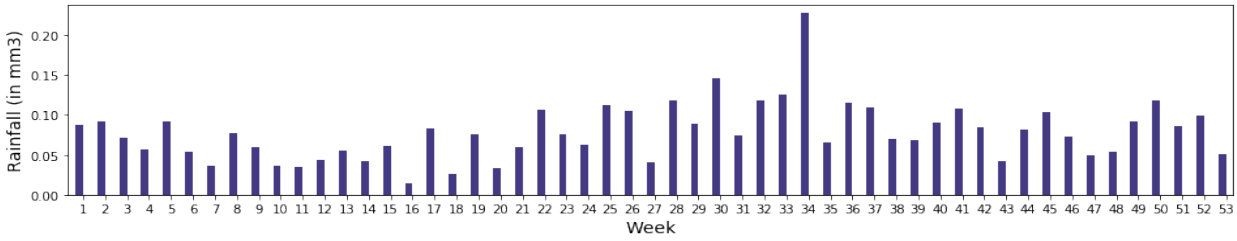

Figure 6: Average hourly amount of rainfall per week in the region of Oldenzaal and Losser (based on hourly data over 1 January 2010 - 25 February 2018).

the case of WWTP Losser. This small discrepancy could be due to the fact that the MLP model in this study is trained to output 6 prediction steps simultaneously. It could be that optimizing a specific MLP for each prediction step will result in a slight increase in performance. However, due to a limited computation budget, this approach was not feasible for this study.

Surprisingly, significant improvements were obtained in this study in predicting t+2 and t+3 with MLP for both WWTPs. The explanation appears to be a limitation of the previous study, which is that shift variables were included for only one of the explanatory variables [16].

It can be concluded that the results obtained in this study can be compared to the results of the earlier study. In line with Wei *et al.* (2013) [28], the results further show that a MLP is not well-suited to predict wastewater inflow for more than a few hours ahead.

4.5.2 Experiment 2: Evaluating a prediction model purely based on seasonality. The results in Tables 1 and 2 clearly show that a prediction model purely based on seasonality produces poor results. Both WSC and sGRU perform poorly compared to the other models. The fact that the WSC outperforms the sGRU could be due to the fact that only relatively little data was available for training the sGRU. There are, for instance, months for which the sGRU only gets to see the situation of one year. Thus, the monthly seasonality is, for some months, purely based on one year. The same is true for certain weeks. As such, it could very well be that this model, therefore, does not generalize well to the test data, which concerned the following year.

Interestingly, the performance of the WSC and sGRU models on predicting t+12 and t+24 is only moderately worse than the performance by the MLP on those time steps (see Tables 1 and 2). In the case of predicting t+12 for WWTP Oldenzaal, the WSC model even performs better than the MLP (see Tables 1 and 2). Thus, it seems the MLP model is not able to learn much more from the data for these prediction steps than the average seasonal pattern.

4.5.3 Experiment 3: Evaluating the performance of a GRU model to a MLP model. The results of the GRU model compared to those of the MLP model, shown in Tables 1 and 2, evidently show that a GRU is much better suited for multivariate time series forecasting. It shows that the MLP model was not able to learn the time related pattern of the data well. The results of the GRU model support

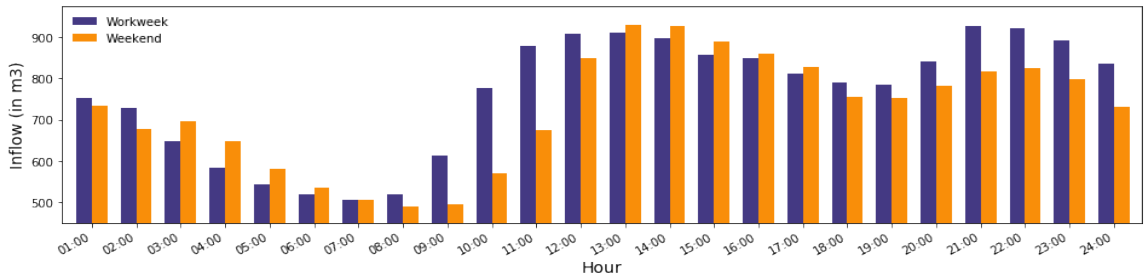

Figure 7: Average inflow per hour at WWTP Oldenzaal (based on hourly inflow over 2 October 2016 - 17 April 2018).

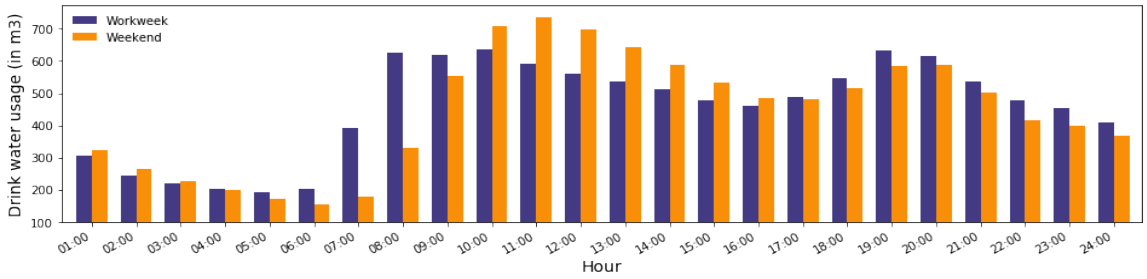

Figure 8: Average drink water usage per hour in the region Oldenzaal-Losser and Oldenzaal-De Lutte (based on hourly drink water usage over 2 October 2016 - 17 April 2018).

Table 1: Results for all models on predicting the inflow at WWTP Oldenzaal for the 6 prediction time steps. The best result is shown in bold. Model with * indicates it was trained with additional training data for evaluating continuous learning.

		MLP [16]	WSC	MLP	WSC+rMLP	GRU	WSC+rGRU	sGRU	sGRU+rGRU	sGRU+rGRU*
t+1	RMSE	232.58	451.61	229.98	203.64	120.24	93.80	477.12	**93.70**	93.71
	MAE	105.29	206.42	130.60	108.07	66.27	36.69	237.09	**34.38**	**34.38**
	MAPE	15.08%	25.94%	17.73%	14.34%	9.34%	4.36%	32.06%	**4.11%**	**4.11%**
t+2	RMSE	411.94	451.61	257.87	248.84	124.91	106.72	476.97	**105.33**	**105.33**
	MAE	163.66	206.42	134.86	122.53	62.60	48.16	236.97	**46.73**	**45.72**
	MAPE	20.90%	25.94%	17.23%	15.59%	8.11%	5.91%	32.09%	**5.65%**	**5.64%**
t+3	RMSE	431.20	451.61	296.48	287.30	142.96	112.36	477.07	**106.45**	**106.44**
	MAE	185.99	206.43	155.53	136.02	60.51	45.88	236.87	**39.33**	**39.32**
	MAPE	24.13%	25.95%	20.02%	16.89%	7.38%	5.70%	32.04%	**4.52%**	**4.52%**
t+6	RMSE	-	451.61	391.66	377.03	123.41	85.08	477.07	**85.96**	**85.96**
	MAE	-	254.74	206.40	159.79	63.32	28.40	236.90	**22.94**	**22.94**
	MAPE	-	25.94%	26.88%	19.38%	8.51%	3.17%	32.05%	**2.36%**	**2.36%**
t+12	RMSE	-	452.80	446.63	421.84	83.40	66.10	476.87	**58.35**	**58.35**
	MAE	-	207.32	230.97	178.46	44.53	25.23	236.56	**19.15**	**19.12**
	MAPE	-	25.97%	30.56%	21.50%	6.68%	2.93%	32.03%	**2.55%**	**2.54%**
t+24	RMSE	-	452.22	426.73	411.35	77.78	48.74	474.67	**47.57**	**47.57**
	MAE	-	206.55	203.10	176.38	46.42	22.52	234.39	**16.08**	**16.02**
	MAPE	-	26.12%	27.50%	21.61%	7.24%	2.91%	32.17%	**2.03%**	**2.03%**

previous research that, even for small time steps ahead forecasting, a gated RNN significantly outperforms a MLP [30]. The results further show that the GRU model was able to learn both the short- and long-term dependencies, as even for t+6, t+12, and t+24 accurate predictions are made (see Tables 1 and 2).

Surprisingly, it even outperforms for t+6, t+12, and t+24 for WWTP Oldenzaal compared to t+1, t+2, and t+3 (see Tables 1 and

Table 2: Results for all models on predicting the inflow at WWTP Losser for the 6 prediction time steps. The best result is shown in bold. Model with * indicates it was trained with additional training data for evaluating continuous learning.

		MLP [16]	WSC	MLP	WSC+rMLP	GRU	WSC+rGRU	sGRU	sGRU+rGRU	sGRU+rGRU*
t+1	RMSE	119.10	235.62	119.88	107.95	44.76	24.38	243.98	**14.55**	**12.84**
	MAE	54.10	111.15	59.30	54.05	33.41	14.43	111.01	**6.31**	**5.70**
	MAPE	15.08%	36.32%	21.19%	19.48%	14.50%	5.59	38.09%	**2.23%**	**2.12%**
t+2	RMSE	167.19	235.62	127.20	124.84	47.16	28.42	243.99	**14.36**	**13.80**
	MAE	78.53	111.14	61.54	60.16	20.94	15.17	111.03	**5.54**	6.18
	MAPE	20.90%	36.33%	21.28%	20.49%	7.41%	5.86	38.14%	**2.07%**	2.30%
t+3	RMSE	191.29	235.61	138.21	140.40	40.96	26.02	244.10	**16.30**	**15.38**
	MAE	91.23	111.12	68.59	65.09	28.67	14.95	110.44	**5.65**	6.52
	MAPE	24.13%	36.33%	24.51%	21.53%	11.53%	5.75%	37.75%	**1.79%**	2.18%
t+6	RMSE	-	235.60	184.51	192.39	37.66	28.16	244.28	**21.21**	**19.37**
	MAE	-	111.08	91.60	84.43	32.00	15.84	109.36	**11.50**	**10.14**
	MAPE	-	36.32%	33.62%	26.40%	14.33%	6.18%	37.03%	**4.50%**	**3.78%**
t+12	RMSE	-	235.55	206.38	204.50	24.66	23.59	244.10	**13.80**	**12.19**
	MAE	-	110.96	104.71	92.32	16.62	14.29	109.35	**5.84**	6.24
	MAPE	-	36.33%	39.98%	30.39%	7.44%	5.43%	37.18%	**2.02%**	2.42%
t+24	RMSE	-	229.50	199.94	197.79	39.15	21.81	236.61	**11.06**	11.22
	MAE	-	107.74	93.91	89.49	28.94	13.17	105.78	**5.62**	6.43
	MAPE	-	36.21%	35.28%	30.41%	11.86%	5.18%	37.18%	**2.29%**	2.59

2). This seems an indication that the model was not able to get passed a certain local optimum. An observation during training was that the GRU model was highly prone to get stuck in bad local optima. Care needed to be taken to not have the performance for predicting t+1, t+2, and t+3 get to far ahead of the other 3 prediction steps during training, as the model always got stuck in a bad local optimum in such a case. Thus, if a specific GRU model were to be trained for each prediction step, it is expected that the performance for all time steps will be more equal to each other. Further research is needed to determine whether this is indeed the explanation for this surprising result.

With MAPE scores of below 10% for all predictions steps, the GRU model is usable in practice.

4.5.4 Experiment 4: Evaluating seasonality extraction based on lookup tables and residual learning using a MLP. Results in Tables 1 and 2 show that seasonality extraction in combination with residual learning increases the performance of a MLP model for the forecasting task. These results show that a combination of seasonality extraction and residual learning helps optimization of the MLP for this task. While the performance of the WSC+rMLP is improved compared to the MLP model, it is still not usable in practice as the MAPE value is above the 10% threshold for all prediction steps.

4.5.5 Experiment 5: Evaluating seasonality extraction based on lookup tables and residual learning using a GRU. The results in Tables 1 and 2 show that the WSC+rGRU is able to significantly outperform the GRU model. The relatively large increase in performance (see Tables 1 and 2) seems to be explainable by the seasonal extraction and residual learning approach strengthening each other. It shows that this approach aids the optimization of the GRU. It was observed during training that the WSC+rGRU was much less prone to get stuck in bad local optima, compared to the GRU model.

4.5.6 Experiment 6: Evaluating using a GRU for seasonality extraction in combination with residual learning using another GRU. From the results in Tables 1 and 2 it is clear that the sGRU+rGRU outperforms all other models. Thus, it is shown that using a GRU to extract the seasonality in combination with a GRU for residual learning most effectively leverages the strengths of using GRU, seasonality extraction, and residual learning for this forecasting task. Comparing these results with the results obtained by the sGRU alone and the WSC+rGRU (see Tables 1 and 2), it seems the sGRU+rGRU is better able to extract the seasonality from the data. In case the sGRU was fully optimized, the performance of the sGRU+rGRU would not surpass that of the WSC+rGRU model. However, by allowing the sGRU of the sGRU+rGRU model to train further, the sGRU+rGRU was able to optimize beyond that of the WSC+rGRU model. The MAPE score is far below 10% for all prediction steps, and, thus, the model is usable in practice.

4.5.7 Experiment 7: Continuing learning. As a final step, the best performing model was used to evaluate the effect of allowing a model to continue learning as new data becomes available. To this end, the validation set covering the period 01:00 30-09-2017 up to 23:00 31-01-2018 was used as additional training data to update the best performing model. The dataset covering the period 03:00 02-10-2016 up until 01:00 30-09-2017 that was previously used for training now functioned as validation set during training.

As the best performing model was the sGRU+rGRU model (see results in Tables 1 and 2), this model was used for evaluating continuous learning. In the case of WWTP Oldenzaal, the performance

appears to have marginally improved. Overall, the performance for WWTP Losser also appears to have improved slightly (see Tables 1 and 2). Although, for some time steps the performance went down slightly (see Tables 1 and 2).

These results are inconclusive on the effect of continuous learning. The changes in performance are too small to conclusively state that they are significant. One explanation for this result could be that the additional training dataset was relatively small. A larger set may be needed to obtain further significant improvements in performance.

5 CONCLUSION

In this study, a model is proposed (sGRU+rGRU) that combines the strengths of using GRU, seasonality extraction, and residual learning for multivariate time series forecasting. It shows that it is this combination of approaches that results in the best performance. Using a GRU model with seasonal and residual features as input (GRU), or a model with seasonality extraction that only uses a GRU for residual learning (WSC+GRU) are not able to reach the performance of the sGRU+rGRU. It also shows that a MLP model is not well-equipped for the task. Seasonality extraction in combination with residual learning using a MLP (WSC+rMLP) does lead to increased performance compared to a MLP model.

No conclusive result is obtained on the effect of updating a model as new data becomes available. Further research would need to be done to reach a more conclusive answer on the influence of continuous learning on model performance. Another interesting direction for further research would be to investigate the performance effect of using online learning.

One limitation of this study is that each model was trained to predict all 6 prediction steps simultaneously. Further improvements in performance may be obtained when different models are optimized separately for each time step. This could also clear up the surprising result that some models were able to obtain better performance for later prediction steps. Another limitation is the relatively small dataset that was used for this study. The effectiveness of using a GRU to extract the seasonality from the inflow data may be larger if the GRU is able to learn from data spanning a longer time period, involving different consecutive years.

In addition, the strong correlation between wastewater inflow at municipal WWTPs, rainfall and human drink water usage is made evident in this study. It shows that wastewater inflow can be accurately forecast based on recent weather conditions, drink water usage, and preceding volumes of inflow. Taking into account the time dynamics and seasonality patterns of these sources is thereby of great importance.

ACKNOWLEDGMENTS

I would like to thank Ynformed to have entrusted me with this project. It was a pleasure to work in such a friendly, welcoming environment. I also would like to thank Dutch Water Authority Vechtstromen for providing whatever data I needed, and always being reachable for any questions I had. I'm also grateful to water company Vitens for providing me with their data. Last but definitely not least, I could not have done this project without the active contribution and enthusiasm of Thomas Mensink. He was there to help me out whenever I got stuck or was in need of new inspiration; thank you.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv e-prints* abs/1409.0473 (Sept. 2014). https://arxiv.org/abs/1409.0473

[2] G.E.P. Box, G.M. Jenkins, G.C. Reinsel, and G.M. Ljung. 2015. *Time Series Analysis: Forecasting and Control.* Wiley.

[3] George.E.P. Box and Gwilym M. Jenkins. 1976. *Time Series Analysis: Forecasting and Control.* Holden-Day.

[4] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. 2016. Recurrent Neural Networks for multivariate time series with missing values. (2016). https://arxiv.org/abs/1606.01865

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. 2018. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Trans. Pattern Anal. Mach. Intell.* 40, 4 (2018), 834–848.

[6] Kyunghyun Cho, Bart van Merrienboer, fjaglar Glehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation.. In *EMNLP*, Alessandro Moschitti, Bo Pang, and Walter Daelemans (Eds.). ACL, 1724–1734.

[7] Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning, December 2014.*

[8] Djork-Arn Clevert, Thomas Unterthiner, and Sepp Hochreiter. 2015. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). *CoRR* abs/1511.07289 (2015).

[9] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATSfi10). Society for Artificial Intelligence and Statistics.*

[10] Ramesh K Goel, Joseph R V Flora, and J Paul Chen. 2005. Flow Equalization and Neutralization. In *Physicochemical Treatment Processes*, Y.-T. Hung L. K. Wang and N. K. Shammas (Eds.). The Humana Press Inc., Totowa, NJ, 21–45. http://www.springer.com/gp/book/9781588291653

[11] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep learning.* Massachusetts Institute of Technology, Cambridge, MA.

[12] James Douglas Hamilton. 1994. *Time series analysis.* Princeton Univ. Press, Princeton, NJ.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *CVPR*. IEEE Computer Society, 770–778.

[14] Geoffrey Hinton, Li Deng, Dong Yu, George Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara Sainath, and Brian Kingsbury. 2012. Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. *IEEE Signal Processing Magazine* 29, 6 (Nov 2012), 82–97.

[15] Sepp Hochreiter and Jrgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9 (1997), 1735–1780.

[16] Carina Ryanne Jordens. 2018. *Prediction of wastewater flow rate.* Technical Report. Tilburg University.

[17] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014).

[18] Jan Koutník, Klaus Greff, Faustino Gomez, and Jrgen Schmidhuber. 2014. A Clockwork RNN. In *Proceedings of the 31 st International Conference on Machine Learning*. Beijing, China.

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 1097–1105. http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[20] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. 2017. Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks. *CoRR* abs/1703.07015 (2017).

[21] Pengfei Liu, Xipeng Qiu, Xinchi Chen, Shiyu Wu, and Xuanjing Huang. 2015. Multi-Timescale Long Short-Term Memory Neural Network for Modelling Sentences and Documents. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015.* 2326–2335. http://aclweb.org/anthology/D/D15/D15-1280.pdf

[22] Asier Mujika, Florian Meier, and Angelika Steger. 2017. Fast-Slow Recurrent Neural Networks. In *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.). Curran Associates, Inc., 5915–5924. http://papers.nips.cc/paper/7173-fast-slow-recurrent-neural-networks.pdf

[23] Razvan Pascanu, Çalar Gülçehre, Kyunghyun Cho, and Yoshua Bengio. 2014. How to Construct Deep Recurrent Neural Networks. http://arxiv.org/abs/1312.6026
[24] STOWA. 2003. *Rioolvreemd water: onderzoek naar hoeveelheden en oorsprong afvalwater.* Technical Report.
[25] STOWA. 2005. *DWAAS. Vervolgonderzoek rioolvreemd water.* Technical Report.
[26] STOWA. 2009. *HAAS- hemelwaterafvoer analyse systematiek.* Technical Report.
[27] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to Sequence Learning with Neural Networks. In *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 3104–3112.
[28] Xiupeng Wei, Andrew Kusiak, and Hosseini Rahil Sadat. 2013. Prediction of Influent Flow Rate: Data-Mining Approach. *Journal of Energy Engineering* 139 (2013), 118–123. DOI: http://dx.doi.org/10.1061/(ASCE)EY.1943-7897
[29] Hsiang-Fu Yu, Nikhil Rao, and Inderjit S. Dhillon. 2016. Temporal Regularized Matrix Factorization for High-dimensional Time Series Prediction. In *Neural Information Processing Systems (NIPS)*.
[30] Duo Zhang, Erlend Skullestad Hølland, Geir Lindholm, and Harsha Ratnaweera. 2017. Hydraulic modeling and deep learning based flow forecasting for optimizing inter catchment wastewater transfer. *Journal of Hydrology* (2017), 1–11. DOI: http://dx.doi.org/10.1016/j.jhydrol.2017.11.029
[31] Guoqiang Peter Zhang. 2003. Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing* 50 (2003), 159–175.
[32] Guoqiang Peter Zhang. 2003. Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing* 50 (2003), 159–175.
[33] G. Peter Zhang and Min Qi. 2005. Neural network forecasting for seasonal and trend time series. *European Journal of Operational Research* 160, 2 (2005), 501–514. https://EconPapers.repec.org/RePEc:eee:ejores:v:160:y:2005:i:2:p:501-514