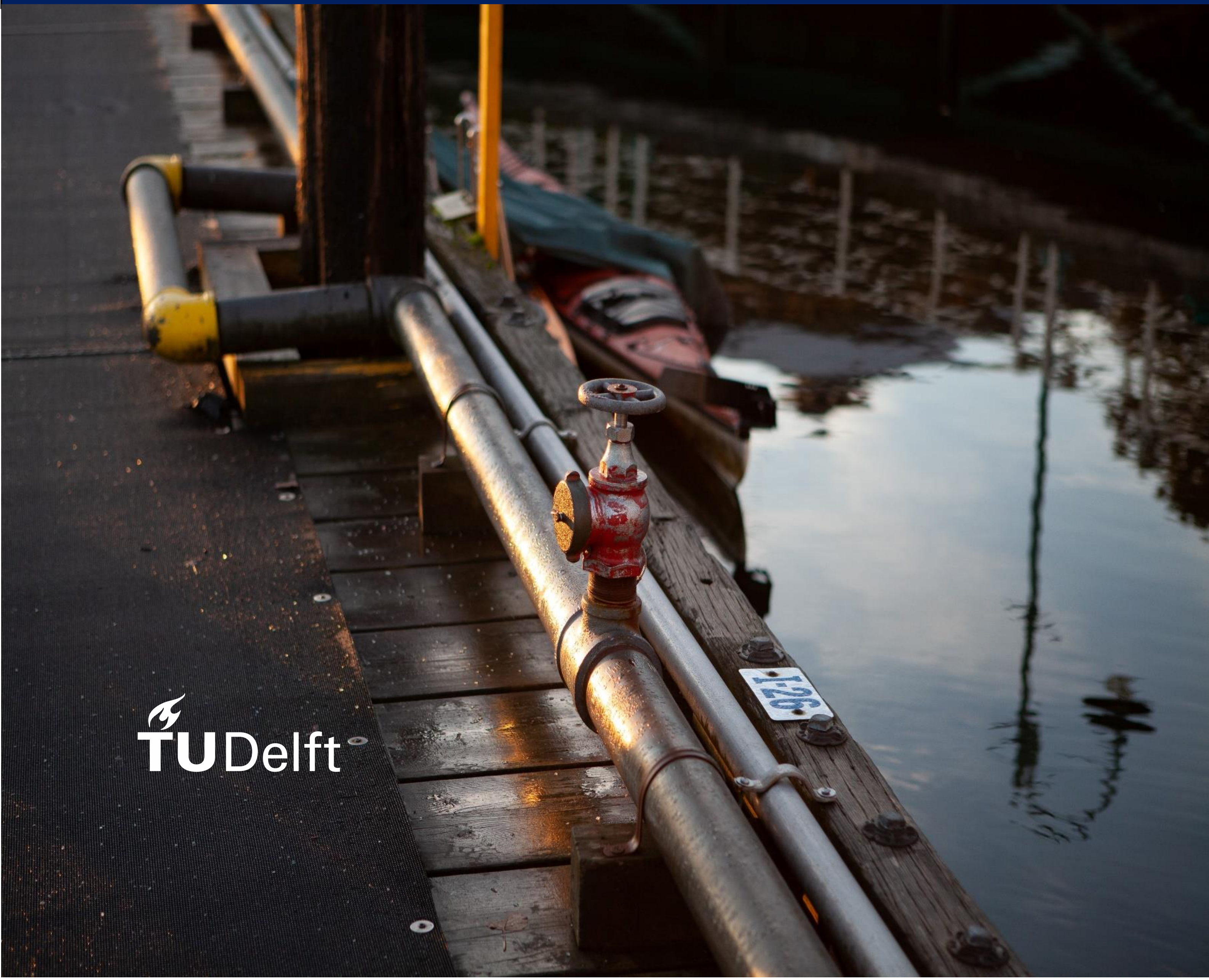
Short-term Water Demand Forecasting at a District Level Using Deep Learning Techniques

Diego Mauricio Corredor Mora



Short-term Water Demand Forecasting at a District Level Using Deep Learning Techniques

By

Diego Mauricio Corredor Mora

To obtain the degree of
Master of Science
in Civil Engineering
track Water Management
at the Delft University of Technology,
to be defended publicly on Wednesday, August 25th, 2021, at 2:30 PM.

Student number:	5145287	
Thesis committee:	Dr. Riccardo Taormina,	TU Delft, chair
	Dr. ir. Mirjam Blokker,	TU Delft
	Dr. Nazli Aydin,	TU Delft
	Dr. Andrea Cominola,	TU Berlin.

Acknowledgments

I would like to thank God for life, and all the opportunities I have been given. I dedicate this work to my parents Arturo and Yaneth, for their unconditional support, trust, and love in each step of my life. Likewise, to Juanita who is my total motivation. I have been far away from home for almost five years and never felt lonely nor hopeless. Thanks for transmitting such strength to me.

I would like to thank my supervisor Riccardo Taormina for his guidance, knowledge, and for pushing me to improve. I had no idea about deep learning before starting this project, and Riccardo spoon-fed me. He gave me the opportunity to learn something different. Moreover, I thank Dr. Andrea Cominola, Dr. Mirjam Blokker, and Dr.Nazli Aydin for being part of my thesis committee, and giving me advice and feedback along the way.

Being far away might be challenging to some people, but I have encountered the best companions who have made this journey smooth, pleasant, and exciting. I thank Natalie Zhang for being next to me every day (through a screen), for supporting me, and for sending me her love from the distance. To my friends and tutors Rafael and Rodrigo for always having an answer to all my silly questions, and for helping me throughout this process. Equally to Maleja, Diana, Alejandro, Mateo and Marisol for their friendship. You guys changed my experience in the Netherlands and made me feel like in a family. Last, but not least, to my friends all over the world who have always been there for me.

Abstract

It is vital for adequate management, and operation of water distribution systems (WDS) to have reliable short-term water demand forecasts. Conventional time-series models present limitations when dealing with non-linear changes in water demand. Thus, it is proposed to employ deep learning algorithms to offer a more reliable forecast. Three models are used, two 1-dimensional convolutional neural networks (1D-CNN), (a simple CNN, and a dilated causal CNN), and a recurrent neural network, particularly a long short-term memory (LSTM) based model. The performance of the models is tested on seven real-life water distribution systems in Italy with different uses and number of users. Also, a comparison with benchmark algorithms based on time-window techniques and pattern-based models is made. Additionally, the use of meteorological variables such as rainfall occurrence, temperature, and relative humidity is intended to test whether there is a positive effect on the forecast. Furthermore, a global model is built taking several years of data for training to test whether this bigger model increases generalization and improves accuracy in comparison to the individual cases. In addition, transfer learning is employed to predict individual cases and a WDS in the Netherlands. Lastly, a bigger global model is built and trained with 14 years of data to improve the performance of transfer learning on the Dutch WDS.

To begin with, it was seen that 1D-CNNs outperformed the LSTM-based model, and the benchmark algorithms using data of the water demand, and a binary index indicating whether it is a weekday or a weekend day for six of the seven case studies. For the remaining case study, the results indicated that there is less than 1% in error between the best benchmark model and the proposed 1D-CNN algorithms. Moreover, the addition of meteorological variables showed to improve the calibration performance of the models but worsened the predictions on unseen data. It was observed that a simple 1D-CNN overfits when adding these extra variables due to its lack of regularization. Also, the global model showed to improve in accuracy compared to the individual models. The use of transfer learning (TL) did not indicate to improve the performance of one of the case studies, nonetheless, TL showed that by only using 75% of the data for training, the model offers a good generalization on the case with sudden changes in demands by the rapid increase of users due to seasonal touristic activities. For the Dutch WDS, TL performed similarly to the individual model, there errors ranged between 15% and 16% using different quantities of data for training. In addition, when having no data for training, the pre-trained model displayed showed lower than using 25% of data for training for the Italian cases. Lastly, the bigger global model performed in the same way as the smaller global model on the Dutch WDS. Also, when having no data for training, the model performed better.

The code of this thesis is available at:
https://github.com/dcorredor20/water_demand_forecasting.git

List of Abbreviations

1D-CNN	One Dimensional Convolutional Neural Network
ANN	Artificial Neural Network
bLSTM	Bidirectional LSTM
CART	Classification and Regression Tree
CNN	Convolutional Neural Network
DL	Deep Learning
DT	Decision Tree
FC	Fully connected
GRU	Gated Recurrent Unit
LSTM	Long Short-Term Memory
RF	Random Forest
RNN	Recurrent Neural Network
SVM	Support Vector Machine
TL	Transfer Learning
MAE	Mean Absolute Error
LR	Learning Rate
TCN	Temporal Convolutional Network
ReLU	Rectified Linear Unit
FNN	Feed Forward Network

Contents

List of Figures

List of tables

1 Introduction

Water Supply systems are under growing stress due to the rapid population increase and economic development worldwide (Esen et al., 2020). Moreover, climate change is expected to rise global water demand (X. Wang et al., 2016), and the exposure of the cities to floods and droughts (Güneralp et al., 2015). Therefore, the importance of water demand prompts water managers and policymakers to promote strategies that encourage the optimization of potable water (Haque et al., 2015). Then, a reliable water demand forecast can be used to determine the water demand for some hours ahead or tomorrow and help the water utilities make decisions on the operations of their treatment plants and wells to meet these demands (Donkor et al., 2014).

There has been a development of different techniques for water demand forecasting based on qualitative methods, univariate time series analysis, moving average and exponential smoothing models, stochastic processes models, time series regression models, scenario-based approaches, and artificial neural networks (ANNs) among others (Donkor et al., 2014). Furthermore, for short-term water demand forecasting, data-driven models, window moving techniques, and pattern-based methods have been employed (Pacchin et al., 2019). In this case, the data-driven model uses the multilayer perceptron (MLP) (which is one of the most common ANNs) to predict water demands of a time horizon of 24-h. An extra binary input indicating whether it is a weekday or weekend day is considered in the model. Also, the pattern-based model provides predictions using periodic patterns that affect the variations of water demand time series such as seasonal and weekly cyclical patterns due to daily water demands, and a daily recurrent pattern of hourly water demands. Next, the moving window model was designed to forecast average water demand with a 48-h horizon taking timesteps of 15 minutes (Bakker et al., 2013). This model accounts for an extra demand due to sprinkles. In addition, another method based on the average moving window was developed. The model uses a narrow window of 4 weeks that moves together with the forecasting time. The length of the window was proposed to capture the seasonal fluctuations in water consumption. Apart from these prediction methods, machine learning (ML) algorithms have been recently used to find patterns in the interaction of sociodemographic and infrastructure factors that could be ignored by conventional statistical models (Villarin & Rodriguez-Galiano, 2019). Equally, Guo et al., (2018) mentioned that deep learning (DL) methods are rarely implemented in water demand forecasting and the performance is uncertain compared to seasonal auto-regressive integrated moving average (SARIMA) models. DL models are composed of multiple layers to understand non-linear representations of data with numerous levels of abstraction. Among the DL methods, RNNs have been widely used for time series and sequence prediction purposes. Equally, 1D-CNNs offer high performance when dealing with 1D signals. Compared to conventional 2D-CNN architectures, 1D-CNNs have lower computational complexity, meaning that they can be used for real-time and low-cost applications. Some of these functions are real-time electrocardiogram (ECG) monitoring, vibration-based structural damage detection for civil infrastructural purposes, monitoring the condition when rotating mechanical machine parts, (Kiranyaz et al., 2021) among others.

The objective of this study is to improve the performance of short-term water demand forecasting at a district level through deep learning techniques such as convolutional neural networks and recurrent neural networks. It is expected that DL algorithms will improve the accuracy of water demand predictions compared to other benchmark algorithms. This will have a positive impact on different actors such as policymakers as they could create strategies to encourage users to reduce the use of water of different appliances at household levels. Moreover, an accurate water demand projection may influence the decision-making on enhancing operations of the existing infrastructure. Also, it will potentially boost the use of new technologies in order to improve data collection and future research for improvement.

This document is structured as follows. It starts with the literature review of the state of art of short-term water demand forecasting. In addition to this section, the research questions are presented. Chapter 2 contains the background of the most relevant concepts of deep learning and water demand modeling. Chapter 3 presents the data that was used for this project, and the way it was cleaned and processed. Chapter 4 shows the proposed methodologies and experiments to answer the research questions. Additionally, the development of the models is explained in detail. Chapter 5 illustrates the results obtained and the corresponding discussion. Lastly, in Chapter 6 the conclusions based on the research question are found together with the recommendations for future work.

1.1 Literature review

The prediction of the water demand has been an important factor for the design and management of water distribution networks. To satisfy the users, infrastructure is planned and built based on demand projections. Over the years researchers and water managers have developed different techniques to forecast water demands for different time horizons. Long-term water demand forecasting goes from periods above 10 years (Mu et al., 2020). Medium-term water demand predictions are made on a weekly to monthly basis (Tiwari & Adamowski, 2015). Guo et al., (2018) claimed that short-term (less than a week) water demand forecasting improves the operations of water treatment plants and pumping stations, the management of an urban water distribution system, water pricing policy assessment, water conservation programs when droughts occur, etc (Jain et al., 2001).

It has been found that ML models showed to outperform traditional forecasting techniques based on time series analysis such as the auto-regressive integrated moving average (ARIMA) and seasonal auto-regressive integrated moving average (SARIMA) (Antunes et al., 2018; Guo et al., 2018). Then, early use of artificial neural networks for water demand forecasting was done by Jain et al., (2001). Two goals were set: 1) The effective use of ANNs for water demand forecasting and 2) occurrence of rainfall is a factor with more relevance than the amount of rainfall when modeling the short-term water demand forecasts. They showed that ANNs outperform regression and time series methods and that at the location of the study (The Indian Institute of Technology) and that there was a higher correlation of the pick daily water demand between the rainfall occurrence than the amount of rainfall. It is concluded that the weekly water demand process is dynamic and driven by the temperature but disturbed by the occurrence of rainfall. Moreover, Adamowski, (2008) compared numerous linear regressions, time series analysis, and ANNs as methods for peak daily water demand forecasting. The author showed that ANNs have better performance for the forecast of peak daily summer water demand. Also, this peak daily demand is higher correlated to rainfall occurrence than the rainfall amount. Also, Ghalehkhondabi et al., (2017) reviewed the methods used for demand predictions published between 2005 and 2015. They included artificial neural networks (ANNs), support vector machines, fuzzy and neuro-fuzzy models, meta-heuristics, and system dynamics. The authors showed that the use of soft computing is effective for short-term water demand forecasting. Tiwari & Adamowski, (2015) found that when having only a year of data, ANNs would not work effectively. Then, they implement a wavelet-bootstrap machine-learning approach showing that a hybrid wavelet-bootstrap-artificial neural network (WBANN) model is useful to assess uncertainty in terms of confidence bands. Also, the study shows that wavelet-artificial neural network (WANN) and WBANN models are effective despite the lack of data. Pacchin et al., (2019) compared six different models to predict the short-term water demand of seven districts in Italy. These models are an ANN, a pattern-based model, a probabilistic Markov chain-based model, a naïve benchmark model, and two pattern-based models employing moving window techniques. The authors found that the accuracy of data-driven and pattern-based techniques is comparable. Guo et al., (2018) explored the capability of deep learning in short-term water demand forecasting. The authors developed a gated recurrent unit network (GRUN) that forecasts water demand at 15-min and 24-hour resolution. The GRUN model was compared with ANN and SARIMA models. it was found that the GRUN model outperformed the ANN and SARIMA models for both time horizon forecasts. Later, Mu et al., (2020) developed a model based on a long short-term memory (LSTM) to forecast short-term water demand at a district level for the city of Hefei, China. The authors showed that LSTM displays a high performance when predicting water demands at high resolution (15 minutes).

For time series forecasting, Temporal convolutional networks TCN have been proposed (Lara-Benítez et al., 2020; Lea et al., 2017). TCNs were designed to work with sequential data and are trained faster than RNNs (Oord et al., 2016). In principle, TCNs are based on 1-dimensional convolutional neural networks (1D-CNN). Moreover, for water resources, the application of 1D-CNNs has been shown to perform better than a multi-layered perceptron (MLP) when predicting monthly rainfall (Haidar & Verma, 2018). Lang et al., (2019) introduced the application of CNN for electricity load forecasting. It was shown that the use of simple 1D-CNN architectures can achieve a good forecast. Also, Lara-Benítez et al., (2020) uses TCNs to forecast energy demand. They tested the performance of TCNs on the national electric demand and the power demand of charging stations for electric vehicles. Hewage et al., (2020) showed that TCNs produce better predictions than an LSTM algorithm and other conventional machine learning models for weather forecasting. These are some examples of the uses of CNNs for time series forecasting. In addition, the previous studies concluded that TCNs performed better than RNNs. Guo et al., (2018) mentioned that the use of DL has been rarely used for water demand forecasting. In addition, although the addition of meteorological has been proposed to improve the forecasts of conventional water demand models, no studies have employed DL algorithms for this purpose.

On the other hand, the training of ANNs can require special hardware and large amounts of data. To tackle these challenges, a method named transfer learning (TL) has been implemented. TL takes an additional data source apart from the regular training data, then, it delivers knowledge from a source task to enhance learning in a target task (Fu & Aldrich, 2018). Kratzert et al., (2018) showed the potential of using LSTMs for runoff simulation from meteorological data. The authors also proposed to test the ability of the LSTMs on single catchments, then a general model was trained with the data of multiple catchments to test whether this improves the prediction on another single catchment. Lastly, the authors employed the pre-trained model based on the general model to test the performance of individual catchments. The motivation for this is to show the use of DL and TL when there is not enough data or there are ungauged locations. Nonetheless, TL has not been applied for water demand forecasting purposes. Thus, its capabilities to train faster with fewer amounts of data opens the opportunity to be exploited to predict water demands.

1.2 Research Questions

DL has been widely used for a lot of different purposes. However, it has been noticed that these algorithms are rarely used for water demand forecasting. Therefore, this work is based on the following questions.

1. *Can 1D-CNNs outperform RNNs and other existing algorithms for water demand forecasting?*
2. *Does external data, such as meteorological data, improve the performance of water demand forecasting?*
3. *Can a general model be used to improve the predictions of water demand in individual cases?*
4. *Can transfer learning improve the performance of deep learning algorithms for water demand forecasting for cases in which the years and the geographical locations are different?*
5. *Can transfer learning be used to predict accurately the water demands of a water distribution system located in another country?*
6. *Can a bigger global model be trained to improve the performance of transfer learning when being employed to predict the water demands of a water distribution system located in another country?*

2 Background

In this chapter, the background and conceptual framework are explained. An introduction of what deep learning is and some of its applications is presented. Next, the main algorithms that are used in deep learning for sequential data modeling. Lastly, additional concepts to understand how ANNs are trained and optimized together with some key performance indicators (KPIs) that are proposed to measure the error in regression models.

Although the terms machine learning and deep learning are not new, they sound around more often due to the exponential increase in computational power. ML and DL present some differences. ML has shown some drawbacks in performance when dealing with large amounts of data whereas DL can keep good performance of models. Also, DL runs operations with high complexity and can find and extract features in labeled, and unlabeled data effectively (Simplilearn, 2021). For time series forecasting, some DL architectures have been used. Moreno et al., (2011) compared the performance of an MLP, Radial Basis Function (RBF), Generalized Regression Neural Network (GRNN) and, RNN. The results showed that the RBF and RNN models presented the best results. However, RBF requires to strictly use a single hidden layer (Chandradevan, 2017).

On the other hand, it was shown that 1D-CNNs have been used recently for the forecasting of time series data. Equally, the success of RNNs in many fields opens an opportunity to apply them to forecast short-term water demand.

2.1Deep Learning

Deep learning can be defined as a branch of machine learning that uses multiple layers of artificial neural networks (ANNs). The use of these deep architectures enables the extraction and learning of complex features from data. DL has been applied to tasks such as speech recognition, visual object recognition, object detection, and other applications (LeCun et al., 2015). The core of DL is to obtain hierarchical features of data in which the higher-level features are defined by the lower-level features (Deng, 2014).

2.1.1 Artificial Neural Networks

Artificial neural networks were designed to mimic the function and structure of the human brain (Brownlee, 2019). The human brain contains more than 100 billion neurons and even more connections that enable learning quickly from experiences. The brain saves the information as patterns. The whole idea of storing and using these patterns to solve complicated tasks opened a field in computing (Anderson & McNeil, 1992).

An ANN can have many layers, Figure 1 displays the basic structure of an artificial neuron. A feed-forward network (FNN) is composed of three layers, the input layer which connects the input variables. Next, the hidden layer and lastly the output layer.

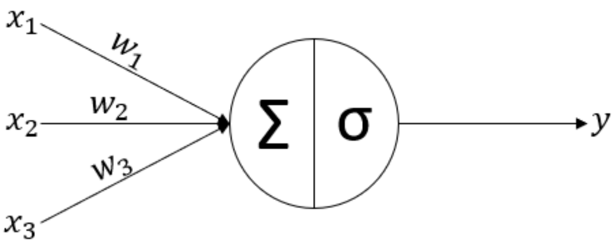


Figure 1. Basic artificial neuron.

Figure 1 shows the inputs can be represented by x_n. Each of these inputs is multiplied by a weight represented by w_n. Then, the node is composed of the sum of the previous multiplications and σ represents an activation function that introduces non-linearity. If bias b is introduced in the operation, the output is represented by the following equation:

$$y = \sigma(w^T x + b) \qquad (1)$$

ANNs can be deeper in the sense that more hidden layers can be added. Figure 2 shows the structure of a deep ANN composed of three hidden layers apart from the input and output layers. The combination of multiple nodes enables the network to learn complex functions.

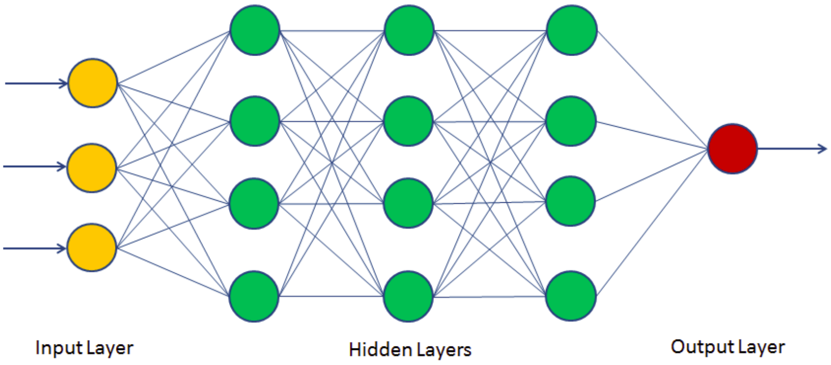


Figure 2. Fully connected layer (Navlani, 2019).

There are a lot of variations in neural networks based on the basic structure of artificial neurons. This thesis, however, will focus on two types: convolutional neural networks (CNNs) and recurrent neural networks (RNNs). Among the RNNs, LSTM models are used.

2.1.2 Convolutional Neural Networks

In principle, CNNs were designed for pattern recognitions within images (O’Shea & Nash, 2015). Also, CNNs succeed at detection, segmentation recognition of items and zones in images. CNNs process data that comes in the style of arrays. 1D-CNNs are used for signals and sequences, 2D-CNNs are used for images or audio spectrograms, and 3D-CNNs for video or volumetric images (LeCun et al., 2015). As the objective of this work is to predict water demand, the arrays can be considered as 1D. Figure 3 shows the architecture of a 1D-CNN.

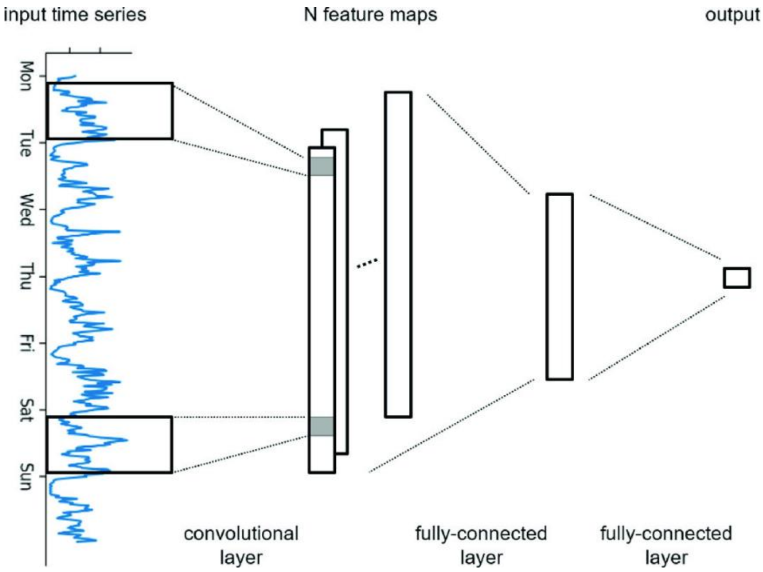


Figure 3. CNN applied to a 1D signal data (Lang et al., 2019)

1D-Convolutional Layers

Within these layers, the input is convolved with several filters or kernels. A kernel k is applied to the data points of the input signal and on every time step the convolution of the filter with the respective overlapping region is calculated. The size of the kernel is a hyperparameter and the larger the kernel size, the larger the receptive field. Figure 4 displays the convolutional operations. The x_n values represent the input variables, w_n are the weights to be convolved with. It is shown in this case a $k = 3$. Finally, c_n represent the results of the convolutions in an output feature map.

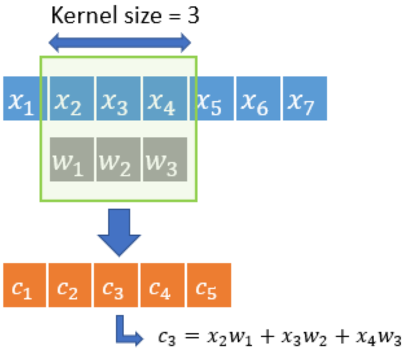

Kernel size = 3
x_1 x_2 x_3 x_4 x_5 x_6 x_7
w_1 w_2 w_3
c_1 c_2 c_3 c_4 c_5
$c_3 = x_2w_1 + x_3w_2 + x_4w_3$

Figure 4. Convolution operation with k=3.

During training, the model is forced to learn the values of every filter.

A conventional convolution like the one shown above has some drawbacks when predicting time series data. When doing convolutions, the output feature map is smaller than the input feature map due to the kernel size. This may lead to a loss of information of the time dependencies. Also, this standard convolution takes data points from the future and the past, this is called a non-causal convolution. This violates the order in which data are modeled and it is inconvenient for real-time applications such as text generation in which one should avoid looking at the future. The problem of taking information from future timesteps can be solved by adding causality to the convolutions.

Causal Convolutions

This type of convolution avoids taking data from future time steps so that the model does not violate the order in which data is trained. Therefore, the forecast $p(x_{t+1} \mid x_1, \ldots, x_t)$ estimated by the model at timestep t must not depend on the future timesteps $x_{t+1}, x_{t+2}, \ldots, x_T$ (Oord et al., 2016). Figure 5 shows what a conventional convolution looks like.

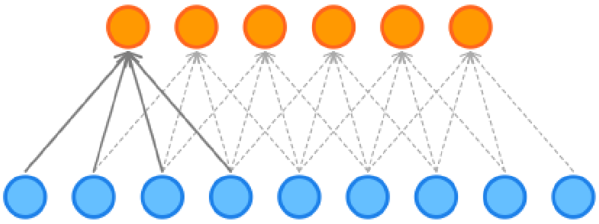


Figure 5. Conventional convolution with k=4 (Batzner, 2019).

Figure 6 shows what a causal convolution looks like for a kernel of size k = 3. It is seen that to predict the first timestep (orange color), only the first input timestep (blue color) is considered with 2 zeros padded on its left. In addition, the size of the input feature map is the same as the output feature map.

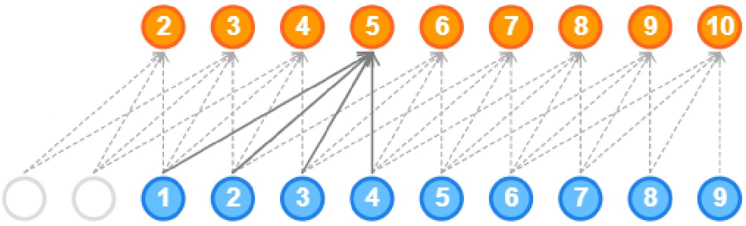


Figure 6. Causal convolution with left padding with k=4 (Batzner, 2019).

Causal convolutions, however, need deep networks with multiple layers of large kernel sizes to expand the receptive field. This will increase the number of parameters in the network making it computationally expensive (Oord et al., 2016). This issue can be solve using dilated convolutions.

Dilated Convolutions

In a dilated convolution, the receptive field is expanded without necessarily increasing the number of weights. This way, the kernel increases the receptive field. Formally, when using 1D-CNNs, the sequence input x $\in \mathbb{R}^n$ and a kernel $f\colon \{0, \ldots, k-1\} \rightarrow \mathbb{R}$, the dilated convolution operator F on the element s of the sequence is defined as follows:

$$F(s) = (x *_d f)(s) = \sum_{i=0}^{k-1} f(i) \cdot x_{s-d \cdot i} \qquad (2)$$

Where d is the dilation factor, k is the kernel size, and $s - d \cdot i$ accounts for the direction of the past (Bai et al., 2018). Figure 7 displays a stack of dilated causal convolutional layers. The hidden layers are stacked with different dilation factors increasing exponentially.

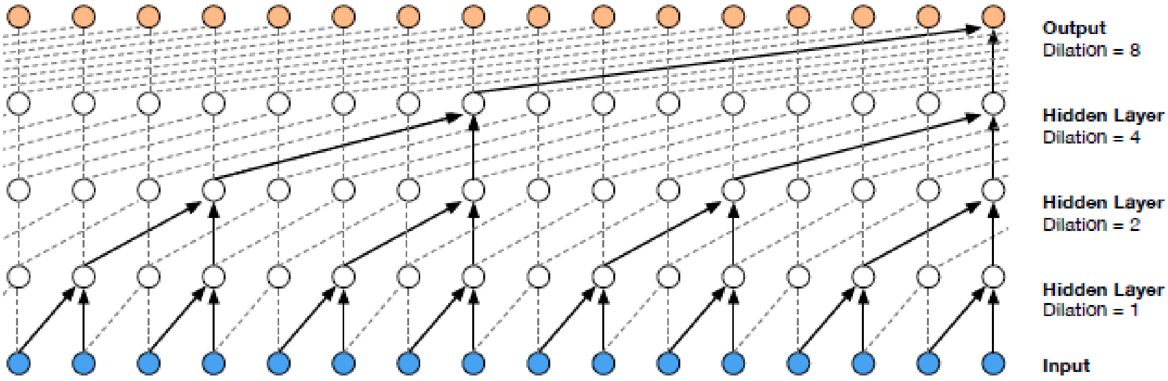


Figure 7. Visualization of a stack of dilated causal convolutional layers (Oord et al., 2016).

2.1.3 Recurrent Neural Networks

An RNN is a type of ANN designed to process sequential information. RNNs are widely used for speech synthesis, music generation, image captioning, video analysis, natural language processing, and time series forecasting among others (Lipton et al., 2015). As RNNs use sequential data, it means that they consider past information that will influence what can happen in the future. This DL architecture can "remember" previous input hidden states. By looping internally, RNNs can retain information. Figure 8 displays a piece of a neural network. A receives the input x_t, processes and outputs h_t, then, a loop allows information to go through the next timestep.

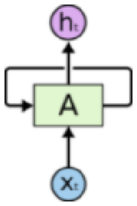


Figure 8. Recurrent neural network representation (Olah, 2015).

RNNs, nonetheless, present some issues when preserving or remembering long-time dependencies. The more information there is, the higher the chance that back-propagation gradients accumulate, vanish, or explode resulting in an inefficient and slow training process (LeCun et al., 2015). Hochreiter & Schmidhuber, (1997) presented LSTM as a solution to the vanishing gradient problem in regular RNNs.

Long Short-Term Memory

LSTMs contain a more complex structure than the standard RNNs. The cell state, which contains three gates, is the difference between LSTMs and regular RNNs. These gates allow the information to be "remembered" or "forgotten" when processing sequences. In principle, the gates are neural networks that figure out what information is permitted in the cell state (Phi, 2018).

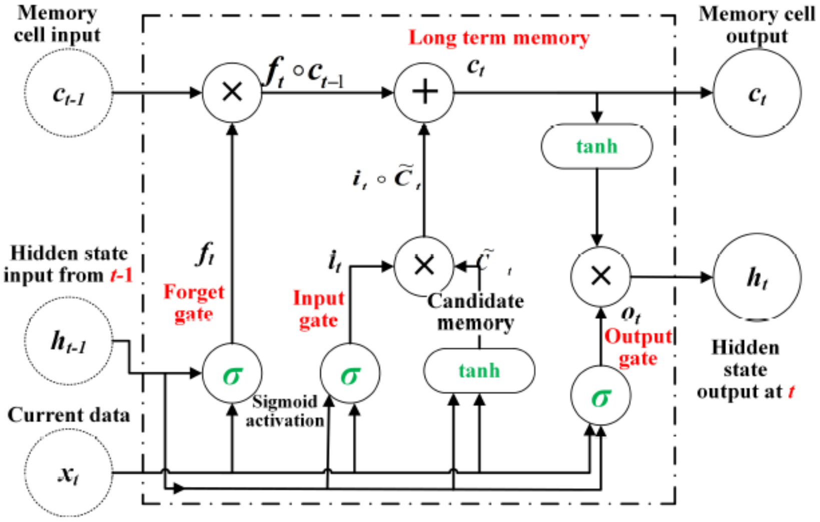


Figure 9. Structure of an LSTM cell (Zebin et al., 2018).

Figure 9 displays the structure of an LSTM cell in detail. Appendix A contains a detailed explanation of how the LSTM works.

2.1.4 Additional deep learning concepts

The following concepts will help the reader understand how ANNs are trained and optimized

Activation functions

In neural networks, activation functions transform the weighted sum of the input into an output. These transformations add non-linearity to the network. As deep neural networks have many layers, non-linearity is needed. The activations functions work as a decision-making function that detects the presence of a certain neural feature. The input will take inputs $-\infty$ to $+\infty$ but outputs values between {0,1} or sometimes {-1,1}. Some of the conventional activation functions are shown below.

- **Linear.** The linear activation function, shown in Figure 10 a, is defined as:

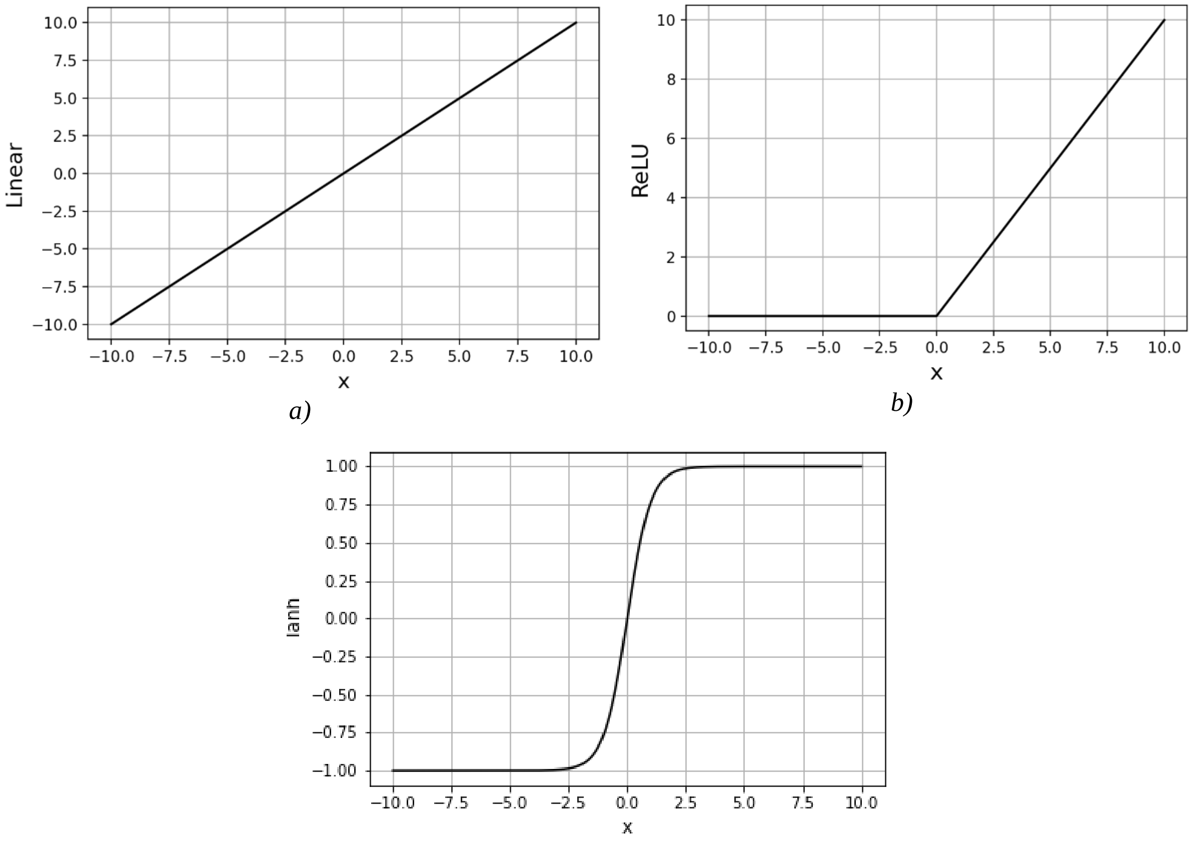
$$y = mx \tag{3}$$

- **ReLU.** The rectified linear unit ReLU, shown in Figure 10 b, is defined as:

$$RelU(x) = \max(0, x) \tag{4}$$

- **Tanh.** The hyperbolic tangent function, shown in Figure 10 c, is defined as:

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{5}$$



a)

b)

c)

Figure 10. Activation functions. a) Linear, b) ReLU, c) Tanh.

Loss function

Another important concept is the loss function. It defines how close the output of the model is to the real values. Depending on the goal of the network, either for classification, clustering, or regression, the loss function is chosen. The objective is to minimize or maximize loss function. As this work is a regression problem, Mean Absolute Error (MAE) has been proposed to be the objective function when training the networks This loss is the mean overseen data of the absolute differences between real and predicted value. The following equation presents the MAE loss function:

$$J(y_i, \hat{y_i}) = \frac{1}{N}\sum_{i=0}^{N} |y_i - \hat{y_i}| \tag{6}$$

Where J denotes the loss function, $\hat{y}_i$ the predicted value and N the number of samples.

The purpose of training neural networks is to find the correct weights to obtain the desired output. Thus, the loss function needs to be minimized in this case. As deep neural networks are trained to learn a large number of parameters, the conventional procedure of computing the derivatives to find the optimal point is not feasible. The process of training neural networks depends on the back-propagation algorithm which essentially takes the information of the loss function $J(w, b)$ to flow backward through the neural network to calculate the gradient. Since the output of the neural network depends on the weights w and the bias terms b, the back-propagation algorithm uses the partial derivatives to update w and b as follows:

$$w := w - \alpha \frac{\partial J(w, b)}{\partial w} \tag{7}$$

$$b := b - \alpha \frac{\partial J(w, b)}{\partial b} \tag{8}$$

Where α represents the learning rate. This hyperparameter tells the amount of the update by the gradient descent algorithm. Gradient descent is a mode to minimize the loss function $J(\Theta)$ which updates the parameters in the opposite direction of the gradient of the loss function $\nabla_{\Theta} J(\Theta)$ concerning the parameters (Ruder, 2016).

ADAM optimization algorithm

The adaptive moment estimation (ADAM) algorithm was proposed by Kingma & Ba, (2017) as a gradient-based optimization of stochastic objective functions. In principle, ADAM computes an exponential weighted moving average of the gradient v_t. Next, the obtained gradient is squared. In addition, ADAM stores an exponentially decaying average of previous gradients m_t. Both, the moving average of past squared and decaying average of past gradients can be obtained as:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) \frac{\partial J}{\partial w} \tag{9}$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) \left[\frac{\partial J}{\partial w}\right]^2 \tag{10}$$

Where m_t and v_t are estimates of the mean, and the variance of the gradients, respectively. The authors suggest values at 0.9 and 0.999 for β_1 and β_2 respectively.

Key Performance Indicators (KPIs)

As defined above, MAE was the proposed metric error to compute and optimize the loss function. Additionally, the coefficient of determination (R^2) was employed to compare the performance of the models. In principle, when this value is closed to 1, it indicates that is a highly reliable model, and when it is close to zero otherwise. R^2 is defined as:

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} \tag{11}$$

Where SS_{res} corresponds to the sum of squared of residual and SS_{tot} is the total sum of squares.

In addition, since MAE is a scale-dependent accuracy measure, it is not possible to compare case studies when having different average water demands. Pacchin et al., (2019) proposed to use MAE% to compare models. MAE% solves the scalability issue (Vandeput, 2019). MAE % is defined as:

$$MAE\% = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{e_i}{\mu_{obs}}\right| * 100 \tag{12}$$

Where N is the amount of data for the specified period, e_i is the error obtained by subtracting the predicted water demand from the real water demand and μ is the mean of the real values.

3 Data Preparation and Processing

3.1 Water Demand Data

Pacchin et al., (2019) collected data from seven different districts in Northern Italy. These data were collected over two years at a time resolution of 1 hour. The data refer to real water distribution networks in which six are residential/industrial, and one corresponds to a touristic area. Table 1 shows the average demands together with the number and type of users. Additionally, the case studies (CS) correspond to CS1 = Castelfranco Emilia, CS2 = Ferrara, CS3 = Cento, CS4 = Vigarano Mainarda, CS5 = Bondeno, CS6 = Ferrara Nord-Ovest, CS7 = Lido di Spina respectively.

Table 1. Average water demands and case studies.

Case Study	CS1	CS2	CS3	CS4	CS5	CS6	CS7
Number of users	7000	120000	9000	2500	7000	20000	300 - 3500
Type of users	Res	Res/Ind	Res/Ind	Res	Res	Res/Ind	Res/Tour
Average demand [L/s] y1	54.04	934.41	100.05	24.12	52.96	177.14	28.86
Average demand [L/s] y2	65.24	965.71	95.95	24.68	56.77	177.41	28.46

It is important to highlight that CS2, CS3, CS4, CS5 and, CS6 are managed by the same water utility company. Also, these five districts are located close to each other, and the data were recorded in 2014 and 2015. The data of CS1 correspond to the years 1998 and 2000. Likewise, data of CS7 were recorded in 2013 and 2014. Apart from the hourly water demands, the dataset of Pacchin et al., (2019) has additional data such as the day of the week and whether there was a holiday, thus, it is given an index of 1 for the weekend days and holidays and 0 for the weekdays. This information is valuable because it is used as a binary index input for the models. This decision is made based on the idea that the water demand is higher on weekend days and public holidays than on weekdays. Figure 11 displays the hourly water demand of CS2 in 2014 during the first weeks of February. From Figure 11, it is observed that the time series starts with two peaks showing the demand of two consecutive weekend days (Saturday and Sunday) followed by the five-weekday demand. Then, once again the high peaks are seen.

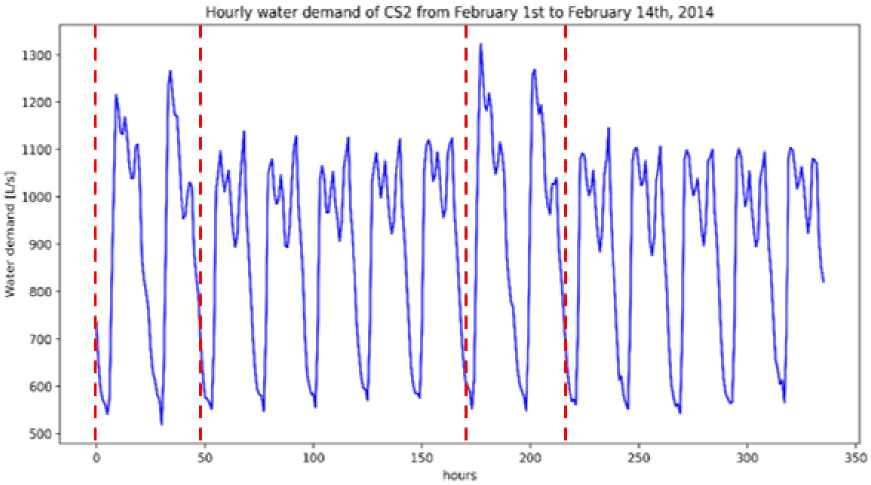


Figure 11. Hourly water demand of CS2 from February 1st to February 14th, 2014.

3.2 Meteorological Data

The Regional Agency for Prevention, Environment, and Energy of Emilia-Romagna (Arpae) [1] is in charge of concession, monitoring, control, and prevention to encourage sustainability, health and land protection of the resources, and environmental consciousness. Arpae's website has a section to download meteorological data through their application named Dext3r [2]. This application allows the user to select the period, variables, and location. This latter option can be chosen by basin, province, map, and more. As previous research showed that there is a correlation between the peak daily water demand and the rainfall occurrence (Adamowski, 2008), it was decided to add this idea together with some other climatological variables such as maximum, average, and minimum hourly temperatures, hourly relative humidity.

3.3 Water Demand Data (The Netherlands)

Hourly water demand from a tourist place in the north of The Netherlands was provided. It contains data of 4 years (2014-2017). The table below shows the average water demand for the 4 years. The number of users varies from 3000 to 15000.

Table 2. Hourly Water Demand (The Netherlands)

Year	2014	2015	2016	2017
Average demand [L/h]	64.30	66.10	66.82	69.18

[1] https://www.arpae.it/it/arpae/arpae
[2] https://simc.arpae.it/dext3r/

3.4 Data Pre-processing

The following flowchart displays the pre-processing that the water demand and the other variables need before being transformed into the input of the models.

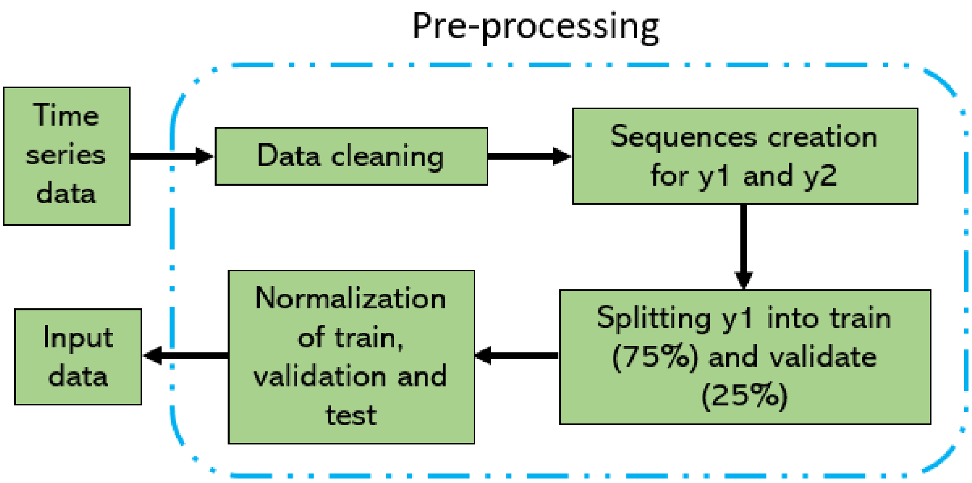

Pre-processing
Time series data
Data cleaning
Sequences creation for y1 and y2
Splitting y1 into train (75%) and validate (25%)
Normalization of train, validation and test
Input data

Figure 12. Data pre-processing flowchart.

3.4.1 Data cleaning and correlation between water demand and climatologic variables

Before building the sequences of the meteorological data, there was a process of cleaning these data. For instance, some missing values were interpolated using interpolation methods. Among the 8760 values (hours in a 365-days year), there were less than 2% of missing values, and there were no outliers. When there was a considerable number of missing values between 2-5% the next closest station was used to complete those data points.

Next, in order to choose the variables to use for training, linear correlations were performed. The Pearson’s correlation coefficient is calculated as follows:

$$r_{x,y} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{13}$$

Where n is the sample size, x_i, y_i are the sample points indexed with i.

The table below shows the results of the correlation coefficients of the climatological variables with the water demands of the CS2 to CS7. It can be seen that the values of CS1 are missing. This was not applied to this case study as the data corresponds to 1998 and 2000 and there was not meteorological available data for these periods. The meteorological variables, except for precipitation amount, were not available. However, the correlation between rainfall amount and occurrence has the same pattern as the other case studies.

Table 3. Correlation of meteorological variables with water demand for CS2 up to CS7 in 2014.

Variable	CS2	CS3	CS4	CS5	CS6	CS7
Precipitation amount	-0.026	-0.017	-0.019	-0.012	-0.025	-0.011
Precipitation occurrence	-0.065	-0.055	-0.064	-0.048	-0.057	-0.109
Maximum temperature	0.431	0.396	0.407	0.367	0.345	0.705
Minimum temperature	0.418	0.337	0.396	0.341	0.329	0.703
Average temperature	0.425	0.387	0.402	0.354	0.337	0.705
Relative humidity	-0.428	-0.36	-0.382	-0.317	-0.381	-0.526

It is found that in general, the maximum temperature presents a higher correlation ranging from 0.34 to 0.42 with water demand than the average and minimum temperatures in the districts with residential and industrial users. CS7 shows the highest correlation between the maximum temperature and water demand at 0.705. This is expected as the demand increases in the summer. Additionally, as the temperature rises, relative humidity decreases. Thus, the relative humidity shows a negative correlation with water demand ranging between -0.317 to -0.526. It is seen that there is a higher correlation between the precipitation occurrence than the amount of precipitation. Therefore, the additional data are the precipitation occurrence, maximum temperature, and relative humidity.

3.4.2 Building sequences

The first goal is to build a dataset of sequences to be the input in the deep learning models. The sequences were made through a simple function. This function allows to automatically convert the time series data to sequences. Furthermore, the function also returns the target sequences, thus, the final dataset is in the {<features>, <target>} format. This is the way any machine learning model is trained; thus, it can be interpreted as a supervised learning model.

To understand how the sequences are created, the lookback concept is implemented. The main idea is to use the data from previous time steps to predict the next time step. For instance, the feature values at a given time step *x(t)* are *x(t-1), x(t-2), …., x(t-n)* where n is the lookback. For this study, *n* equals 168 which is equivalent to the total hours in seven days. This information is used to predict the next 24-time steps. Figure 13 displays this idea. It can be seen that in order to get a target of 24-time steps, the previous 168-time steps are taken. Also, Represents the number of sequences or samples that are possible to build. Then, the function moves a step forward to create the next *N + 1* sample.

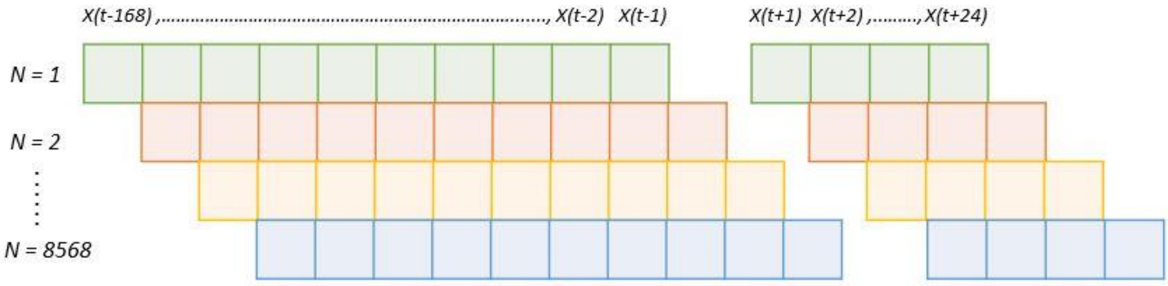

X(t-168) ,…………………………………………………………………………, X(t-2) X(t-1)
X(t+1) X(t+2) ,………, X(t+24)
N = 1
N = 2
N = 8568

Figure 13. Building sequences.

The same function is used to build the sequences of the other variables (temperature, relative humidity, rainfall occurrence, binary index of the type of day).

3.4.3 Normalization, standardization, and dataset subdivision

After data are converted into sequences, every array of variables is normalized to avoid problems of signal saturation (Hsu et al., 1995). The water demand, maximum temperature, and relative humidity are scaled in the interval [0,1]. Formally, normalization can be done as:

$$x_{norm} = \frac{x_i - x_{min}}{x_{max} - x_{min}} \tag{14}$$

Where x_i are data points $(x_1, x_2, \dots, x_n)$, and x_{norm} are the normalized data points. Once the models are trained, there is a de-normalization and de-scaling procedure to analyze the real values.

Additionally, standardization is also used to build a bigger model. For this case, the distribution of the water demands differs in most districts due to the number of users. Standardization is applied as follows:

$$Z = \frac{x_i - \mu}{\sigma} \tag{15}$$

Where x_i are data points, μ is the mean, σ is the standard deviation and Z is the Z-score or the standardized data points. The inverse process of de-standardization and de-scaling is performed after. Figure 12 shows the flowchart of the pre-processing stage. Furthermore, as there are two years of recorded data, year 1 is used for training a validation splitting it into 75% and 25% respectively. The data of year 2 is, therefore, used for testing the models.

Figure 12 shows that the last step is the data normalization after splitting to train and validation. The *MinMaxScaler* object of *Scikit-learn* is used. This object comes with the *fit( )* function that estimates the maximum and minimum observable values. Then, this function is applied to the dataset for training. After that, the *transform( )* function with the normalized data is applied to the training, validation, and test datasets. Next, the input data is ready to feed the models.

4 Experimental Settings

This chapter explains the proposed methodology, experiments, and a detailed description of the proposed models. Experiments are presented in the same order as the research questions to understand why they were proposed in that way. It starts with the first two research questions that involve the comparison of the performance between RNNs with 1D-CNNs together with benchmark models, and the performance when adding external variables. Then, how a bigger model is built to find whether predictions on individual models are better when having more data. Next, how TL is applied on cases other cases with different years of data, location, and types of users. Lastly, how TL is employed to predict water demands of another water distribution system in a different geographical position.

4.1Experiments

4.1.1 Modeling with water demand data and additional variables

To answer the first and second research questions, the following methodology and experiments were proposed. Once the data are normalized and scaled, it is ready to go into the models for training. Three algorithms were used to compare their performance: an LSTM, simple CNN, and dilated causal convolution (see 4.2.).

Table 4 shows the different combinations of variables that feed the models. Systematically, the variables were added one by one to train the models. For instance, the first input is the sequences with water demand only. Then, the precipitation occurrence is added, as well as the rest of the variables. Next, two variables are added to the water demand. The same procedure is done until all the variables at the same time are employed.

Table 4. Combination of input variables for the models. WD = Water Demand, P = Precipitation occurrence, RH = relative humidity, T = Temperature, D = Binary index for week/weekend day.

No. of variables	Input variables
1	WD
2	WD + (P or RH or T or D)
3	WD + D + (P or RH or T)
4	WD + D + P + (RH or T)
5	WD + D + P + RH or T

The modeling was done for all case studies. However, for CS1 in which there is no meteorological data, only experiments with the type of the day and rainfall occurrence were performed. Figure 14 displays the flowchart showing the proposed methodology to train the models. As mentioned in 3.4, there is a process of de-normalization and de-scaling to obtain the predictions.

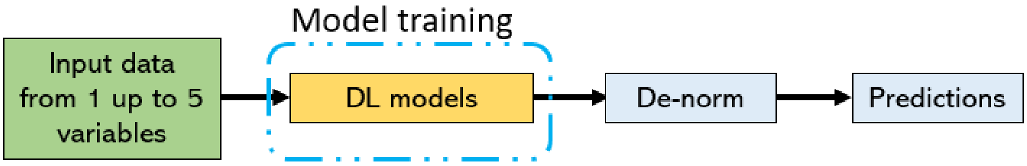


Figure 14. Flowchart of DL models training, de-normalization, and predictions.

4.1.2 Building a global model

Another approach considered was to create a bigger dataset. Instead of feeding the DL models with individual datasets, a more robust one was built. As mentioned in section 3.1, CS2, CS3 CS4, CS5, and CS6 are managed by the same water utility company. Also, it is assumed that the meteorological conditions of these areas do not differ much from each other. These five case studies were used to create the bigger dataset, with five years of data. There is, however, a difference when comparing with the previous methodology. There was no normalization but standardization. This was applied to the datasets individually to, subsequently, concatenate them. The standardization was performed through the *StandardScaler( )* object of *Scikit-learn*.

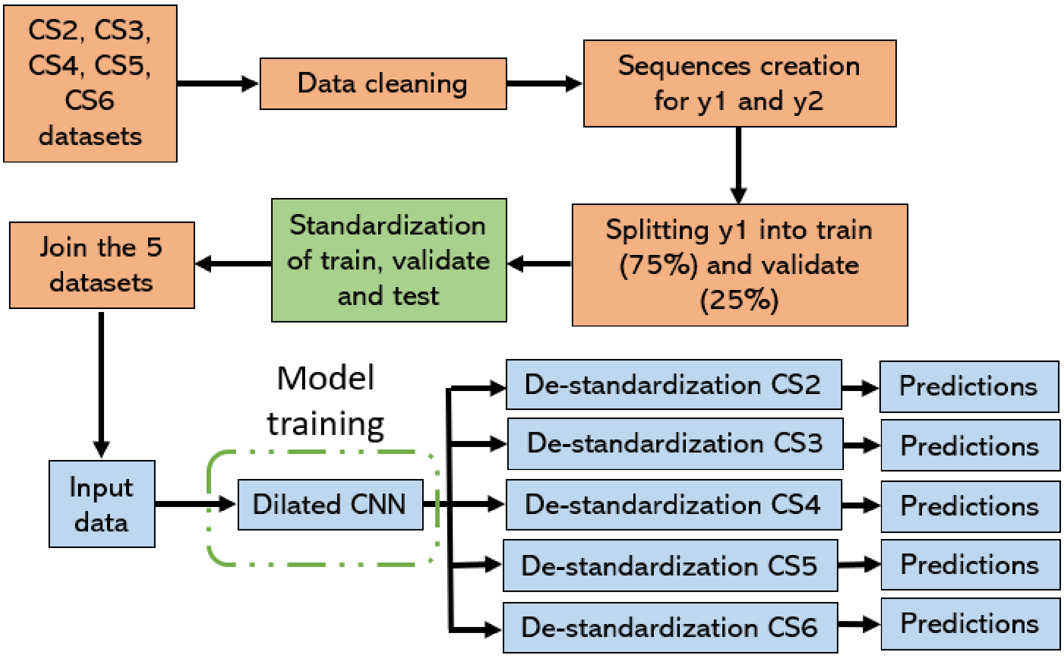
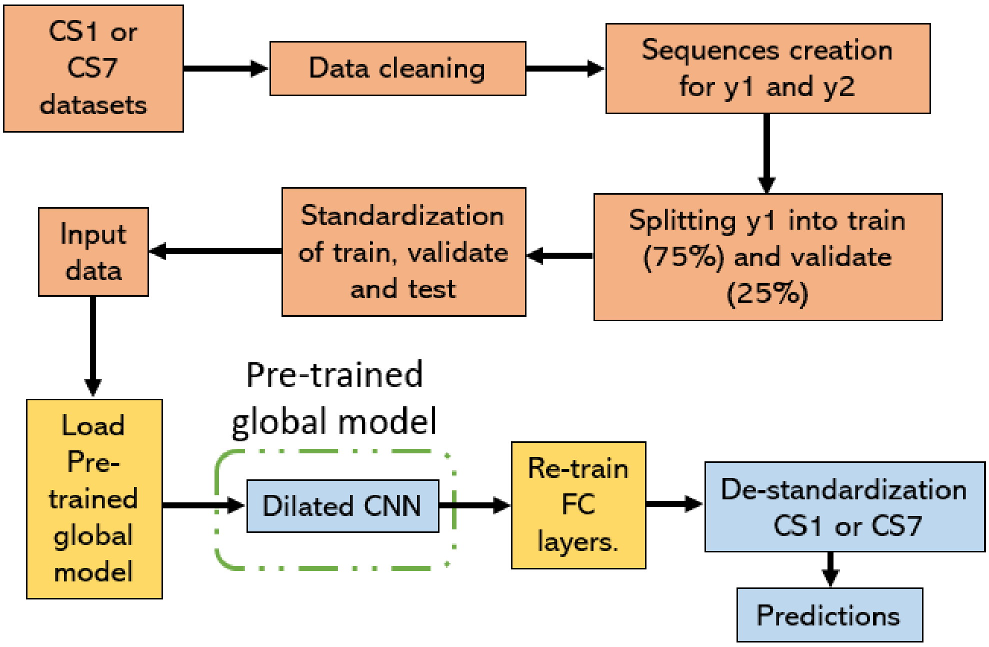


Figure 15. Flowchart of the global model construction.

The *fit( )* function has stored the mean (μ) and standard deviation (σ) of the water demands of every dataset. In order to test the model with the individual datasets CS2 to CS6, it is necessary a process of de-standardization and de-scaling of the datasets with the respective scaler. These predictions obtained from a bigger dataset will be compared with the ones computed individually.

4.1.3 Transfer learning approach

As it was proposed by Kratzert et al., (2018), a global or general model was built. The data of CS2 to CS6 were used, but not CS1 and CS7. TL is proposed as a solution to forecast demands in CS1 and CS7. The "best" pre-trained model is saved. Then this model is uploaded, and the last fully connected layer(s) are re-trained to obtain predictions. Then, these findings are compared with the results obtained from the individual datasets CS1 and CS7. An additional approach to understand the capabilities of TL is to train models with fewer data (Donges, 2021). To compare the performance of this approach, the errors will be obtained using 0, 25, 50, and 75% of the data of y1 for CS1 and CS7 individually. Next, the procedure is repeated but employing the pre-trained model. Thus, 25% of data correspond to the first three months of the year, 50% to half of the year, and 75% to 9 months. The following flowchart shows the process to implement TL and obtain predictions.



CS1 or CS7 datasets
Data cleaning
Sequences creation for y1 and y2
Splitting y1 into train (75%) and validate (25%)
Standardization of train, validate and test
Input data
Load Pre-trained global model
Pre-trained global model
Dilated CNN
Re-train FC layers.
De-standardization CS1 or CS7
Predictions

Figure 16. Flowchart of transfer learning implementation.

The same method is applied to data from a real Dutch water distribution system to test the performance of the pre-trained global model. As there are 4 years of data, the proposed experiments are to take the data of 2014 and predict 2017. Then, take 2014 and 2015 to predict 2017. Lastly, take 2014, 2015, and 2016, to predict 2017 instead of using 25, 50, and 75% of data for training.

4.1.4 Building a bigger global model

Same as 4.1.2, a bigger global model was proposed to apply TL to the Dutch case study. For this experiment, all data from Italy was used to train a more robust model. Thus, the new global model was trained with 14 years of data instead of 5. The figure below displays the flowchart of the process to build and train this model.

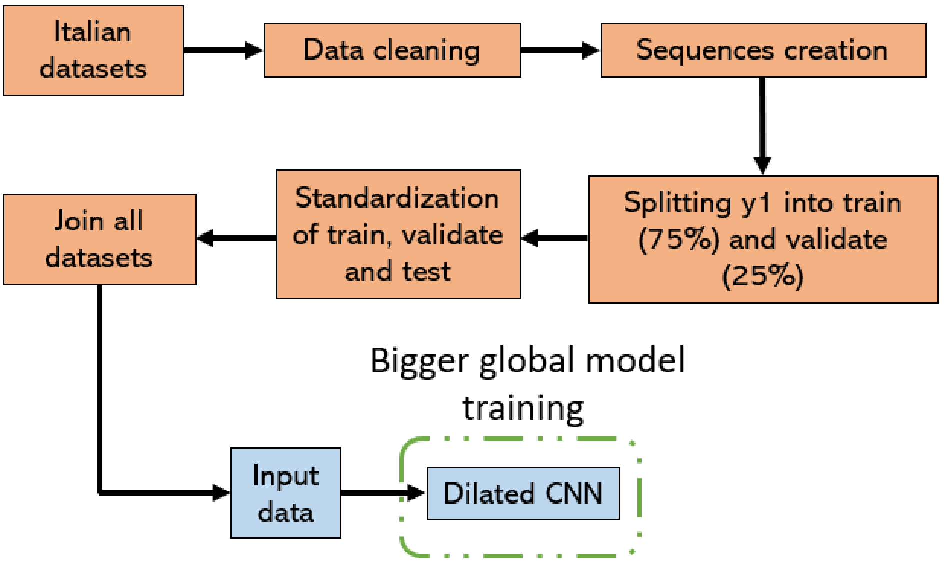

Italian datasets
Data cleaning
Sequences creation
Splitting y1 into train (75%) and validate (25%)
Standardization of train, validate and test
Join all datasets
Bigger global model training
Input data
Dilated CNN

Figure 17. Bigger global model flowchart.

4.1.5 Additional experiments

Other experiments were considered to test and compare their performance with the results obtained by the proposed experimental setups. Firstly, an additional binary input of the type of day is introduced in the dilated CNN model. This input connects to the 24 hours that correspond to the time horizon of prediction. For instance, the sequences take the previous 168 hours that might correspond to the hours from a Sunday at 00:00 to Saturday at 23:00, thus, the prediction will be made for the following Sunday. Then, there will be an input of 24 ones as it is a weekend day. After the model is trained, this extra input is added to refine the prediction on the 24-h output.

Secondly, instead of using the rainfall occurrence only, it is wanted to know the effect of the amount of rainfall of the last 48, 72, and 96 hours on the water demand. A third experiment is similar to the binary index of the type of day. An extra variable with the information of the corresponding month of the year is proposed to see the effect that the month has on the water demand.

These additional approaches were tested and compared with the results proposed in the previous experiments. Although positive results were obtained, they are not displayed in this report as they did not outperform the results achieved by the methodologies in 4.1.1 and 4.1.2.

4.2 Models

This section contains in detail, the architecture of the LSTM, simple CNN, and the dilated causal CNN models together with their hyperparameters. There was initial research on the architectures used for time series forecasting followed by an iterative process to find the models with better performances. This process involves the search of hyperparameters such as the number of units, layers, activation functions, learning rates, etc. Once the “best” hyperparameters were found, the three models were set. As the training of deep neural networks is computationally expensive, the GPUs of Google Colaboratory were employed. This online platform is a reliable tool to accelerate the training processes.

In general, the three models were trained using the ADAM optimization algorithm, and MAE as the loss function. Additionally, the *EarlyStopping( )* function was used to reduce the chance of overfitting during training. This technique will stop the process of training once the loss stops decreasing after 6 consecutive epochs. The training was made with batches of 32 samples, the maximum number of epochs was set at 200, however, the early stopping did not let the model run them all. In Addition, the *ReduceLROnPlateau* technique was used to improve the learning process. This function reduces the learning rate at a factor of 0.5 when the loss does not reduce over 6 epochs in this case. For instance, if the learning rate is equal to 0.01, it will be reduced to 0.005.

4.2.1 LSTM Model

This type of RNN was chosen due to its advantages of processing sequential data and, overcoming the short-term memory problem that occurs when employing conventional RNNs as mentioned in section 2.1.3. Regular RNNs tend to forget information about the events that happened a while ago as they get new information *(vanishing gradient problem).*

The LSTM architecture is presented as follows:

- Input + LSTM(64, tanh) + LSTM(64, tanh) + Dropout(0.20) + FC(64, ReLU) + FC(24, linear)

Apart from early stopping, a Dropout layer was added to the model after the second LSTM layer to reduce overfitting. In principle, it is possible to randomly “turn-of” a percentage of the neurons during training. Thus, the layer is forced to learn the same concept with different neurons resulting in better generalization. The figure below displays the flowchart of the LSTM model.

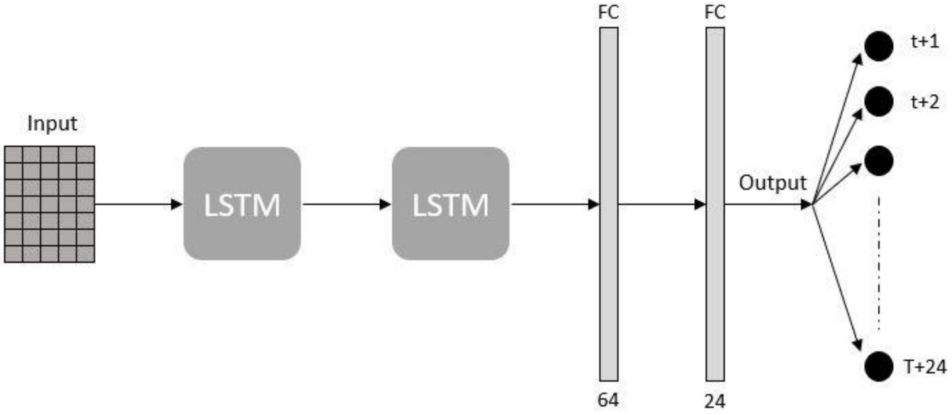


Figure 18. LSTM architecture.

4.2.2 Dilated Causal Convolution

As mentioned in section 2.1.2, The idea of dilated convolutions is to increase exponentially the receptive field and at the same time adding causality to the model. The literature names these convolutions as dilated temporal convolutions or dilated TCNs (Lea et al., 2017). To build a TCN, two aspects need to be fulfilled: 1) the length of the output feature map must be the same as the length of the input after convolving, and 2) causality must be considered. This means that no data from future timesteps are fed into the network. (Bai et al., 2018; Lea et al., 2017; Shen et al., 2020). To accomplish both requirements a number $k - 1$ (k is the filter size) zeros are padded to the left side. Moreover, TCNs are built with residual temporal blocks (Bai et al., 2018; Wan et al., 2019; Y. Wang & Liu, n.d.) with a defined number of convolutional layers. The filter size is 2 for the proposed architecture, and it is kept for all blocks. To increase the receptive field, the dilation factor (d) is doubled for every block. The blocks for the proposed architecture are composed of a convolutional layer with 32 filters of size equals 2, a *batch normalization* layer that applies a normalization that re-centers the output of the convolutional layers. Next, a ReLU transformation to add non-linearity to the data, and a dropout layer for regularization. In addition, a residual connection is added with a 1x1 convolution to improve the propagation of the gradient thought the entire network (Oord et al., 2016). The residual connection also helps the length of the output keep the same length as the input (Bai et al., 2018). Appendix E. Figure 1 shows the dilated CNN diagram.

The following figure represents the proposed residual temporal block.

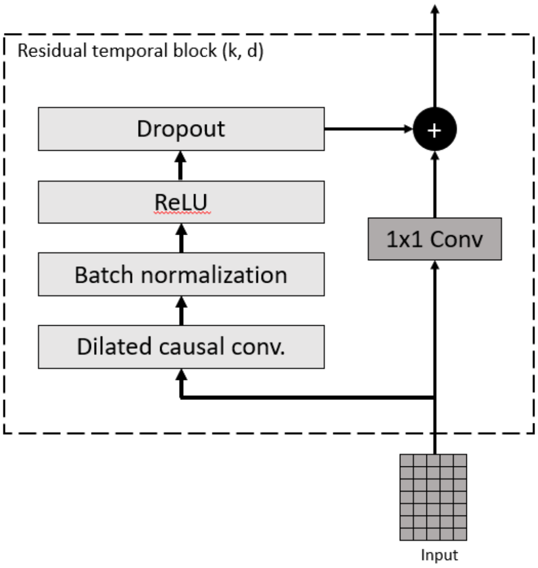


Figure 19. Residual temporal block.

Five of these blocks were stacked to build a deeper network increasing the receptive field. The final block would have a dilation factor equal to 16. It was found that without adding more blocks, the network can perform well. The output of the last block goes through a fully connected layer to generate the predictions (see Figure 20).

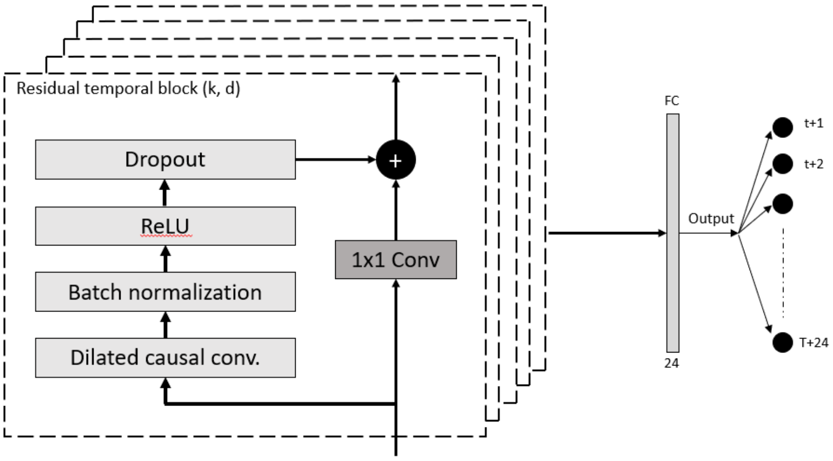


Figure 20. Dilated CNN architecture.

4.2.3 Simple CNN

Additionally, in order to compare the LSTM and dilated CNN models, a simple CNN was built. It does not have dropout layers for regularization. Also, it is composed of eight convolutional layers with filter size $k = 12$. The output of the convolution operations goes through two FC layers of 64, and 24 units, respectively.

The Simple CNN architecture is presented as follows:

- Input + 1D-Conv(8, $k = 12$, ReLU) + FC(64, ReLU) + FC(24, linear)

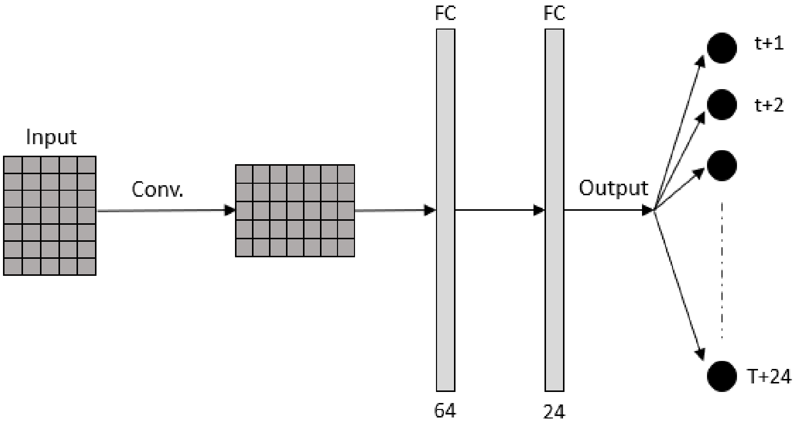


Figure 21. Simple CNN architecture.

It is seen that the number of trainable parameters is much lower compared to the Dilated CNN model. However, if more convolutional layers were added to the simple model, the number of trainable parameters would increase dramatically making the training process slower and inefficient. Thus, the advantages of the dilated CNN are highlighted as it is possible to build deeper networks without increasing too much the number of trainable parameters.

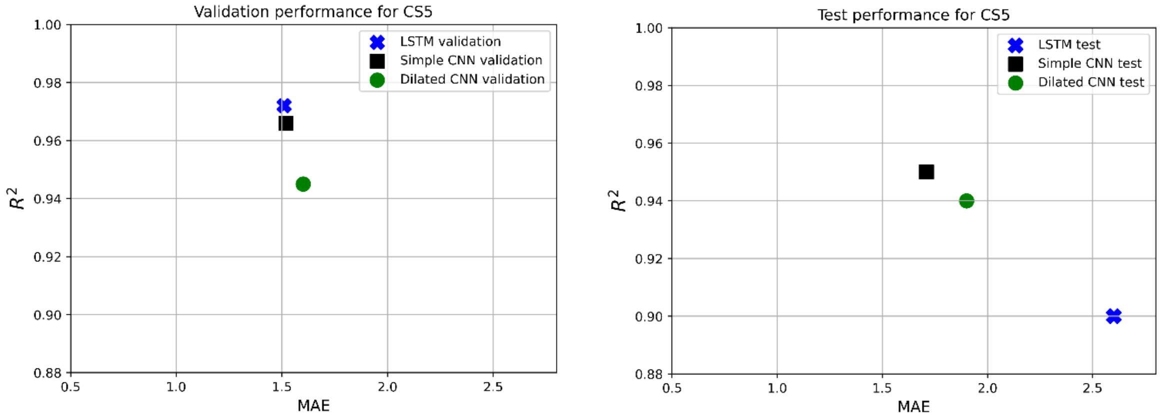
5 Results and Discussion

This chapter presents the results of the proposed experimental setups mentioned in section 4. A comparison with the benchmark algorithms is discussed together with the performance of the algorithms used. Furthermore, every time a model is run, the results will differ slightly. Therefore, the results to be presented are the average values after running five times every combination presented in Table 4. The order of the results is presented according to the research questions mentioned in section 1.2.

5.1.1.1 Research question 1

Can 1D-CNNs outperform RNNs and other existing algorithms for water demand forecasting?

The performance of the models proposed in section 4.2 is compared in terms of the coefficient of determination (R^2) and MAE for all seven case studies. The following graph shows the measures of performance using water demand data on CS5 without the addition of extra variables. The graphs of the other case studies are presented in appendix B.



a) b)

Figure 22. Performance of DL models on CS5. a) Validation, b) Test.

The figures above display results for the validation and test phases corresponding to data from year 1 and year 2 respectively. Overall, it is observed that the three models perform similarly on the validation phase, and it is true for the other case studies. For all case studies, R^2 ranges between 0.94 and 0.98. Also, MAE values present a comparable behavior. There are, however, two trends. For CS1, CS4, CS5, and CS7, where the average water demand is less than 100 L/s, the difference between MAE values is small compared to the other case studies. This behavior is expected as MAE results are tied to the distribution network sizes. The results of the test stage show that CNN models perform better than the LSTM. Although the LSTM model shows high performance for validation, when testing on unseen data, it does not generalize well. In all case studies, the LSTM model performed worse. On the other hand, both CNN models have a similar performance. For all case studies the difference in R^2 and

MAE is smaller compared to the validation stage. This shows that CNNs outperform RNNs for water demand forecasting. Moreover, Bai et al., (2018) found that not only the accuracy of CNNs is higher than RNNs but also the memory retention when predicting on different sequence modeling tasks.

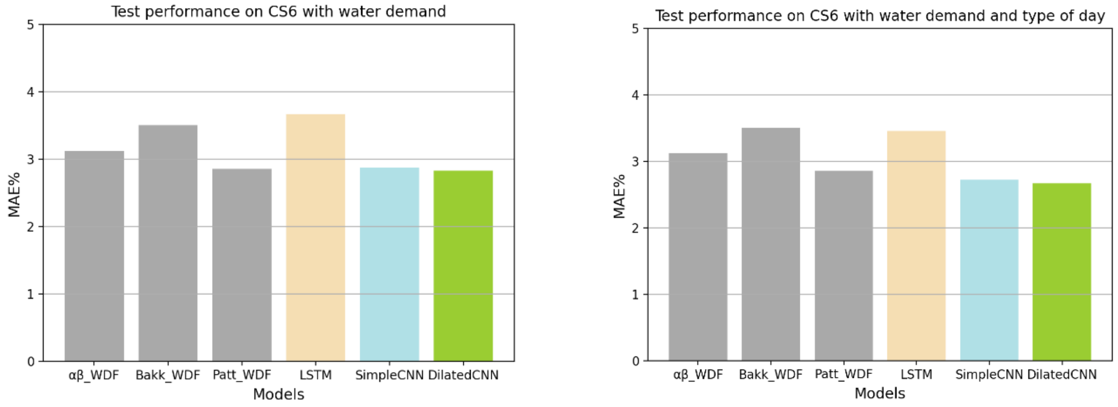
The rest of the results are based MAE% (See equation 12). This measure is proposed to compare to compare the performance of CNNs and RNNs with benchmark algorithms.

Next, to fully answer the first research question, the results of three benchmark algorithms are presented in Table 5. Patt_WDF (Alvisi et al., 2007) corresponds to a pattern-based model that forecasts hourly water demand. The Bakk_WDF (Bakker et al., 2013) and αβ_WDF (Pacchin et al., 2017) models forecast hourly water demands using time window techniques.

Table 5. Results in terms of MAE% of benchmark algorithms (Pacchin et al., 2019)

$MAE\%_{y2}$	CS1	CS2	CS3	CS4	CS5	CS6	CS7
αβ_WDF	**5.8**	3.724	4.091	4.001	**3.090**	3.122	**16.387**
Bakk_WDF	7.4	3.868	4.483	4.678	3.288	3.501	26.046
Patt_WDF	6.5	**3.561**	**3.551**	**3.444**	3.823	**2.857**	17.373

The following figure will show the performance of the benchmark, and the three models in terms of MAE% for year 2 (test phase), and all case studies. On the left, the figure depicts the performance only using water demand data whereas, on the right, the binary type of day index is included. The results obtained by Pacchin et al., (2019) have included this extra variable on their models. Figure 23 displays in light grey color the target results and in light yellow, light blue, and green, the results obtained on CS6.



a) b)

Figure 23. Test performance and comparison with benchmark models for CS6. a) Water demand only, b) Water demand and type of day.

First, in general, the LSTM model performed worse than the CNNs. When looking at the benchmark results, the Patt_WDF model shows better results than αβ_WDF and Bakk_WDF in most case studies. The bar charts in a) show that CNNs can predict water demand at a comparable level to the benchmark models simply using water demand data. In b), the CNNs show lower errors than the Patt_WDF model. The following table shows the results for all case studies. The values highlighted in bold represent the lowest error in each case study.

Table 6. Performance of the proposed models in terms of MAE%.

	Water demand only			Water demand + day		
Case/Models	LSTM	Simple CNN	Dilated CNN	LSTM	Simple CNN	Dilated CNN
CS1	10.6	**5.86**	6.98	9.57	**5.69**	6.73
CS2	4.8	**3.23**	3.82	4.37	**3.09**	3.22
CS3	4.8	**3.64**	3.81	4.27	3.62	**3.52**
CS4	6	3.77	**3.67**	4.04	3.48	**3.35**
CS5	**2.77**	3.26	3.29	4.13	**2.77**	3.05
CS6	3.67	2.875	**2.83**	3.46	2.72	**2.67**
CS7	15.8	15.5	**15.01**	14.695	14.93	**14.585**

CS1 displays higher errors compared to the other case studies where the errors do not exceed 4% considering that is also a residential district. These higher values might be due to the difference in demand between 1998 and 2000 since there was an increase of 20% in water demand. Nonetheless, for CS1, the simple CNN and dilated CNN present differences in errors at 0.68% and 1.18 % respectively with the best benchmark model. Moreover, it is observed that the simple CNN and dilated CNN outperform αβ_WDF and Bakk_WDF models using only water demand data. At CS7, the errors given by the benchmark models are high (16% - 26%), and 14% to 16 % by the proposed models. This might indicate that demands for cases with high variations are not feasible to forecast with high accuracy. These extreme demands are due to the sudden variations during the summer. The dilated CNN presents the lowest errors at 15.01% using water demand only, and 14.58% after adding the extra variable. Additionally, when adding the binary index of the type of day, the three models performed better. The LSTM showed to improve considerably for most case studies. In addition, the dilated CNN has outperformed the benchmark algorithms for CS2, CS3, CS4, CS5, CS6, and CS7. Only at CS1, there is a lower error from the αβ_WDF model, but the difference with the dilated CNN is less than 1%.

As it is shown that CNNs perform better when forecasting hourly water demand, the following results are based on the dilated CNN only. However, in appendix C, the results of the performance of the LSTM and simple CNN were added too.

5.1.1.2 Research question 2

Does external data, such as meteorological data, improve the performance of water demand forecasting?

This part involves the addition of meteorological variables to the model to determine whether this improves accuracy. As shown in Table 4, the variables were added one by one to the water demand data to train the models. The following bar chart shows the performance of the dilated CNN model for CS3 as an example. Also, the error of the benchmark model is shown for comparison.

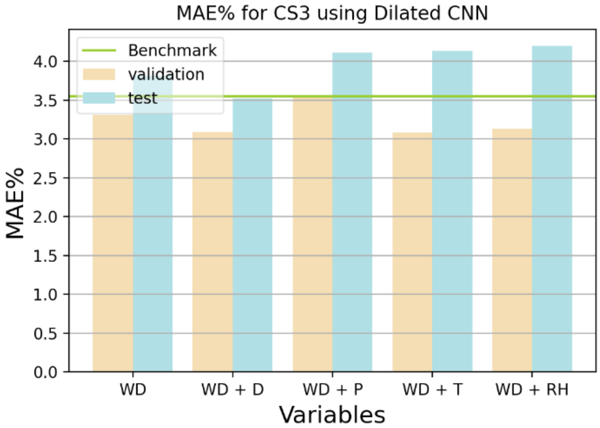


Figure 24. Performance of dilated CNN for CS3 after adding extra variables to the water demand individually.

The following table shows the errors obtained using the dilated CNN for all case studies starting by using water demand data only, and the addition of the extra variables. The values highlighted in bold color indicate the best performances. Appendix C.1 contains the performances for the LSTM and simple CNN models after adding a variable to the water demand data.

Table 7. Performance of dilated CNN after adding extra variables to the water demand individually.

Case/Variables	WD	WD + D	WD + P	WD + T	WD + RH
CS1	6.98	**6.73**	8.07		
CS2	3.82	**3.22**	3.66	3.48	3.42
CS3	3.81	**3.52**	4.11	4.13	4.2
CS4	3.67	**3.35**	3.81	3.9	4.02
CS5	3.29	**3.05**	3.17	3.62	4.33
CS6	2.83	**2.67**	2.77	3.155	2.87
CS7	15.01	**14.585**	14.7	15.44	15.7

Overall, the addition of meteorological variables enhances the performance for year 1 (validation). The mean percentage values are lower when adding extra variables than when using water demand data only (see appendix C.1). However, this is not reflected in year 2. Most cases show that meteorological data worsen the predictions on unseen data. This is an example of overfitting. For CS3, CS4, and CS5 the mean percentage values are increased by the temperature, relative humidity, and rainfall occurrence going to above 4%. As for the LSTM and simple CNN models, the results given in appendix C.1, show that the LSTM model has a similar response by the addition of meteorological variables. For year 1, the model increases its performance, but when testing on unseen data (year 2), it performs poorly compared to other algorithms. For CS7, the best performances are given with the binary variables showing an improvement of 1.6% and 1.7% respectively. The simple CNN, in contrast, performed worse in year 1. When adding any additional variable, the model raises the mean percentage values. In year 2, it is seen that outperforms the other algorithms.

Next, Appendix C.2 contains the results of the three models when adding two variables to the water demand data. When adding meteorological data, the performance drops further when testing on unseen data. However, when one of these climatologic variables is combined with the type of day variable, the errors are lower compared to using weather data. On the other hand, the simple CNN cannot handle the addition of two or more variables. When using climatologic variables, the mean percentage values in year 2 are very high compared to the LSTM and dilated CNN algorithms. It was also seen that when training the model, it starts overfitting since the only regularization added is the *early stopping* function. For this reason, the simple CNN will not further be analyzed.

Next, when using three additional variables to the water demand data, the same behavior was seen. When using the binary index of the type of day combined with meteorological variables, the model presents low errors for year 2. In contrast, adding only climatological variables affect the performance negatively for all case studies (see appendix C.3). In general, the performance after adding 3 variables to the model does not vary much for year 1 whereas, in year 2, it decreases.

Below in light blue color, the performance of the dilated CNN after adding all meteorological variables and the binary index of the day type is presented. Additionally, the results of the benchmark algorithms together with the best results of the dilated CNN using the binary index of the day type are shown.

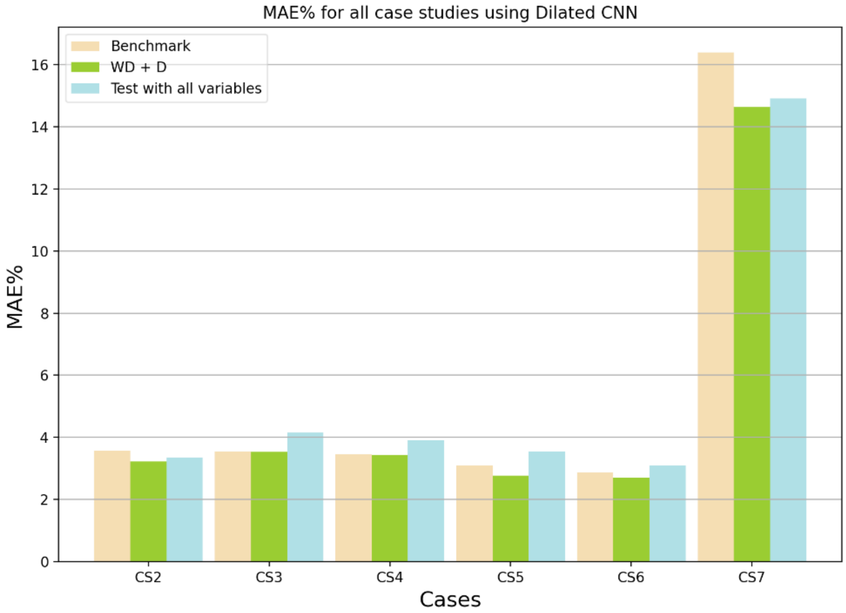


Figure 25. Performance of dilated CNN with all variables for CS2-CS7.

The addition of all variables together did not improve the forecast on unseen data. For CS2 and CS7, the model has outperformed the benchmark algorithms, but it is not as good as using the binary variable of the type of day. CS2, CS3, and CS4 showed a raise in performance compared to the model employing the three climatologic variables. This might be due to the generalization offered by the binary variables. Hourly, the binary variable will find whether a specific hour correspond to a weekday or a weekend day. Between 2014 and 2015, there is a shift of 24 hours, therefore, the model can learn these patterns to generalize better. On the other hand, meteorological variables such as temperature and relative humidity will not have the same values hourly from one year to the next one. Although there is a high correlation between these variables from 2014 to 2015, hourly, the values are not the same.

5.1.1.3 Research question 3

Can a general model be used to improve the predictions of water demand in individual cases?

This part contains the results after using a dataset composed of 5 years of water demand data. As described in section 4.1.2, a global model was built with the data of CS2, CS3, CS4, CS5, and CS6. This bigger dataset is believed to offer a better generalization of the patterns in the data.

The following bar chart displays the performance of benchmark algorithms, individual case studies CS2 to CS6 using water demand and the binary index of the day type, and the bigger dataset using water demand data only.

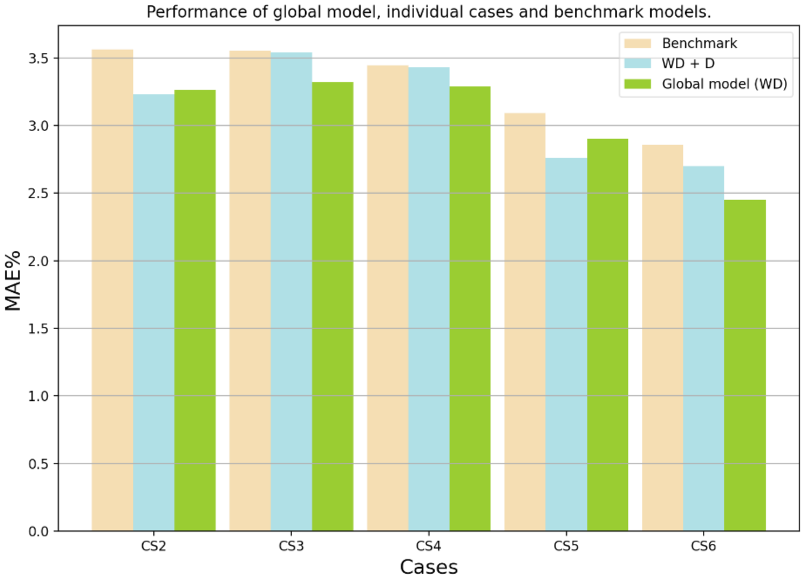


Figure 26. Performance of the dilated CNN on the global dataset using water demand only.

The figure above shows that the global model outperformed the benchmark models for the five case studies. Also, the performance on CS3, CS4, and CS6 is higher when using the global model. A bigger dataset indicated the advantage of having more data to be trained. Thus, better generalization was obtained. Although for CS2 and CS5 the global model did not improve the prediction compared to the individual dataset, the difference in errors is 0.03% and 0.14% respectively. Furthermore, considering that the binary input of the day type offered the best results on individual datasets, this insight was applied to the global dataset too. Figure 27 contains the performance of the global model after adding this variable compared to the previous results.

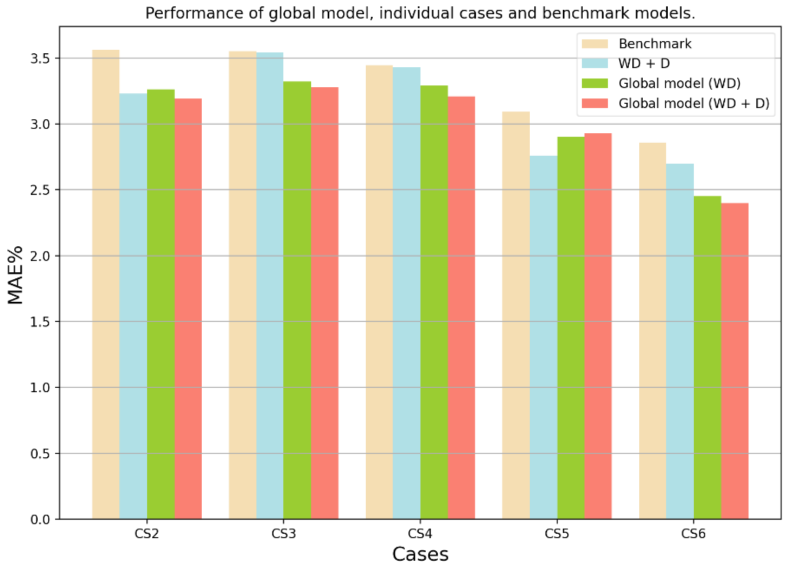


Figure 27. Performance of the dilated CNN on the global dataset using water demand and the binary index of the day type.

It is seen that after adding the binary variable of the type of day has a positive effect on the performance of the global model. Overall, a bigger dataset of the water demand, and the type of day makes the model more robust increasing generalization. The forecast on CS5, nonetheless, decrease a little compared to the cases using only water demand data, and the individual dataset. Yet, the mean percentage values are lower than the benchmark algorithms. Appendix C.4 contains the results of the global models using water demand data only, and the results adding the binary input.

A comparison of the prediction using the dilated CNN and LSTM models can be made together with the best global model. The following graphs on the left display the forecast for a 24-hour horizon, and on the right, the errors of the predictions for CS2. Particularly, the days shown below correspond to high, low, and average demand days. In appendix D contains the graphs for the other case studies.

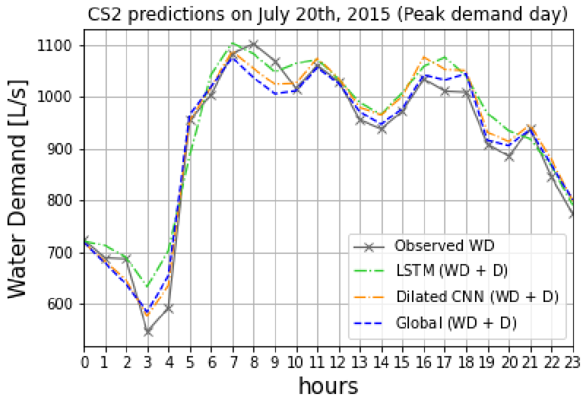


Figure 28. High peak demand predictions with the observed demand over a 24-hour horizon for CS2.

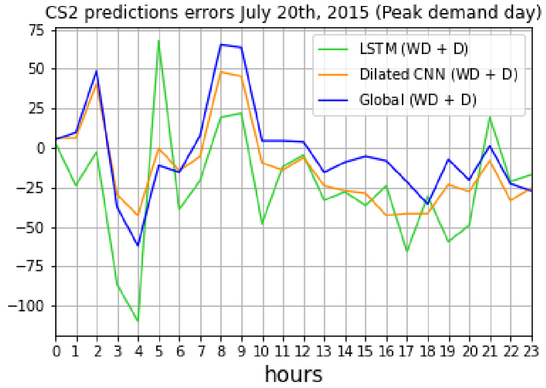


Figure 29. High peak demand errors with the observed water demand over a 24-hour horizon for CS2

Figure 28 shows that in general, the predictions of the models during a high peak in demand are accurate. However, the changes in demand are better generalized by the dilated CNN and the global model. Around 2 – 4 am, during the lowest consumption the LSTM model predicts around 100 L/s more water than the observed demand. This is seen in Figure 29 from the errors. Also, at 5 am, the LSTM underestimates the demand. However, the dilated CNN and the global model also underestimate the demand during the peak of the day. Next, there is the evening demand from 3 – 5 pm. The dilated CNN and the global models recognized these patterns better than the LSTM, and the errors are lower. Below, the predictions and errors for an average demand day are shown.

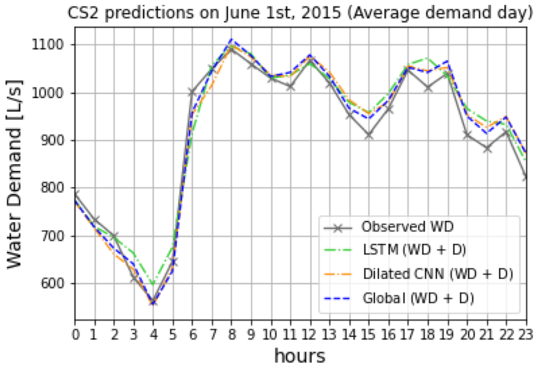


Figure 30. Average demand predictions with the observed water demand over a 24-hour horizon for CS2.

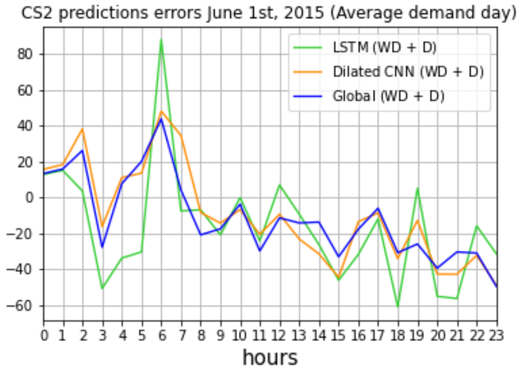


Figure 31. Average demand error with the observed water demand over a 24-hour horizon for CS2.

It is seen that the patterns are similar to the high peak demand. The LSTM presents the same behavior during the early morning. It overestimates and underestimates the demand. Along the day, there is a better generalization from the global and dilated CNN models as the errors are closer to 0. There is, nonetheless, a slight overestimation of water demand from the three models from 7 am to the rest of the day. The figures below show the predictions and errors for a low peak demand day.

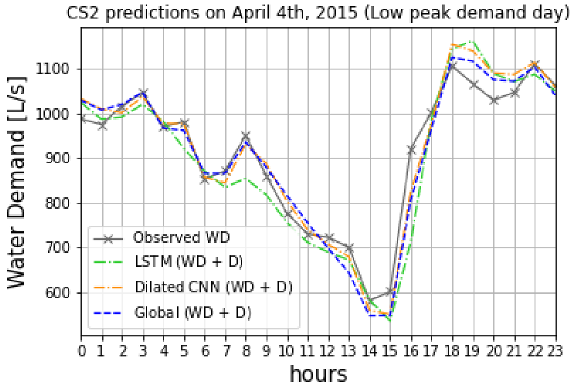


Figure 32. Low peak demand predictions with the observed water demand over a 24-hour horizon for CS2.

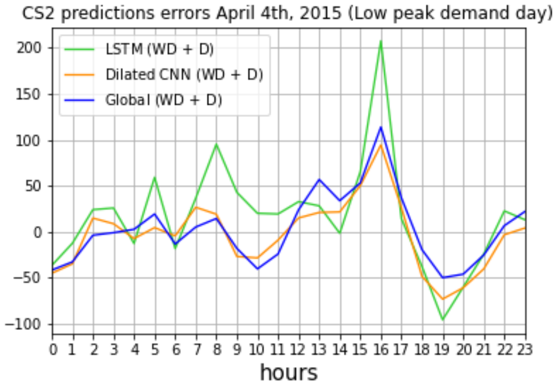


Figure 33. Low peak demand error with the observed water demand over a 24-hour horizon for CS2.

When there is low demand, the LSTM model performs poorly. It underestimates the demands during most of the day. The other two models show lower errors during the first 12 hours of the day. It is seen a sudden change in demand rising from 3 pm, that leads to higher errors from the three models.

5.1.1.4 Research question 4

Can transfer learning improve the performance of deep learning algorithms for water demand forecasting for cases in which the years and the geographical locations are different?

Transfer learning was applied to predict CS1 and CS7. The following graph shows the comparison of the best performance when using 0, 25, 50, and 75% of the data for training on the individual CS1 dataset in year 2 (black color), the performance using TL (green color), and the benchmark model.

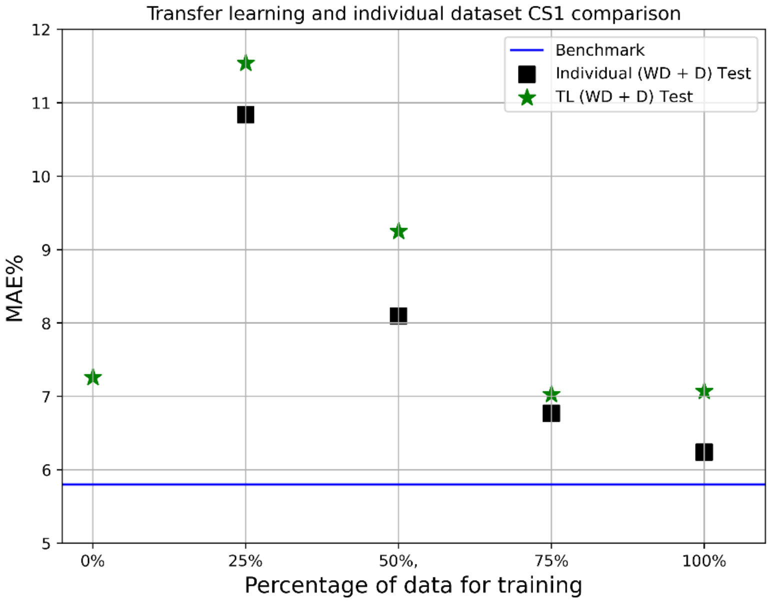


Figure 34. Performance of transfer learning on CS1.

The figure above shows the pre-trained model does not improve the performance of the prediction on CS1. It was reported before that none of the proposed DL models outperformed the best benchmark model (αβ_WDF). Nonetheless, the error difference between the individual dataset and the global one ranges from 0.25% using 75% of the data to 1.15% using half of the data for training. When using no data for training, the pre-trained model showed good performance in comparison to employing 25%, and 50% of data for training. This clearly shows that a pre-trained model can predict water demands with comparable accuracy to the benchmark models and the individual ones. Overall, the results suggest that TL must be employed carefully when fine-tuning the fully connected layers to achieve more accuracy.

Below, the same comparison is displayed for CS7.

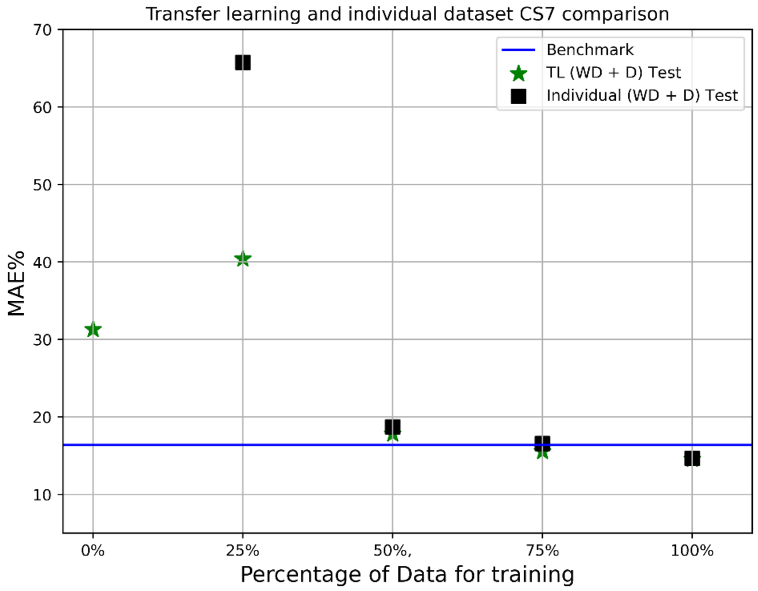


Figure 35. Performance of transfer learning on CS7.

The figure above shows that water demands for a tourist area cannot be modeled using little data. When training the model with 25% of the data, the model does not learn patterns well, and the errors are high. This is clearly due to the way the data was selected. Since the training data corresponds to the demands of January, February, and March, the average demands are very low compared to the demand in the summer months (June, July, August). Therefore, when the model is required to predict on unseen data, the patterns of the water demand during the summer are not learned by the network. However, the pre-trained model contains different water demand patterns. This is reflected by the accuracy of the prediction using 25% of the data. Although the performance is poor, it does improve concerning the predictions with the individual dataset. On the other hand, the pre-trained model outperforms the individual when using 50% and 75% of the data for training by a difference of 0.90% and 0.97%. Also, using only 75% of the data is enough for the pre-trained model to deliver higher performance than the benchmark models. Finally, when using 0% of data for training, the pre-trained model shows to perform better than when using 25% of data for training.

5.1.1.5 Research question 5

Can transfer learning be used to predict accurately the water demands of a district located in a different country?

This part includes the capabilities of using TL to predict the water demands of water distribution systems in a different country. The following graph shows the performance of TL employed in a Dutch WDS.

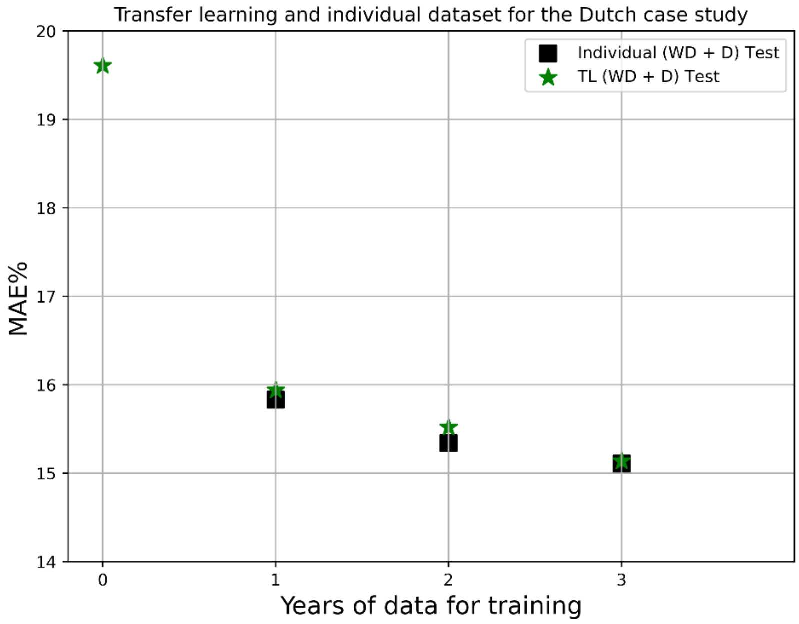


Figure 36. Performance of transfer learning on the Dutch WDS.

Figure 36 shows that there are comparable results when looking at the Italian case study. Both cases are distribution systems for touristic purposes. When using the pre-trained model with 0% of data for training, the predictions show lower errors compared to CS7. This shows that TL is useful for cases in which there is not data, for instance, a new project of development located in different regions. Since there is not a significant increase in the average water demand year by year, the errors after training with 1, 2, or 3 years of data are similar between 15% and 16%. The performance of TL is as accurate as using the individual models.

5.1.1.6 Research question 6

Can a bigger global model be trained to improve the performance of transfer learning when being employed to predict the water demands of a water distribution system located in another country?

As mentioned in section 4.1.4, a bigger global model was built using the 14 years of data of Italy. The model was trained to improve the accuracy of the predictions and make a comparison with the global model using 5 years of data.

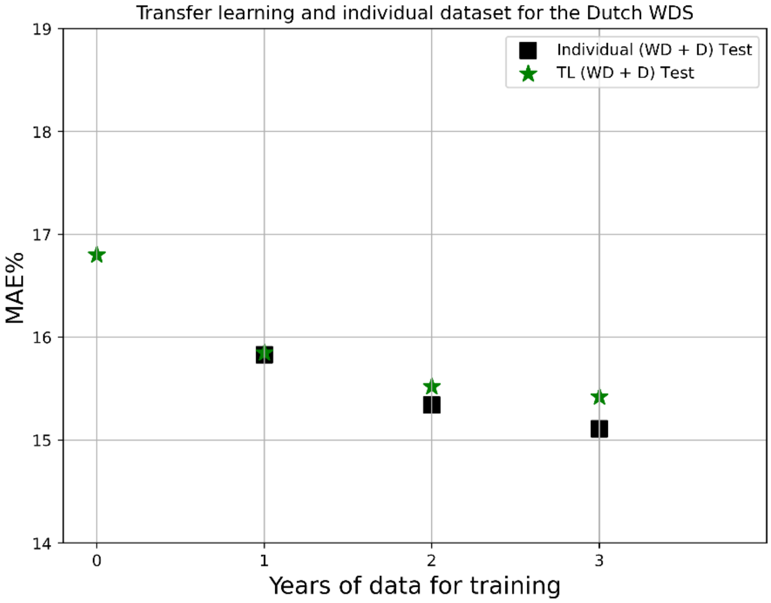


Figure 37. Performance of transfer learning on the Dutch WDS using 14 years of data for training.

The figure above displays that when using a bigger pre-trained model, the prediction using 0% of data for training shows lower errors compared to using the pre-trained model with less data (see Figure 36). Nonetheless, there was no improvement in the accuracy when re-training the last fully connected layers of the bigger global model. When using 2 and 3 years of data for training, there is a slight drop in performance compared to the previous results. This suggests that when applying TL, fine-tuning must be done carefully.

6 Conclusions

This section answers the research questions in section 1.2 by summarizing the results of the previous chapter.

1. *Can 1D-CNNs outperform RNNs and other existing algorithms for water demand forecasting?*

After an extensive process to find suitable hyperparameters and adequate architectures, three models were proposed to be compared with benchmark algorithms. Three DL models were built. A simple 1-dimensional convolutional neural network, a dilated causal convolutional neural network, and a long-short-term memory neural network. As the literature suggested for time series forecasting in other domains, 1D-CNNs outperform RNNs. With this idea in mind, it has been shown that 1D-CNNs perform better than RNNs on short-term water demand forecasting. It was seen that when using water demand data only, the performance of the CNNs was comparable to benchmark models. In addition, when adding extra information such as whether it is a weekday or a weekend day, the three models achieve higher performances, and especially the CNNs outperformed the existing models in 6 of the 7 cases studies. Nonetheless, there was less than a 1% of error difference between the best benchmark model and the 1D-CNNs models. Furthermore, there was an improvement in the forecast of the water demand under sudden changes by the rapid increase of users due to seasonal touristic activities.

2. *Does external data, such as meteorological data, improve the performance of water demand forecasting?*

The addition of meteorological variables was inspired by previous research showing that the amount of occurrence of rainfall improves the accuracy of water demand forecasting models. To address this question, it was decided to explore other variables such as relative humidity and maximum, minimum, and average temperature apart from the precipitation. The correlation of these variables with the water demand was calculated for 6 of the 7 case studies as the data of Castelfranco was not available for the years 1999, and 2000. It was found that the maximum temperature has a higher correlation with water demand followed by the relative humidity. Moreover, there is a higher negative correlation between the rainfall occurrence than the amount of rainfall with water demand. The results suggested that the addition of variables one by one increases the performance of the models at the validation stage. Nonetheless, when testing on unseen data, the three models decrease in performance. Then, when adding two, and three climatological variables to the models, the same behavior was observed. Overall, meteorological data did not improve the performance of short-term water demand forecasting. The addition of these types of variables showed to present overfitting in the three DL models when predicting on unseen data.

3. *Can a general model be used to improve the predictions of water demand in individual cases?*

A global dataset was built with data from 5 districts due to their closeness. These 5 WDS are managed by the same water utility company. Since it was shown that the dilated CNN outperformed the other two models, this architecture was employed and trained with the 5-year-dataset. Two experiments were carried out. First, the network was trained with water demand data only. It was shown that using a global model, the performances are higher compared to the benchmark models and individual models for most case studies. Second, the additional binary variable was added to this global model as it was shown that offers better generalization to the models. This approach showed to improve the accuracy of the global model further. Only 1 out of 5 case studies did not show improvement, however, the error is still lower than the best benchmark model. To sum up, a global model offers more generalization than the individual models even training with water demand data only.

4. *Can transfer learning improve the performance of deep learning algorithms for water demand forecasting for cases in which the years and the geographical locations are different?*

Transfer learning was applied to the other two case studies. The results suggested that the pre-trained model did not improve the forecast on one of the cases studies where there is an increase of the average water demand at 20% from 1998 to 2000. As this case study has a lack of data of one year, the model did not handle the change in the average water demands However, a pre-trained model showed a comparable performance when having no data for training. Additionally, when using different quantities of data for training, the results are also comparable to the benchmark models, and the individual ones.

On the other hand, TL has shown an improvement in the predictions for the case with touristic purposes. The results showed that a pre-trained model can have a higher accuracy when having no data for training compared to training only with 25% of data. Overall, the performance over the 2 cases, suggests that TL must be employed carefully and that fine-tuning require more attention.

5. *Can transfer learning be used to predict accurately the water demands of a district located in a different country?*

Additionally, TL was implemented to predict the water demand of a WDS in the north of the Netherlands. This WDS is mainly used for touristic purposes meaning that the demand patterns present sudden changes throughout the year. In General, TL showed to have similar performances with the individual models when using different quantities of data for training. When using a pre-trained model to forecast water demands with zero data for training, the errors are below 20%. Overall, the results indicate that a pre-trained global model with water demand data of another country can be used to predict water demands with less than 20% error regardless of the amount of data for training.

6. *Can a bigger global model be trained to improve the performance of transfer learning when being employed to predict the water demands of a water distribution system located in another country?*

Lastly, a bigger global model was built and trained with 14 years of data. Compared to the previous case, this bigger model showed an improvement in the prediction of 2.81% when using zero data for training. When re-training the model with 1 year of data, there was a slight decrease in the errors. However, when using 2 and 3 years of data to re-train the model, the errors increase a little compared to the previous results by applying the 5-year pre-trained model.

6.1 Recommendations and Future Work

Training DL models involve different processes that can be improved. Beginning, for instance, by adding more residual blocks to the dilated CNN architecture and finding better hyperparameters to improve the performance of the model. Also, a combination of RNNs and 1D-CNNs can be employed to exploit the capabilities of both types of networks.

The addition of meteorological variables can be further analyzed. As proposed in 2.1.4, the addition of variables in which the data of the prediction horizon is known such as the hourly temperature is encouraged. Normally, these predictions are accurate and can be used to refine the forecasts of the models. The same can be implemented for other climatologic variables (precipitation, relative humidity). Moreover, ranges of temperature can replace the exact values of hourly temperature to add generalization. These might be other approaches when processing meteorological data.

Transfer learning showed different performances for the case studies analyzed. This indicates that this method requires further development to exploit its potential uses. For the case of the Dilated CNN, not only re-training the fully connected layers but also adding convolutional layers or complete residual blocks might improve the performance of the model.

More research is always encouraged. Romero et al., (2021) used continuous kernel convolutions for sequential data (CKConv). The authors obtained state-of-art results when applying the techniques on multiples datasets. Results showed that CKConv outperformed RNNs, TCNs, and other algorithms.

References

Adamowski, J. F. (2008). Peak Daily Water Demand Forecast Modeling Using Artificial Neural Networks. *Journal of Water Resources Planning and Management*, *134*(2), 119–128. https://doi.org/10.1061/(ASCE)0733-9496(2008)134:2(119)

Alvisi, S., Franchini, M., & Marinelli, A. (2007). A short-term, pattern-based model for water-demand forecasting. *Journal of Hydroinformatics*, 12.

Anderson, D., & McNeil, G. (1992). *ARTIFICIAL NEURAL NETWORKS TECHNOLOGY* (A DACS State-of-the-Art Report). Rome Laboratory RL/C3C Griffiss AFB, NY 13441-5700. https://www.csiac.org/wp-content/uploads/2016/02/Artificial-Neural-Networks-Technology-SOAR.pdf

Antunes, A., Andrade-Campos, A., Sardinha-Lourenço, A., & Oliveira, M. S. (2018). Short-term water demand forecasting using machine learning techniques. *Journal of Hydroinformatics*, *20*(6), 1343–1366. https://doi.org/10.2166/hydro.2018.163

Bai, S., Kolter, J. Z., & Koltun, V. (2018). An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. *ArXiv:1803.01271 [Cs]*. http://arxiv.org/abs/1803.01271

Bakker, M., Vreeburg, J. H. G., van Schagen, K. M., & Rietveld, L. C. (2013). A fully adaptive forecasting model for short-term drinking water demand. *Environmental Modelling & Software*, *48*, 141–151. https://doi.org/10.1016/j.envsoft.2013.06.012

Batzner, K. (2019, February 28). *Convolutions in Autoregressive Neural Networks*. The Blog. https://theblog.github.io/post/convolution-in-autoregressive-neural-networks/

Chandradevan, R. (2017, August 18). *Radial Basis Functions Neural Networks—All we need to know*. Towards Data Science. https://towardsdatascience.com/radial-basis-functions-neural-networks-all-we-need-to-know-9a88cc053448

Deng, L. (2014). Deep Learning: Methods and Applications. *Foundations and Trends® in Signal Processing*, *7*(3–4), 197–387. https://doi.org/10.1561/2000000039

Donges, N. (2021, July 13). *What is transfer learning? Exploring the popular deep learning approach*. Builtin. https://builtin.com/data-science/transfer-learning

Donkor, E. A., Mazzuchi, T. A., Soyer, R., & Alan Roberson, J. (2014). Urban Water Demand Forecasting: Review of Methods and Models. *Journal of Water Resources Planning and Management*, *140*(2), 146–159. https://doi.org/10.1061/(ASCE)WR.1943-5452.0000314

Esen, Ö., Yıldırım, D. Ç., & Yıldırım, S. (2020). Threshold effects of economic growth on water stress in the Eurozone. *Environmental Science and Pollution Research*, *27*(25), 31427–31438. https://doi.org/10.1007/s11356-020-09383-y

Fu, Y., & Aldrich, C. (2018). Froth image analysis by use of transfer learning and convolutional neural networks. *Minerals Engineering*, *115*, 68–78. https://doi.org/10.1016/j.mineng.2017.10.005

Ghalehkhondabi, I., Ardjmand, E., Young, W. A., & Weckman, G. R. (2017). Water demand forecasting: Review of soft computing methods. *Environmental Monitoring and Assessment*, *189*(7), 313. https://doi.org/10.1007/s10661-017-6030-3

Güneralp, B., Güneralp, İ., & Liu, Y. (2015). Changing global patterns of urban exposure to flood and drought hazards. *Global Environmental Change*, *31*, 217–225. https://doi.org/10.1016/j.gloenvcha.2015.01.002

Guo, G., Liu, S., Wu, Y., Li, J., Zhou, R., & Zhu, X. (2018). Short-Term Water Demand Forecast Based on Deep Learning Method. *Journal of Water Resources Planning and Management*, *144*(12), 04018076. https://doi.org/10.1061/(ASCE)WR.1943-5452.0000992

Haidar, A., & Verma, B. (2018). Monthly Rainfall Forecasting Using One-Dimensional Deep Convolutional Neural Network. *IEEE Access*, *6*, 69053–69063. https://doi.org/10.1109/ACCESS.2018.2880044

Haque, M. M., Egodawatta, P., Rahman, A., & Goonetilleke, A. (2015). Assessing the significance of climate and community factors on urban water demand. *International Journal of Sustainable Built Environment*, *4*(2), 222–230. https://doi.org/10.1016/j.ijsbe.2015.11.001

Hewage, P., Behera, A., Trovati, M., Pereira, E., Ghahremani, M., Palmieri, F., & Liu, Y. (2020). Temporal convolutional neural (TCN) network for an effective weather forecasting using time-series data from the local weather station. *Soft Computing*, *24*(21), 16453–16482. https://doi.org/10.1007/s00500-020-04954-0

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8). https://doi.org/10.1162/neco.1997.9.8.1735

Hsu, K., Gupta, H. V., & Sorooshian, S. (1995). Artificial Neural Network Modeling of the Rainfall-Runoff Process. *Water Resources Research*, *31*(10), 2517–2530. https://doi.org/10.1029/95WR01955

Jain, A., Varshney, A. K., & Joshi, U. C. (n.d.). *Short-Term Water Demand Forecast Modelling at IIT Kanpur Using Artificial Neural Networks*. 23.

Kingma, D. P., & Ba, J. (2017). Adam: A Method for Stochastic Optimization. *ArXiv:1412.6980 [Cs]*. http://arxiv.org/abs/1412.6980

Kiranyaz, S., Avci, O., Abdeljaber, O., Ince, T., Gabbouj, M., & Inman, D. J. (2021). 1D convolutional neural networks and applications: A survey. *Mechanical Systems and Signal Processing*, *151*, 107398. https://doi.org/10.1016/j.ymssp.2020.107398

Kratzert, F., Klotz, D., Brenner, C., Schulz, K., & Herrnegger, M. (2018). Rainfall–runoff modelling using Long Short-Term Memory (LSTM) networks. *Hydrology and Earth System Sciences*, *22*(11), 6005–6022. https://doi.org/10.5194/hess-22-6005-2018

Lang, C., Steinborn, F., Steffens, O., & Lang, E. W. (2019). Electricity Load Forecasting—An Evaluation of Simple 1D-CNN Network Structures. *ArXiv:1911.11536 [Cs, Stat]*. http://arxiv.org/abs/1911.11536

Lara-Benítez, P., Carranza-García, M., Luna-Romera, J. M., & Riquelme, J. C. (2020). Temporal Convolutional Networks Applied to Energy-Related Time Series Forecasting. *Applied Sciences*, *10*(7), 2322. https://doi.org/10.3390/app10072322

Lea, C., Flynn, M. D., Vidal, R., Reiter, A., & Hager, G. D. (2017). Temporal Convolutional Networks for Action Segmentation and Detection. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1003–1012. https://doi.org/10.1109/CVPR.2017.113

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. https://doi.org/10.1038/nature14539

Lipton, Z. C., Berkowitz, J., & Elkan, C. (2015). A Critical Review of Recurrent Neural Networks for Sequence Learning. *ArXiv:1506.00019 [Cs]*. http://arxiv.org/abs/1506.00019

Moreno, J. J. M., Pol, A. P., & Gracia, P. M. (n.d.). *Artificial neural networks applied to forecasting time series*. 8.

Mu, L., Zheng, F., Tao, R., Zhang, Q., & Kapelan, Z. (2020). Hourly and Daily Urban Water Demand Predictions Using a Long Short-Term Memory Based Model. *Journal of Water Resources Planning and Management*, *146*(9), 05020017. https://doi.org/10.1061/(ASCE)WR.1943-5452.0001276

Navlani, A. (2019, December 9). *Neural Network Models in R*. Datacamp. https://www.datacamp.com/community/tutorials/neural-network-models-r

Olah, C. (2015, August 27). *Understanding LSTM Networks*. https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Oord, A. van den, Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., & Kavukcuoglu, K. (2016). WaveNet: A Generative Model for Raw Audio. *ArXiv:1609.03499 [Cs]*. http://arxiv.org/abs/1609.03499

O'Shea, K., & Nash, R. (2015). An Introduction to Convolutional Neural Networks. *ArXiv:1511.08458 [Cs]*. http://arxiv.org/abs/1511.08458

Pacchin, E., Alvisi, S., & Franchini, M. (2017). A Short-Term Water Demand Forecasting Model Using a Moving Window on Previously Observed Data. *Water*, *9*(3), 172. https://doi.org/10.3390/w9030172

Pacchin, E., Gagliardi, F., Alvisi, S., & Franchini, M. (2019). A Comparison of Short-Term Water Demand Forecasting Models. *Water Resources Management*, *33*(4), 1481–1497. https://doi.org/10.1007/s11269-019-02213-y

Phi, M. (2018, September 24). *Illustrated Guide to LSTM's and GRU's: A step by step explanation*. Towards Data Science. https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21

Romero, D. W., Kuzina, A., Bekkers, E. J., Tomczak, J. M., & Hoogendoorn, M. (2021). CKConv: Continuous Kernel Convolution For Sequential Data. *ArXiv:2102.02611 [Cs]*. http://arxiv.org/abs/2102.02611

Ruder, S. (2016, January 19). *An overview of gradient descent optimization algorithms*. Sebastian Ruder. https://ruder.io/optimizing-gradient-descent/

Shen, Z., Zhang, Y., Lu, J., Xu, J., & Xiao, G. (2020). A novel time series forecasting model with deep learning. *Neurocomputing*, *396*, 302–313. https://doi.org/10.1016/j.neucom.2018.12.084

Simplilearn. (2021, May 13). *The Best Introduction to Deep Learning—A Step by Step Guide*. Simplilearn. https://www.simplilearn.com/tutorials/deep-learning-tutorial/introduction-to-deep-learning

Tiwari, M. K., & Adamowski, J. F. (2015). Medium-Term Urban Water Demand Forecasting with Limited Data Using an Ensemble Wavelet–Bootstrap Machine-Learning Approach. *Journal of Water Resources Planning and Management*, *141*(2), 04014053. https://doi.org/10.1061/(ASCE)WR.1943-5452.0000454

Vandeput, N. (2019, July 5). *Forecast KPIs: RMSE, MAE, MAPE & Bias*. Towards Data Science. https://towardsdatascience.com/forecast-kpi-rmse-mae-mape-bias-cdc5703d242d

Villarin, M. C., & Rodriguez-Galiano, V. F. (2019). Machine Learning for Modeling Water Demand. *Journal of Water Resources Planning and Management*, *145*(5), 04019017. https://doi.org/10.1061/(ASCE)WR.1943-5452.0001067

Wan, R., Mei, S., Wang, J., Liu, M., & Yang, F. (2019). Multivariate Temporal Convolutional Network: A Deep Neural Networks Approach for Multivariate Time Series Forecasting. *Electronics*, *8*(8), 876. https://doi.org/10.3390/electronics8080876

Wang, X., Zhang, J., Shahid, S., Guan, E., Wu, Y., Gao, J., & He, R. (2016). Adaptation to climate change impacts on water demand. *Mitigation and Adaptation Strategies for Global Change*, *21*(1), 81–99. https://doi.org/10.1007/s11027-014-9571-6

Wang, Y., & Liu, Z. (n.d.). *Multivariate Time Series Prediction Based on Optimized Temporal Convolutional Networks with Stacked Auto-encoders*. 16.

Zebin, T., Sperrin, M., Peek, N., & Casson, A. J. (2018). Human activity recognition from inertial sensor time-series using batch normalized deep LSTM recurrent networks. *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC),* 1–4. https://doi.org/10.1109/EMBC.2018.8513115

Appendix A

The first gate is called the *forget gate layer* which looks at h_{t-1} and x_t. It passes through a sigmoid activation function which decides what information is kept and forgotten for each number in the cell state C_{t-1}. Formally, it is expressed as follows:

$$f_t = \sigma\left(w_f[h_{t-1}, x_t] + b_f\right) \tag{3}$$

Where w_f represents forget weights that are learned during training, b_f is the bias. After that, the new information which is going to be retained in the cell state passes through two gates. First, a layer with a sigmoid function called the *input gate layer* which tells the values to be updated. Then, the next gate layer contains a *tanh* function that generates a vector of new entrant values $\widetilde{C_t}$ to be added to the cell state. The following equations represent the *input gate layer* and the candidate memory cell respectively.

$$i_t = \sigma(w_i[h_{t-1}, x_t] + b_i) \tag{4}$$

$$\widetilde{C_t} = \tanh(w_c[h_{t-1}, x_t] + b_c) \tag{5}$$

Where w_i represents the input weights, w_c the candidate wrights, b_c and b_i the corresponding bias. Combining equations 4 and 5, the internal long-term memory or the subsequent cell memory is produced as follows:

$$c_t = f_t \circ c_t + i_t \circ \widetilde{C_t} \tag{6}$$

Where ◦ represents an element-wise multiplication.

Finally, the output gate can be generated. First, the output values of the cell state are filtered by a sigmoid layer. Next, the cell state goes through a *tanh* function and is multiplied by the output of the sigmoid gate. The hidden state output (h_t) is calculated as follows:

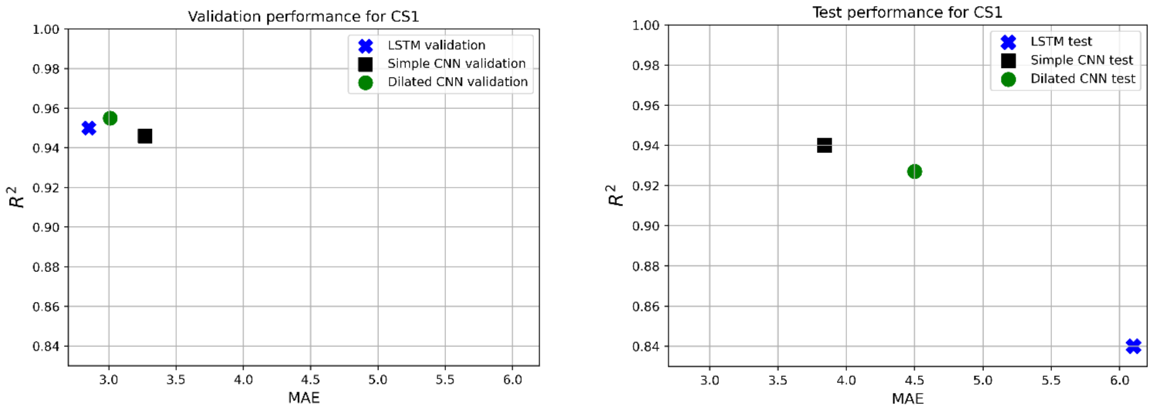
$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{7}$$

$$h_t = o_t \circ \tanh(c_t) \tag{8}$$

Where w_o signifies the output weights and b_o the bias term.

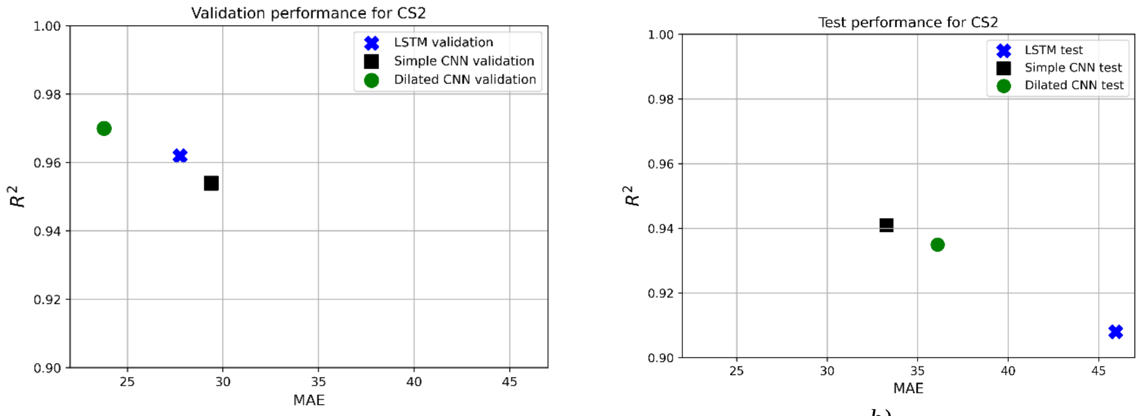
Appendix B

The following graphs display the results of the performance of the three models in terms of R^2 and MAE using water demand data only.



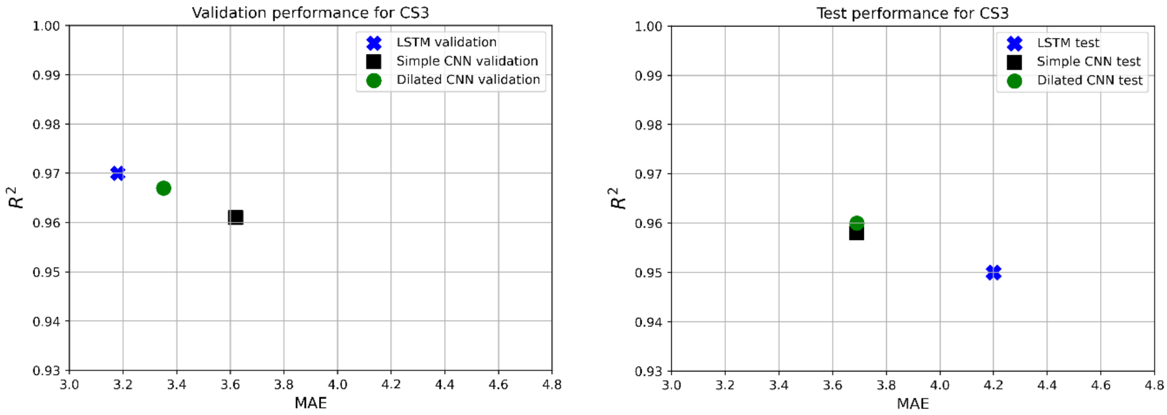
a) *b)*

B. Figure 1. Performance of DL models on CS1. a) Validation, b) Test.



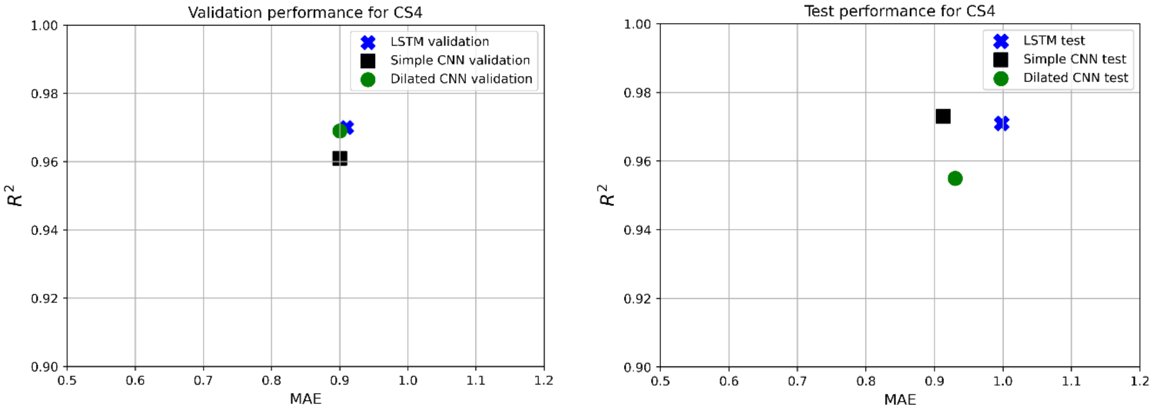
a) *b)*

B. Figure 2. Performance of DL models on CS2. a) Validation, b) Test.



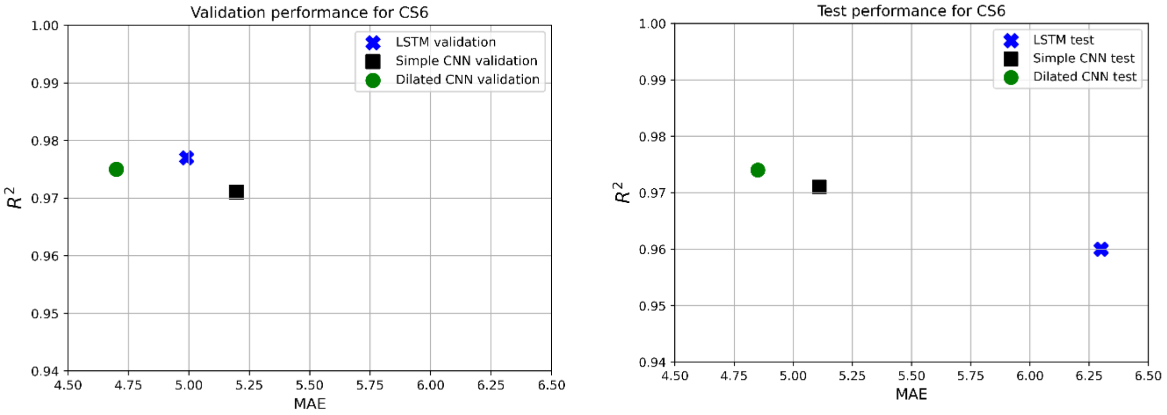
a) *b)*

B. Figure 3. Performance of DL models on CS3. a) Validation, b) Test.



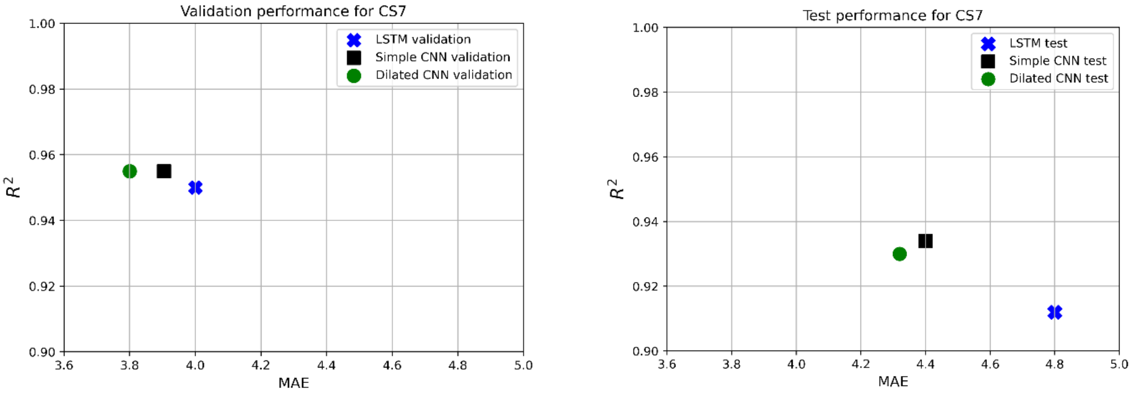
a) b)

B. Figure 4. Performance of DL models on CS4. a) Validation, b) Test.



a) b)

B. Figure 5. Performance of DL models on CS6. a) Validation, b) Test.



a) b)

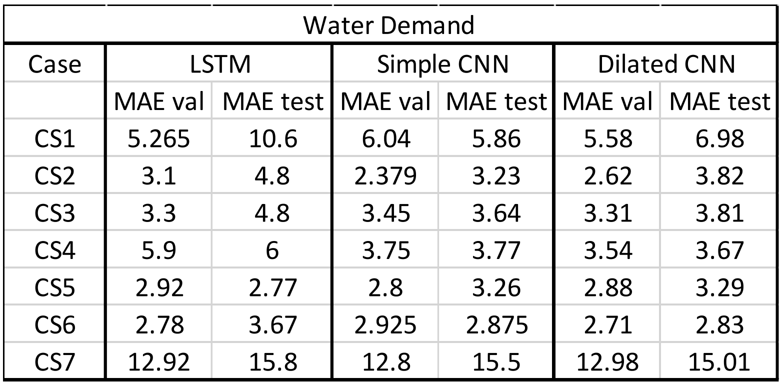
B. Figure 6 . Performance of DL models on CS7. a) Validation, b) Test.

Appendix C

Appendix C.1

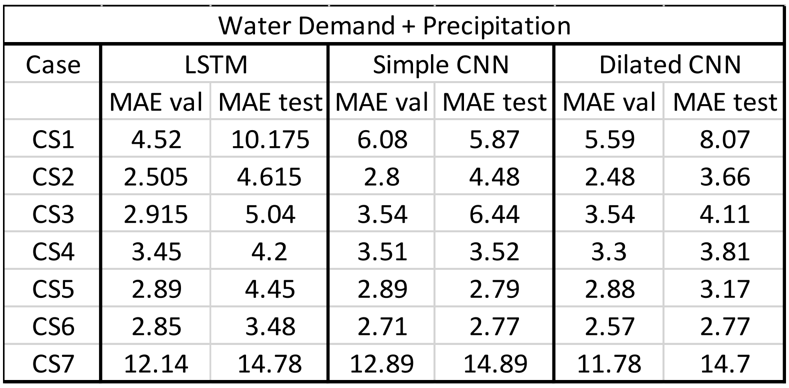
The following graphs correspond to the performance of the three models in terms of MAE% using a variable at a time.

C.1 Table 1. Performance of three models in terms of MAE% for water demand only



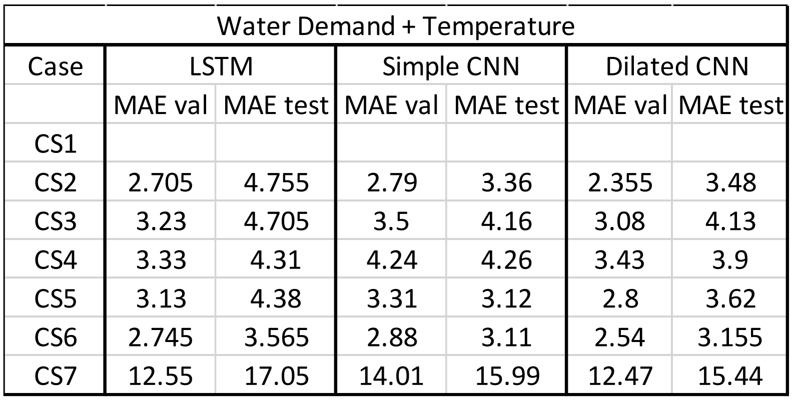
Water Demand						
Case	LSTM		Simple CNN		Dilated CNN	
	MAE val	MAE test	MAE val	MAE test	MAE val	MAE test
CS1	5.265	10.6	6.04	5.86	5.58	6.98
CS2	3.1	4.8	2.379	3.23	2.62	3.82
CS3	3.3	4.8	3.45	3.64	3.31	3.81
CS4	5.9	6	3.75	3.77	3.54	3.67
CS5	2.92	2.77	2.8	3.26	2.88	3.29
CS6	2.78	3.67	2.925	2.875	2.71	2.83
CS7	12.92	15.8	12.8	15.5	12.98	15.01

C.1 Table 2. Performance of three models in terms of MAE% for water demand and rainfall occurrence



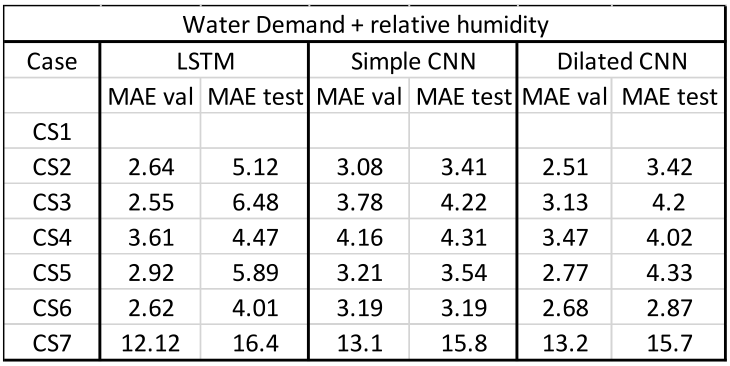
Water Demand + Precipitation						
Case	LSTM		Simple CNN		Dilated CNN	
	MAE val	MAE test	MAE val	MAE test	MAE val	MAE test
CS1	4.52	10.175	6.08	5.87	5.59	8.07
CS2	2.505	4.615	2.8	4.48	2.48	3.66
CS3	2.915	5.04	3.54	6.44	3.54	4.11
CS4	3.45	4.2	3.51	3.52	3.3	3.81
CS5	2.89	4.45	2.89	2.79	2.88	3.17
CS6	2.85	3.48	2.71	2.77	2.57	2.77
CS7	12.14	14.78	12.89	14.89	11.78	14.7

C.1 Table 3. Performance of three models in terms of MAE% for water demand and temperature



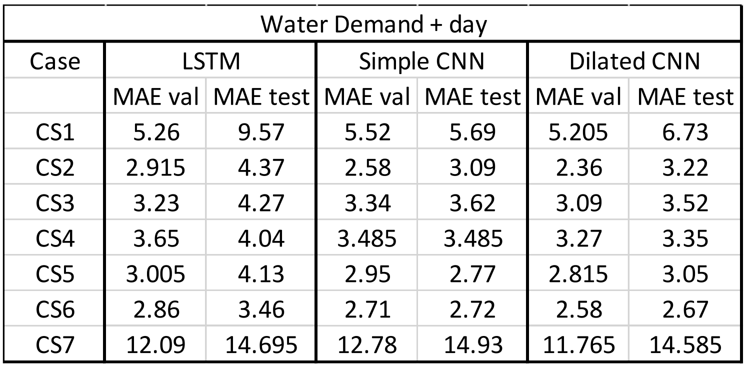
Water Demand + Temperature						
Case	LSTM		Simple CNN		Dilated CNN	
	MAE val	MAE test	MAE val	MAE test	MAE val	MAE test
CS1						
CS2	2.705	4.755	2.79	3.36	2.355	3.48
CS3	3.23	4.705	3.5	4.16	3.08	4.13
CS4	3.33	4.31	4.24	4.26	3.43	3.9
CS5	3.13	4.38	3.31	3.12	2.8	3.62
CS6	2.745	3.565	2.88	3.11	2.54	3.155
CS7	12.55	17.05	14.01	15.99	12.47	15.44

C.1 Table 4. Performance of three models in terms of MAE% for water demand and relative humidity



Water Demand + relative humidity						
Case	LSTM		Simple CNN		Dilated CNN	
	MAE val	MAE test	MAE val	MAE test	MAE val	MAE test
CS1						
CS2	2.64	5.12	3.08	3.41	2.51	3.42
CS3	2.55	6.48	3.78	4.22	3.13	4.2
CS4	3.61	4.47	4.16	4.31	3.47	4.02
CS5	2.92	5.89	3.21	3.54	2.77	4.33
CS6	2.62	4.01	3.19	3.19	2.68	2.87
CS7	12.12	16.4	13.1	15.8	13.2	15.7

C.1 Table 5. Performance of three models in terms of MAE% for water demand and binary input of the days

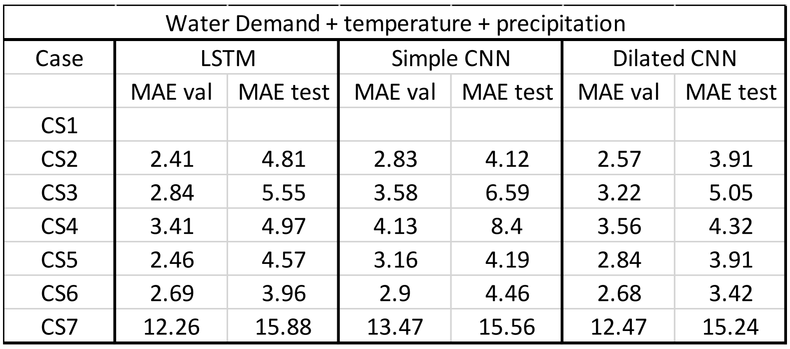


Water Demand + day						
Case	LSTM		Simple CNN		Dilated CNN	
	MAE val	MAE test	MAE val	MAE test	MAE val	MAE test
CS1	5.26	9.57	5.52	5.69	5.205	6.73
CS2	2.915	4.37	2.58	3.09	2.36	3.22
CS3	3.23	4.27	3.34	3.62	3.09	3.52
CS4	3.65	4.04	3.485	3.485	3.27	3.35
CS5	3.005	4.13	2.95	2.77	2.815	3.05
CS6	2.86	3.46	2.71	2.72	2.58	2.67
CS7	12.09	14.695	12.78	14.93	11.765	14.585

Appendix C.2

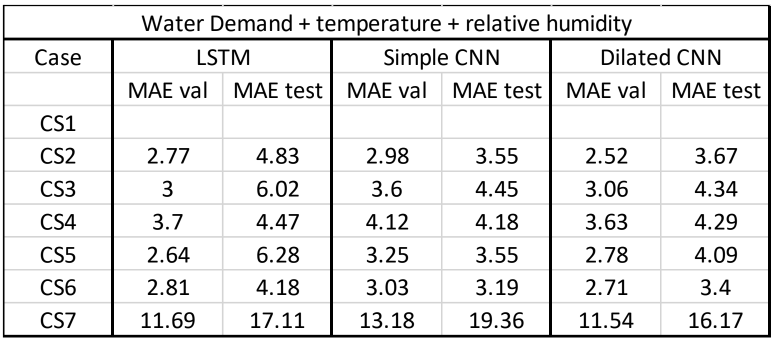
The following tables contain the performance in terms of MAE% using two extra variables.

C.2 Table 1. Performance of three models in terms of MAE% for water demand with temperature and rainfall occurrence.



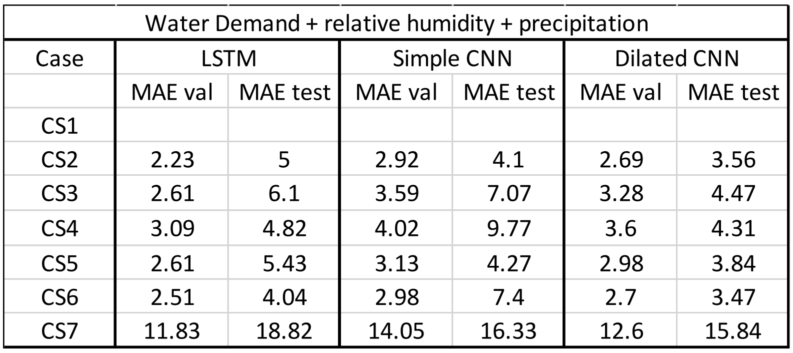
Water Demand + temperature + precipitation						
Case	LSTM		Simple CNN		Dilated CNN	
	MAE val	MAE test	MAE val	MAE test	MAE val	MAE test
CS1						
CS2	2.41	4.81	2.83	4.12	2.57	3.91
CS3	2.84	5.55	3.58	6.59	3.22	5.05
CS4	3.41	4.97	4.13	8.4	3.56	4.32
CS5	2.46	4.57	3.16	4.19	2.84	3.91
CS6	2.69	3.96	2.9	4.46	2.68	3.42
CS7	12.26	15.88	13.47	15.56	12.47	15.24

C.2 Table 2. Performance of three models in terms of MAE% for water demand with temperature and relative humidity.



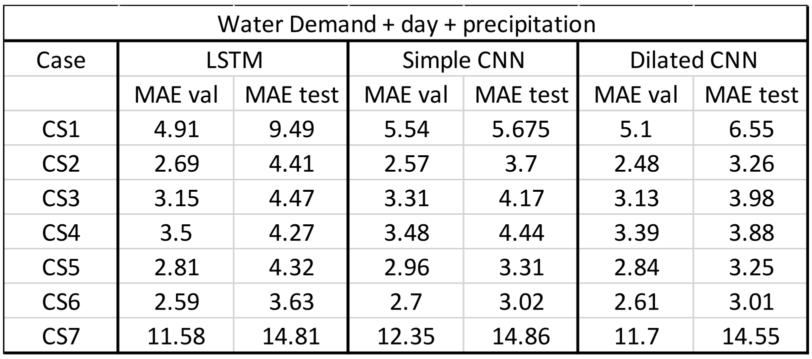
Water Demand + temperature + relative humidity						
Case	LSTM		Simple CNN		Dilated CNN	
	MAE val	MAE test	MAE val	MAE test	MAE val	MAE test
CS1						
CS2	2.77	4.83	2.98	3.55	2.52	3.67
CS3	3	6.02	3.6	4.45	3.06	4.34
CS4	3.7	4.47	4.12	4.18	3.63	4.29
CS5	2.64	6.28	3.25	3.55	2.78	4.09
CS6	2.81	4.18	3.03	3.19	2.71	3.4
CS7	11.69	17.11	13.18	19.36	11.54	16.17

C.2 Table 3. Performance of three models in terms of MAE% for water demand with temperature and rainfall occurrence.



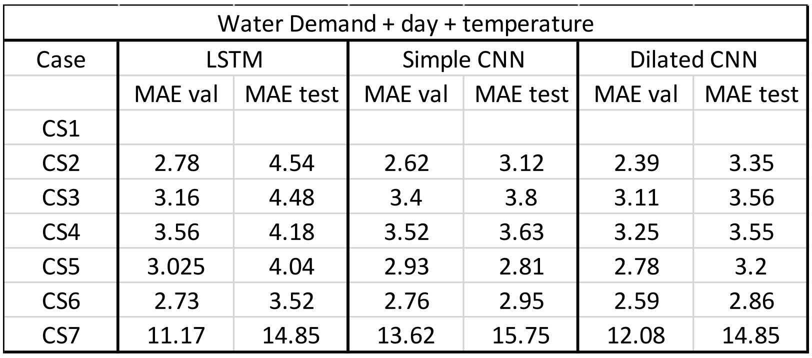
Water Demand + relative humidity + precipitation						
Case	LSTM		Simple CNN		Dilated CNN	
	MAE val	MAE test	MAE val	MAE test	MAE val	MAE test
CS1						
CS2	2.23	5	2.92	4.1	2.69	3.56
CS3	2.61	6.1	3.59	7.07	3.28	4.47
CS4	3.09	4.82	4.02	9.77	3.6	4.31
CS5	2.61	5.43	3.13	4.27	2.98	3.84
CS6	2.51	4.04	2.98	7.4	2.7	3.47
CS7	11.83	18.82	14.05	16.33	12.6	15.84

C.2 Table 4. Performance of three models in terms of MAE% for water demand with binary index of the day and rainfall occurrence.



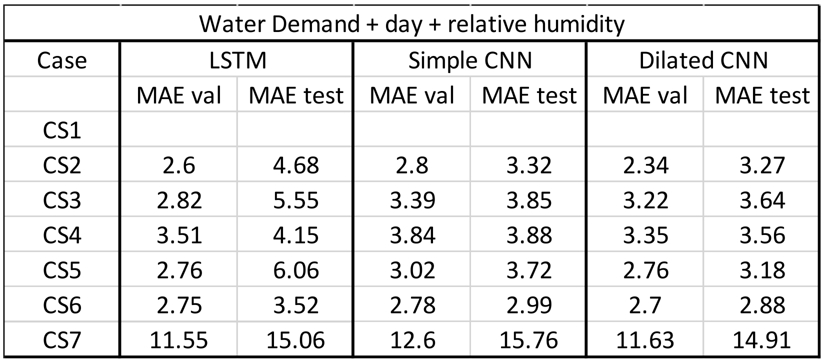
Water Demand + day + precipitation						
Case	LSTM		Simple CNN		Dilated CNN	
	MAE val	MAE test	MAE val	MAE test	MAE val	MAE test
CS1	4.91	9.49	5.54	5.675	5.1	6.55
CS2	2.69	4.41	2.57	3.7	2.48	3.26
CS3	3.15	4.47	3.31	4.17	3.13	3.98
CS4	3.5	4.27	3.48	4.44	3.39	3.88
CS5	2.81	4.32	2.96	3.31	2.84	3.25
CS6	2.59	3.63	2.7	3.02	2.61	3.01
CS7	11.58	14.81	12.35	14.86	11.7	14.55

C.2 Table 5. Performance of three models in terms of MAE% for water demand with binary index of the day and temperature.



Water Demand + day + temperature						
Case	LSTM		Simple CNN		Dilated CNN	
	MAE val	MAE test	MAE val	MAE test	MAE val	MAE test
CS1						
CS2	2.78	4.54	2.62	3.12	2.39	3.35
CS3	3.16	4.48	3.4	3.8	3.11	3.56
CS4	3.56	4.18	3.52	3.63	3.25	3.55
CS5	3.025	4.04	2.93	2.81	2.78	3.2
CS6	2.73	3.52	2.76	2.95	2.59	2.86
CS7	11.17	14.85	13.62	15.75	12.08	14.85

C.2 Table 6. Performance of three models in terms of MAE% for water demand with binary index of the day and relative humidity.

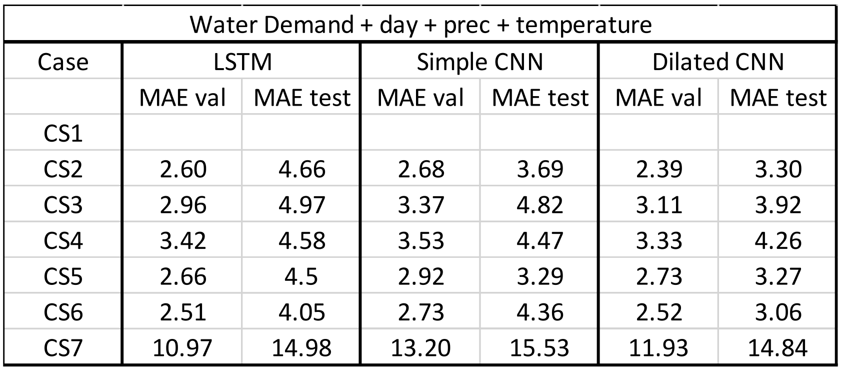


Water Demand + day + relative humidity						
Case	LSTM		Simple CNN		Dilated CNN	
	MAE val	MAE test	MAE val	MAE test	MAE val	MAE test
CS1						
CS2	2.6	4.68	2.8	3.32	2.34	3.27
CS3	2.82	5.55	3.39	3.85	3.22	3.64
CS4	3.51	4.15	3.84	3.88	3.35	3.56
CS5	2.76	6.06	3.02	3.72	2.76	3.18
CS6	2.75	3.52	2.78	2.99	2.7	2.88
CS7	11.55	15.06	12.6	15.76	11.63	14.91

Appendix C.3

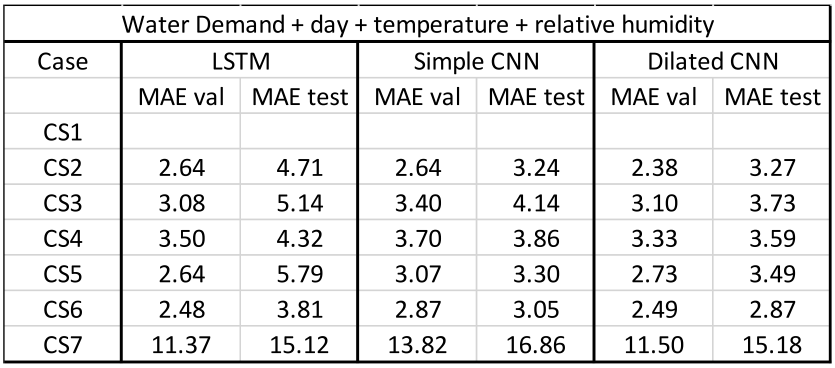
The following tables show the performance of the three models using three extra variables.

C.3 Table 1.Performance of three models in terms of MAE% for water demand with binary index of the day, rainfall occurrence, and temperature.



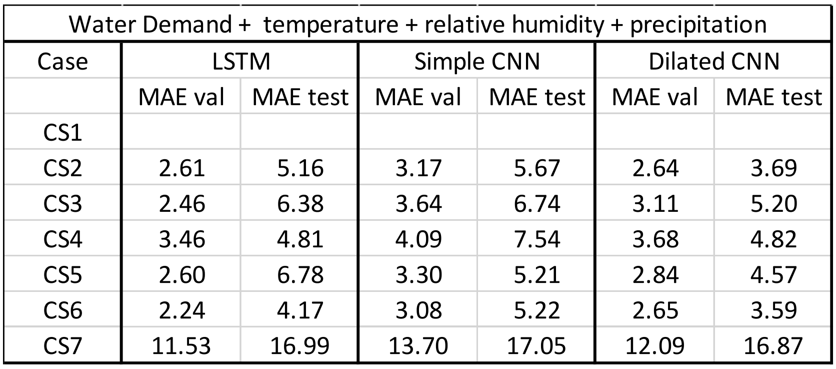
Water Demand + day + prec + temperature						
Case	LSTM		Simple CNN		Dilated CNN	
	MAE val	MAE test	MAE val	MAE test	MAE val	MAE test
CS1						
CS2	2.60	4.66	2.68	3.69	2.39	3.30
CS3	2.96	4.97	3.37	4.82	3.11	3.92
CS4	3.42	4.58	3.53	4.47	3.33	4.26
CS5	2.66	4.5	2.92	3.29	2.73	3.27
CS6	2.51	4.05	2.73	4.36	2.52	3.06
CS7	10.97	14.98	13.20	15.53	11.93	14.84

C.3 Table 2. Performance of three models in terms of MAE% for water demand with binary index of the day, relative humidity, and temperature.



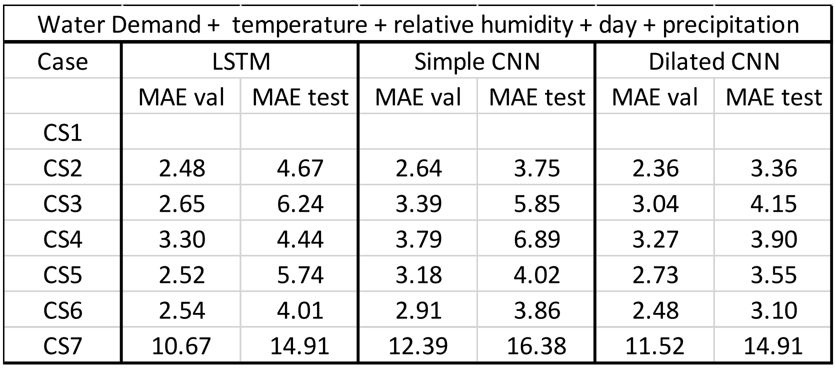
Water Demand + day + temperature + relative humidity						
Case	LSTM		Simple CNN		Dilated CNN	
	MAE val	MAE test	MAE val	MAE test	MAE val	MAE test
CS1						
CS2	2.64	4.71	2.64	3.24	2.38	3.27
CS3	3.08	5.14	3.40	4.14	3.10	3.73
CS4	3.50	4.32	3.70	3.86	3.33	3.59
CS5	2.64	5.79	3.07	3.30	2.73	3.49
CS6	2.48	3.81	2.87	3.05	2.49	2.87
CS7	11.37	15.12	13.82	16.86	11.50	15.18

C.3 Table 3. Performance of three models in terms of MAE% for water demand with temperature, relative humidity, and rainfall occurrence.



Water Demand + temperature + relative humidity + precipitation						
Case	LSTM		Simple CNN		Dilated CNN	
	MAE val	MAE test	MAE val	MAE test	MAE val	MAE test
CS1						
CS2	2.61	5.16	3.17	5.67	2.64	3.69
CS3	2.46	6.38	3.64	6.74	3.11	5.20
CS4	3.46	4.81	4.09	7.54	3.68	4.82
CS5	2.60	6.78	3.30	5.21	2.84	4.57
CS6	2.24	4.17	3.08	5.22	2.65	3.59
CS7	11.53	16.99	13.70	17.05	12.09	16.87

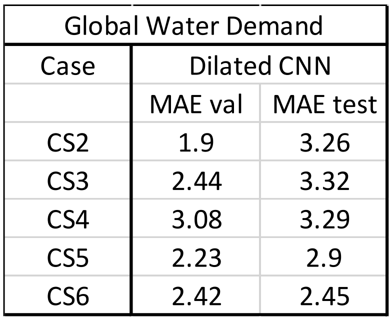
C.3 Table 4. Performance of three models in terms of MAE% using all variables.



Water Demand + temperature + relative humidity + day + precipitation						
Case	LSTM		Simple CNN		Dilated CNN	
	MAE val	MAE test	MAE val	MAE test	MAE val	MAE test
CS1						
CS2	2.48	4.67	2.64	3.75	2.36	3.36
CS3	2.65	6.24	3.39	5.85	3.04	4.15
CS4	3.30	4.44	3.79	6.89	3.27	3.90
CS5	2.52	5.74	3.18	4.02	2.73	3.55
CS6	2.54	4.01	2.91	3.86	2.48	3.10
CS7	10.67	14.91	12.39	16.38	11.52	14.91

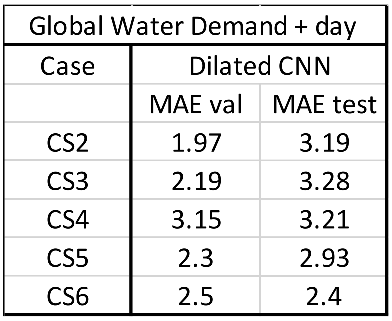
Appendix C.4

C.4 Table 1. Performance of the global model in terms of MAE% using water demand only.



Global Water Demand		
Case	Dilated CNN	
	MAE val	MAE test
CS2	1.9	3.26
CS3	2.44	3.32
CS4	3.08	3.29
CS5	2.23	2.9
CS6	2.42	2.45

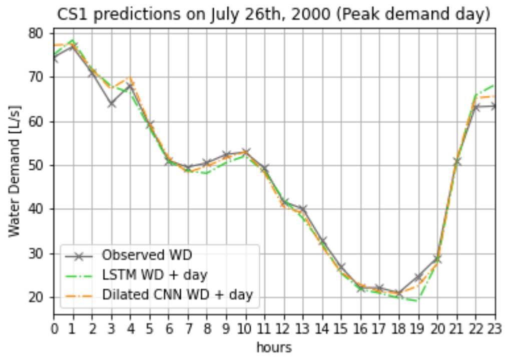
C.4 Table 2. Performance of the global model in terms of MAE% using water demand and binary index of the day type.



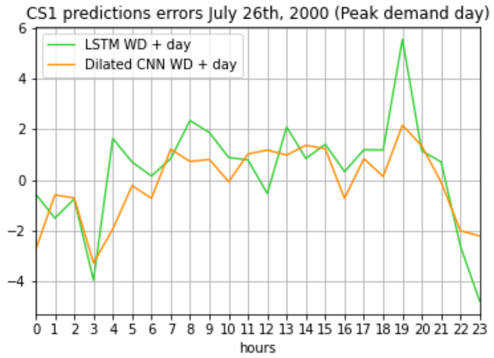
Global Water Demand + day		
Case	Dilated CNN	
	MAE val	MAE test
CS2	1.97	3.19
CS3	2.19	3.28
CS4	3.15	3.21
CS5	2.3	2.93
CS6	2.5	2.4

Appendix D

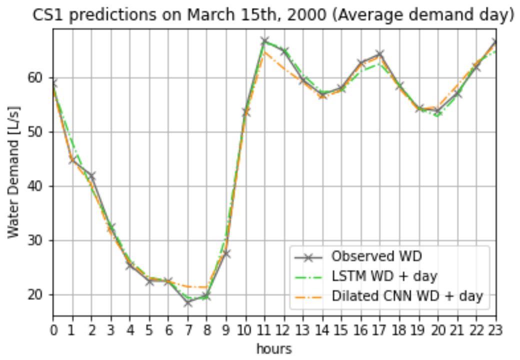
The following graphs correspond to the predictions for the high, low, and average water demands and their corresponding errors.



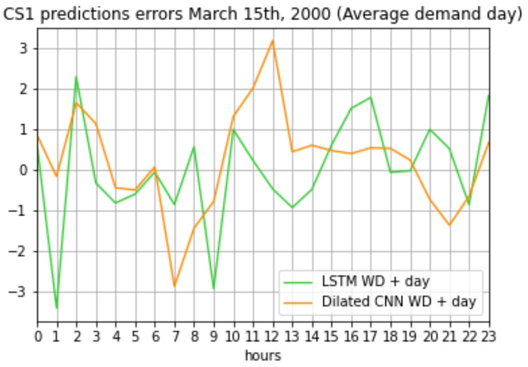
D. Figure 1. High peak demand predictions with the observed demand over a 24-hour horizon for CS1.



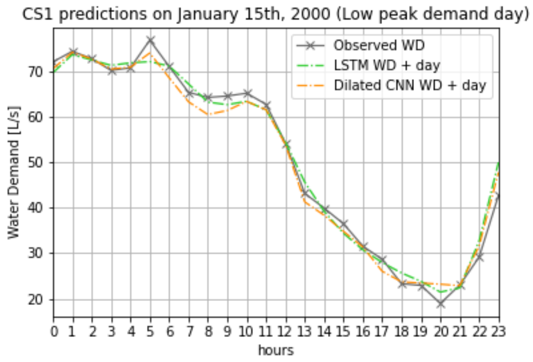
D. Figure 2. High peak demand errors with the observed water demand over a 24-hour horizon for CS1.



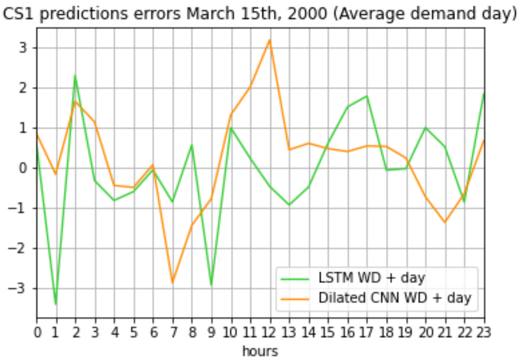
D. Figure 3. Average demand predictions with the observed water demand over a 24-hour horizon for CS1.



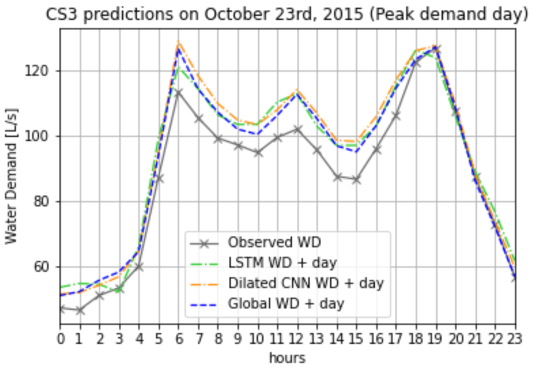
D. Figure 4. Average demand error with the observed water demand over a 24-hour horizon for CS1.



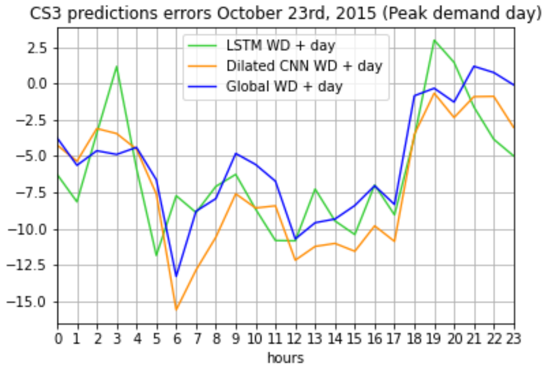
D. Figure 5. Low peak demand predictions with the observed water demand over a 24-hour horizon for CS1.



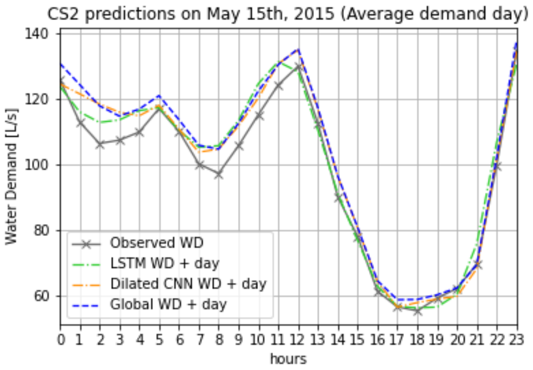
D. Figure 6. Low peak demand errors with the observed water demand over a 24-hour horizon for CS1.



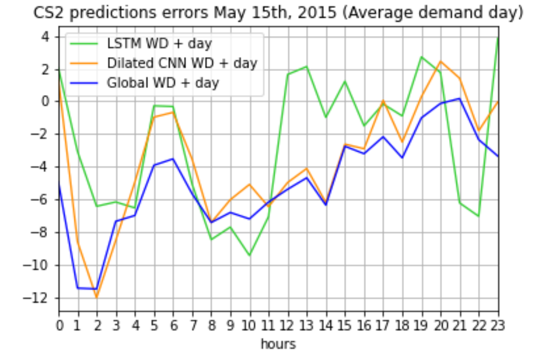
D. Figure 7. High peak demand predictions with the observed demand over a 24-hour horizon for CS3.



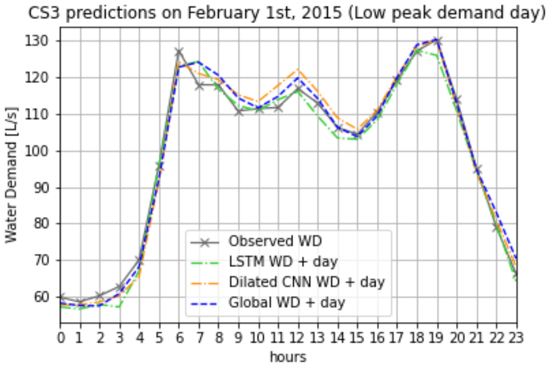
D. Figure 8. High peak demand errors with the observed water demand over a 24-hour horizon for CS3.



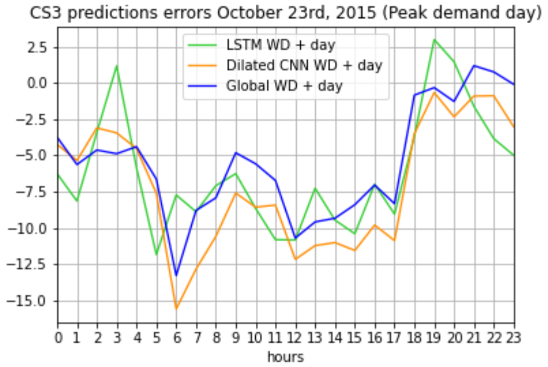
D. Figure 9. Average demand predictions with the observed water demand over a 24-hour horizon for CS3.



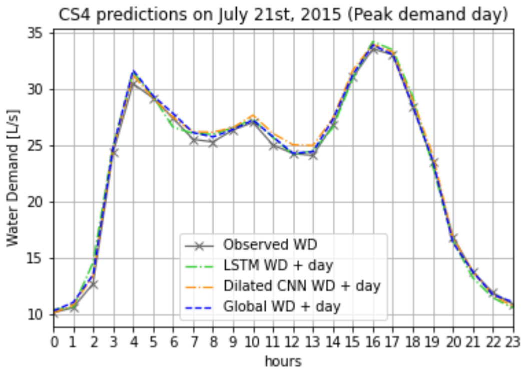
D. Figure 10. Average demand error with the observed water demand over a 24-hour horizon for CS3.



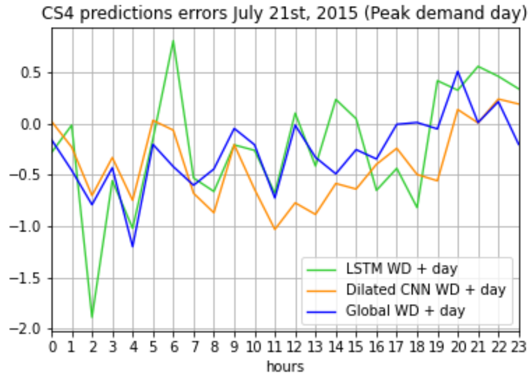
D. Figure 11. Low peak demand predictions with the observed water demand over a 24-hour horizon for CS3.



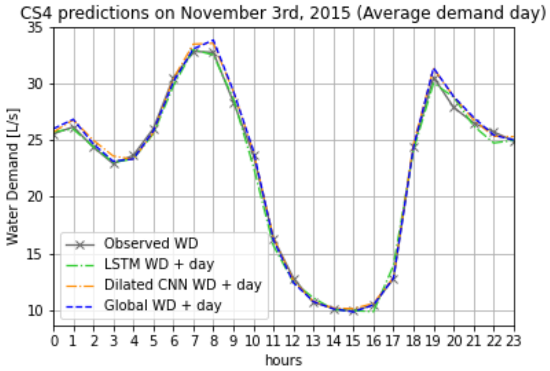
D. Figure 12. Low peak demand errors with the observed water demand over a 24-hour horizon for CS3.



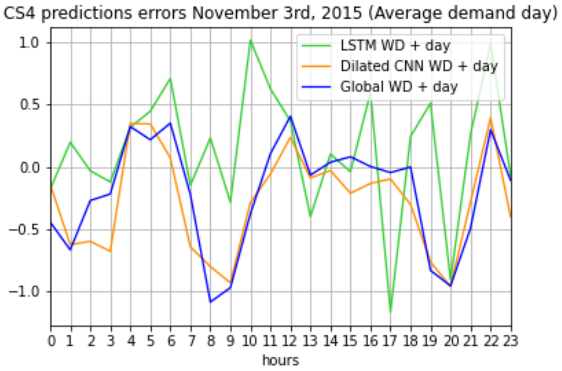
D. Figure 13. High peak demand predictions with the observed demand over a 24-hour horizon for CS4.



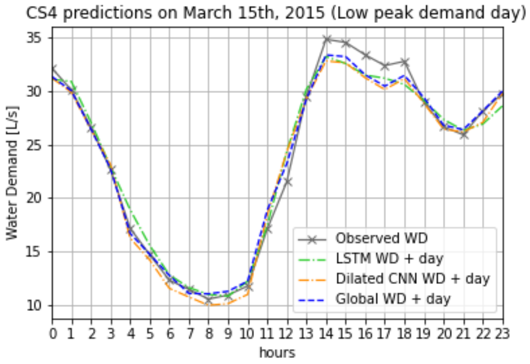
D. Figure 14. High peak demand errors with the observed water demand over a 24-hour horizon for CS4.



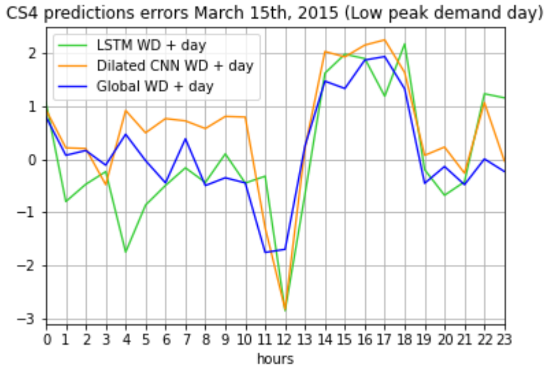
D. Figure 15. Average demand predictions with the observed water demand over a 24-hour horizon for CS4.



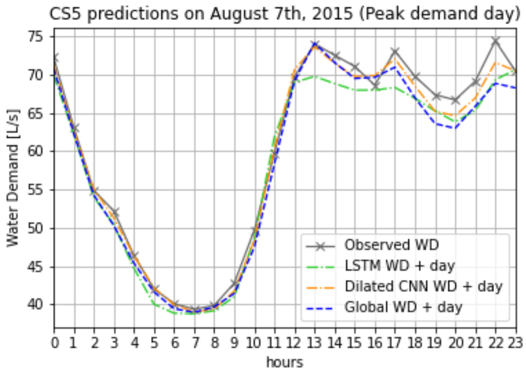
D. Figure 16. Average demand error with the observed water demand over a 24-hour horizon for CS4.



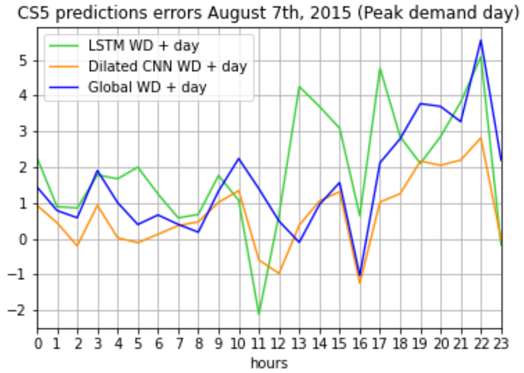
D. Figure 17. Low peak demand predictions with the observed water demand over a 24-hour horizon for CS4.



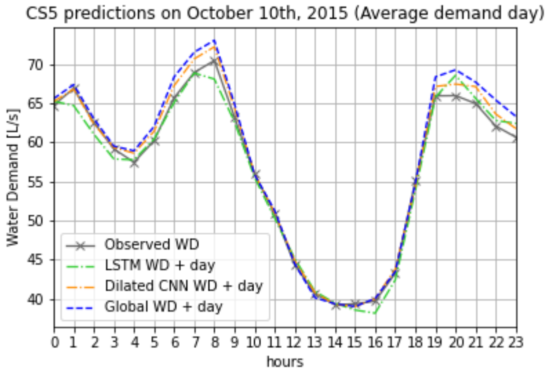
D. Figure 18. Low peak demand errors with the observed water demand over a 24-hour horizon for CS4.



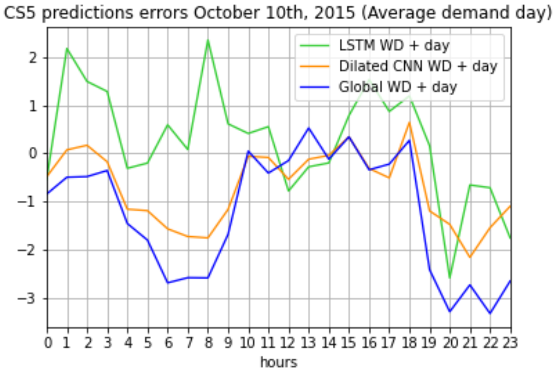
D. Figure 19. High peak demand predictions with the observed demand over a 24-hour horizon for CS5.



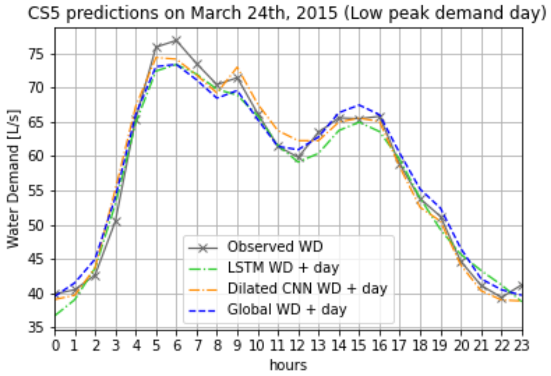
D. Figure 20. High peak demand errors with the observed water demand over a 24-hour horizon for CS5.



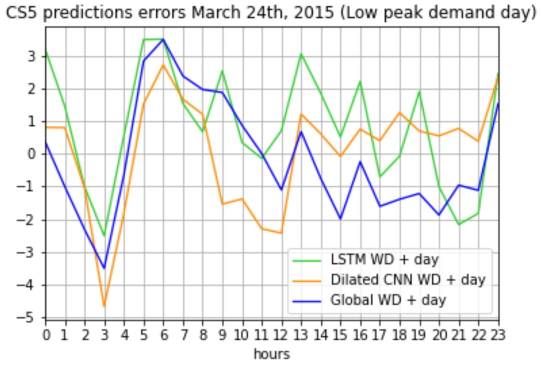
D. Figure 21. Average demand predictions with the observed water demand over a 24-hour horizon for CS5.



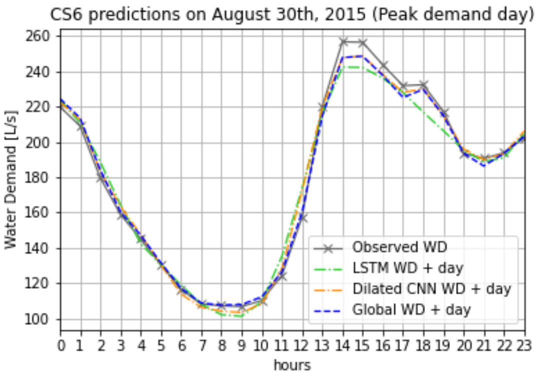
D. Figure 22. Average demand error with the observed water demand over a 24-hour horizon for CS5.



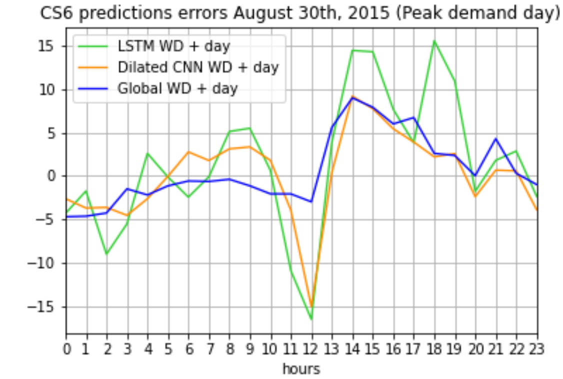
D. Figure 23. Low peak demand predictions with the observed water demand over a 24-hour horizon for CS5.



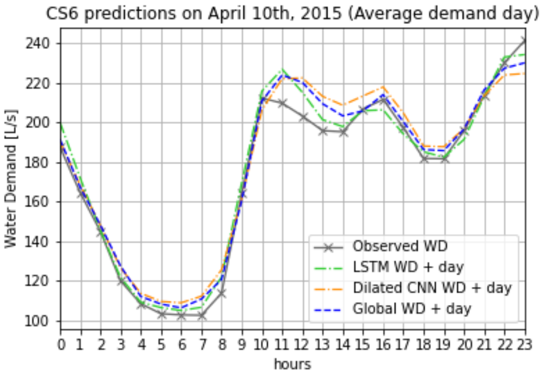
D. Figure 24. Low peak demand errors with the observed water demand over a 24-hour horizon for CS5.



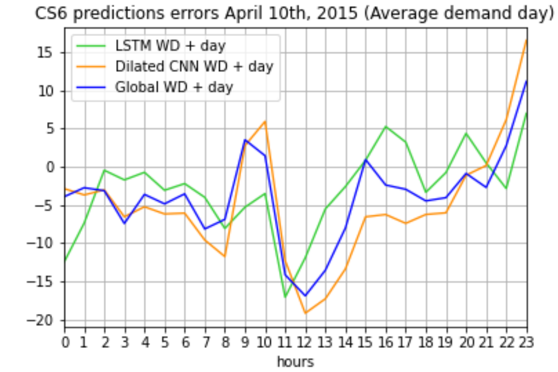
D. Figure 25. High peak demand predictions with the observed demand over a 24-hour horizon for CS6.



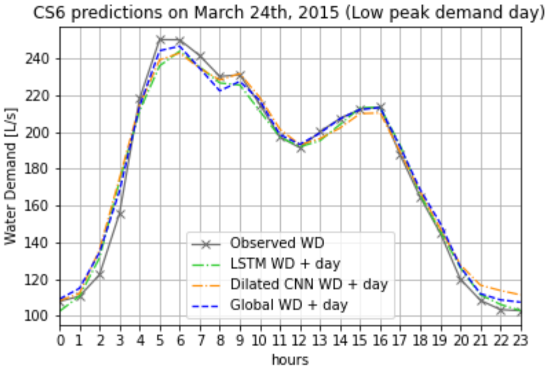
D. Figure 26. High peak demand errors with the observed water demand over a 24-hour horizon for CS6.



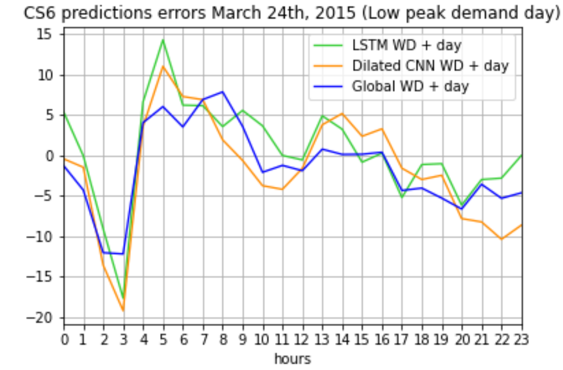
D. Figure 27. Average demand predictions with the observed water demand over a 24-hour horizon for CS6.



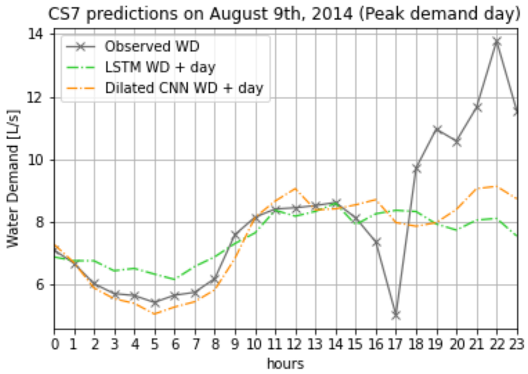
D. Figure 28. Average demand error with the observed water demand over a 24-hour horizon for CS6.



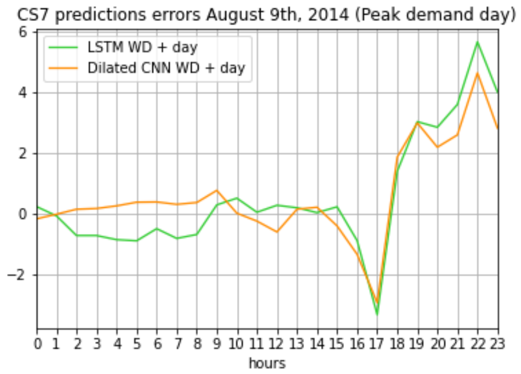
D. Figure 29. Low peak demand predictions with the observed water demand over a 24-hour horizon for CS6



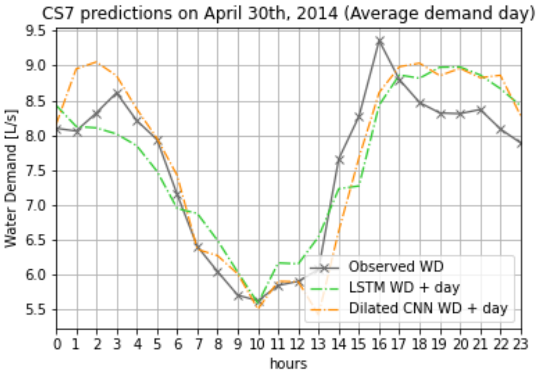
D. Figure 30. Low peak demand errors with the observed water demand over a 24-hour horizon for CS6.



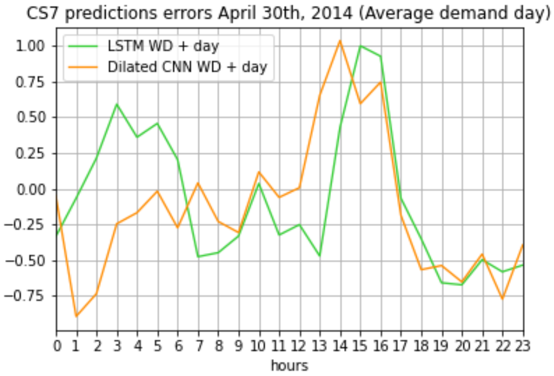
D. Figure 31. High peak demand predictions with the observed demand over a 24-hour horizon for CS7.



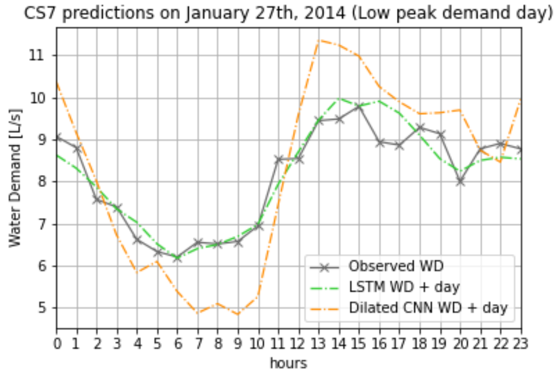
D. Figure 32. High peak demand errors with the observed water demand over a 24-hour horizon for CS7.



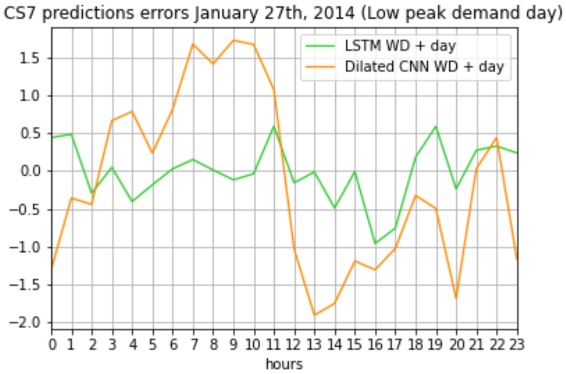
D. Figure 33. Average demand predictions with the observed water demand over a 24-hour horizon for CS7.



D. Figure 34. Average demand error with the observed water demand over a 24-hour horizon for CS7.



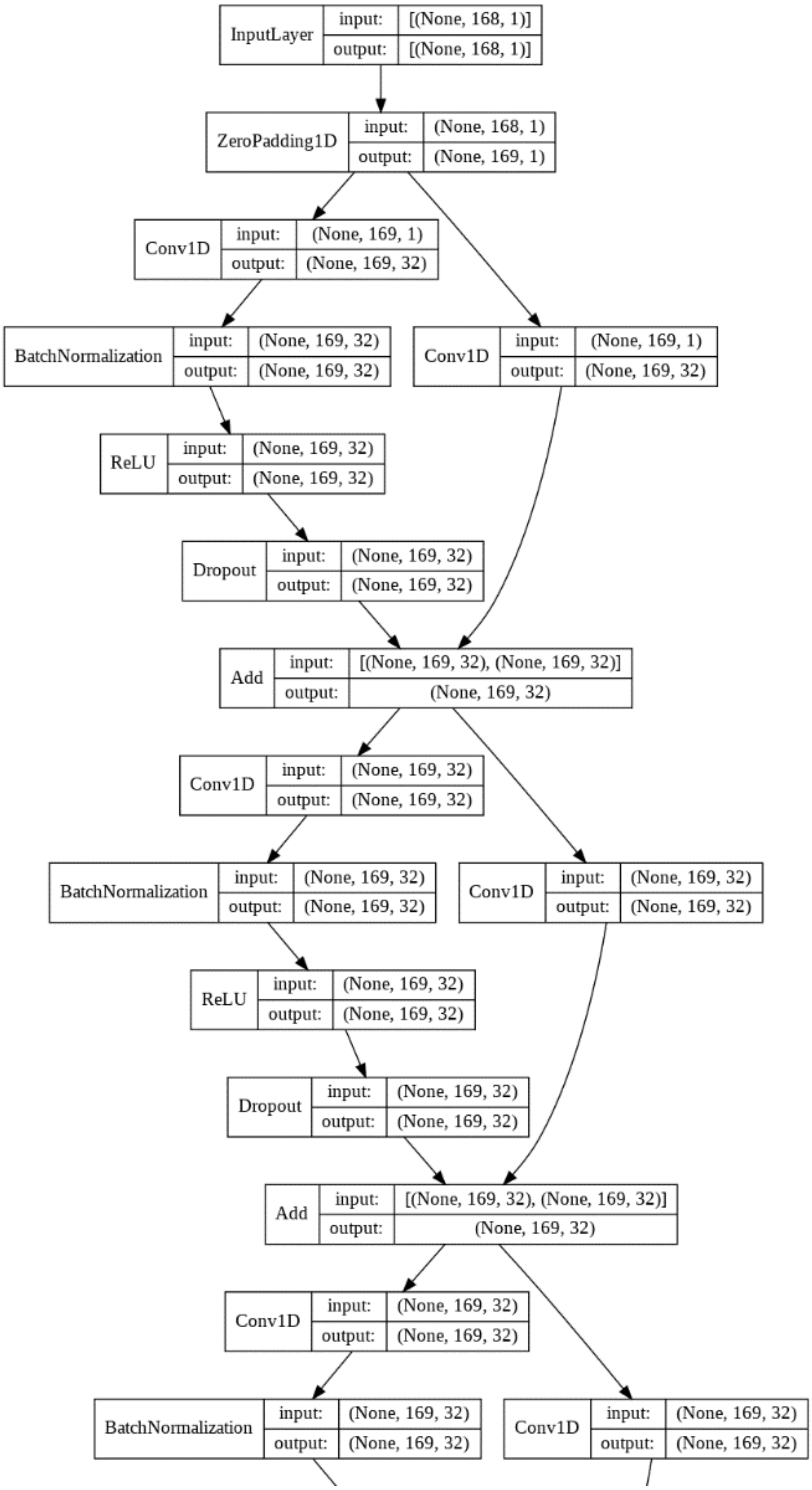
D. Figure 35. Low peak demand predictions with the observed water demand over a 24-hour horizon for CS7.

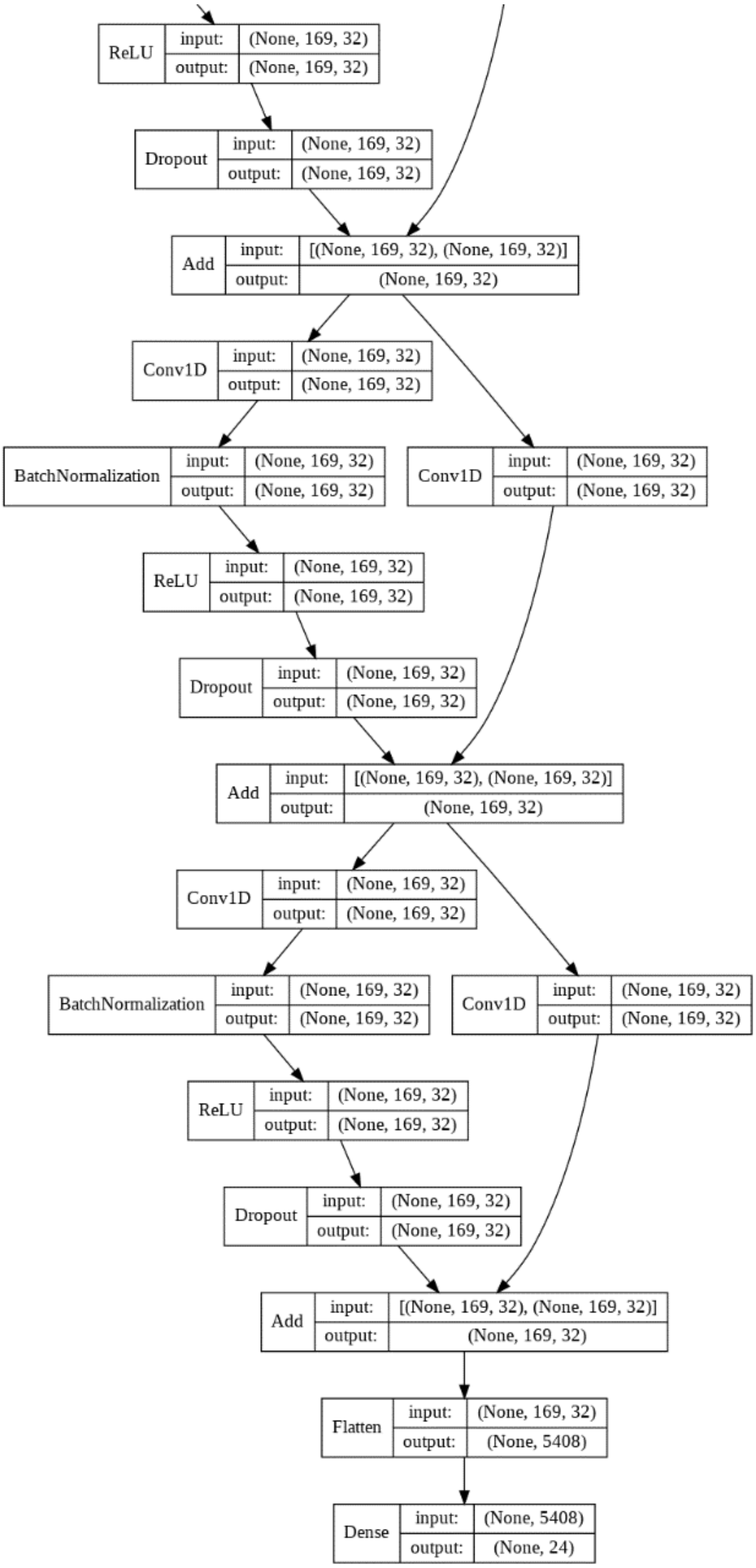


D. Figure 36. Low peak demand errors with the observed water demand over a 24-hour horizon for CS7.

Appendix E

The following flowchart illustrates in detail the architecture of the dilated CNN.





E. Figure 1. Dilated CNN architecture